



UNIVERSIDADE D  
COIMBRA

AUTONOMOUS OCCURRENCES

Tomás Gabriel Godinho Martins Lopes



UNIVERSIDADE D  
COIMBRA

Tomás Gabriel Godinho Martins Lopes

## AUTONOMOUS OCCURRENCES

Dissertation in the context of the Master in Informatics Engineering, Specialization in Software Engineering advised by Prof. Pedro Furtado and Eng. Francisco Monsanto and presented to the Faculty of Sciences and Technology / Department of Informatics Engineering.

September 2021

---

## Acknowledgements

Antes de mais, gostava de agradecer aos meus orientadores, Eng. Francisco Monsanto e Professor Pedro Furtado, pelos seus conselhos e sugestões. Gostava também de agradecer a todos na Ubiwhere que me ajudaram e acompanharam nesta etapa, especialmente aos meus colegas de estágio.

Para finalizar, um obrigado especial à minha família, namorada e amigos por toda a paciência e apoio durante esta jornada.

Tomás Lopes.

This page is intentionally left blank.

---

## Abstract

With the complexity of the different cities increasing globally, the need for them to fix occurrences like incidents in public roadways or maintenance in city infrastructures also increased, leading to a necessity for solutions that can manage the before-mentioned problems in a more autonomous and faster way. The problem with incidents is not only to locate them but also to organize them and ensure that they follow a specific flow to achieve the final result that can vary depending on what type of issue is considered. The focus of this internship is to create a system to aid the process since an occurrence is registered until it is resolved. The proposed solution must integrate the existing methods to make those reports so that the already familiar applications used to obtain the information about the occurrences will not get obsolete when connected with the management system proposed. The idea of using a workflow engine will help with the automation of these processes since some office tasks that usually require human force can become autonomous activities. Despite this attempt, it is impossible to make every task autonomous considering that physical tasks will most of the time require a human factor. This internship will focus on making and integrating three applications that plan to integrate autonomous tasks and the ones that require the human workforce. One of these applications will be to register occurrences that the citizens report in their street, another for the system admins to manage the occurrences and the workers, and the last one for the workers to claim the tasks that require their assistance to be completed. The intention of integrating all of this with a workflow engine is due to many of these processes being statics for each type of occurrence and will increase the efficiency and effectiveness of their executions improving the citizens' quality of life.

## Keywords

Occurrences management, workflow engines, smart cities



This page is intentionally left blank.

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Context and Motivation . . . . .	1
1.2	Objectives . . . . .	2
1.3	Document Structure . . . . .	2
<b>2</b>	<b>State of the Art</b>	<b>5</b>
2.1	Smart Cities . . . . .	5
2.1.1	Smart Cities Solutions . . . . .	6
2.2	Open Standards . . . . .	12
2.2.1	Open311 and Open511 . . . . .	13
2.2.2	FIWARE . . . . .	14
2.3	Occurrence Management . . . . .	14
2.3.1	Current Solutions . . . . .	14
2.3.2	Conclusion . . . . .	18
2.4	Workflow Engines . . . . .	18
2.4.1	Existing Market and Solutions . . . . .	20
2.4.2	Bonita BPM . . . . .	23
2.5	Final thoughts . . . . .	24
<b>3</b>	<b>Planning and Methodology</b>	<b>26</b>
3.1	Process Management . . . . .	26
3.2	Tools . . . . .	27
3.3	First Semester . . . . .	27
3.4	Second Semester Planning . . . . .	28
3.5	Gantt Diagram . . . . .	29
3.6	Process Plan and Deviations . . . . .	29
3.7	Risk Management . . . . .	30
3.7.1	Internship Success Criteria . . . . .	30
3.7.2	Impact Scale . . . . .	30
3.7.3	Probability Scale . . . . .	30
3.7.4	Risk Management Analysis . . . . .	32
<b>4</b>	<b>Requirements</b>	<b>36</b>
4.1	Terminology . . . . .	36
4.2	User Stories Structure . . . . .	37
4.3	User Stories . . . . .	38
4.3.1	As a Citizen . . . . .	39
4.3.2	As an admin . . . . .	42
4.3.3	As a worker . . . . .	46
4.4	Non-Functional Requirements . . . . .	48
4.5	Restrictions . . . . .	49

<b>5</b>	<b>Architecture And Technology</b>	<b>51</b>
5.1	Terminology . . . . .	51
5.2	Architecture . . . . .	51
5.3	Technology . . . . .	55
<b>6</b>	<b>Development</b>	<b>57</b>
6.1	Workflow Engine . . . . .	57
6.1.1	Process Definition . . . . .	58
6.1.2	Workflow Engine metrics . . . . .	58
6.2	Django Application . . . . .	59
6.2.1	Models . . . . .	59
6.2.2	REST API . . . . .	60
6.2.3	Django Admin . . . . .	62
6.3	Front End . . . . .	62
6.3.1	Mock-ups . . . . .	63
6.3.2	Application Overview . . . . .	64
6.4	Environment . . . . .	65
6.4.1	Virtualization . . . . .	65
6.4.2	Deployment . . . . .	68
<b>7</b>	<b>Testing</b>	<b>71</b>
7.1	Unit Testing . . . . .	71
7.2	Functional Testing . . . . .	73
<b>8</b>	<b>Final Product</b>	<b>75</b>
<b>9</b>	<b>Conclusion &amp; Future Work</b>	<b>81</b>
9.1	Work done . . . . .	81
9.2	Challenges . . . . .	82
9.3	Future Work . . . . .	82
9.4	Lessons Learned . . . . .	83
<b>10</b>	<b>Annexes</b>	<b>93</b>
10.1	Development . . . . .	94
10.2	Tests . . . . .	95
10.2.1	Unit Testing . . . . .	95
10.2.2	Functional Testing . . . . .	98

This page is intentionally left blank.

# Acronyms

- API** Application Programming Interface.
- BPEL** Business Process Execution Language.
- BPM** Business Process Management.
- BPMN** Business Process Model and Notation.
- CI/CD** continuous integration/continuous delivery.
- CLI** command-line interface.
- CMMN** Case Management Model and Notation.
- DAGs** directed acyclic graphs.
- DMN** Decision Model and Notation.
- FMS** FixMyStreet.
- GPS** Global Positioning System.
- HPC** High-Performance Computing.
- HTTP** The Hypertext Transfer Protocol.
- IDE** Integrated Development Environment.
- INCT** National Science and Technology Institute.
- IoT** Internet of Things.
- JSON** JavaScript Object Notation.
- JVM** Java Virtual Machine.
- NGSI** Next Generation Service Interface.
- ORM** Object-relational-mapping.
- REST** Representational State Transfer.
- RPC** remote procedure call.
- SDK** Software development kit.
- UFRN** Federal University of Rio Grande do Norte.
- UI** User Interface.
- UML** Unified Modeling Language.

**XML** Extensible Markup Language.

**YAML** Yet Another Markup Language.

This page is intentionally left blank.

# List of Figures

2.1	Lisboa Inteligente project home page . . . . .	7
2.2	Smart cities Initiatives in Singapore . . . . .	8
2.3	Benefits for citizens and businesses in the Singapore Smart Nation Project . . . . .	9
2.4	Main page of Application to register occurrences in Amsterdam . . . . .	10
2.5	Bloomington city services web page . . . . .	11
2.6	Bloomington datasets webpage . . . . .	11
2.7	demonstration of InterSCity Platform. Bus Application and Dashboard . . . . .	12
2.8	Open511 Standard applied in Lousiana city . . . . .	13
2.9	Submission report exemple NaMinhaRuaLX . . . . .	15
2.10	Occurrence Search NaMinhaRuaLX . . . . .	15
2.11	Reports on the FixMyStreet application, Manchester . . . . .	16
2.12	SIGOc Simplified Architecture . . . . .	17
2.13	State Machine Workflow Example . . . . .	19
2.14	Components of a Workflow Engine . . . . .	19
2.15	Camunda Workflow engine and Integration Representation . . . . .	22
3.1	Representation of the planned tasks though a Gantt Diagram . . . . .	29
3.2	Tasks duration represented though a Gantt Diagram . . . . .	29
3.3	Risk Exposure Matrix . . . . .	33
5.1	Context Diagram . . . . .	52
5.2	User Integration Architecture . . . . .	53
5.3	Back end detailed proposed Architecture . . . . .	54
5.4	Back end detailed final Architecture . . . . .	54
5.5	Number of npm downloads angular and react . . . . .	55
6.1	Relationships between the models of the back end application . . . . .	59
6.2	Diagram representing the process of obtaining the process id from Camunda . . . . .	60
6.3	Diagram representing the process of obtaining the activity id using the process id . . . . .	61
6.4	Worker application login page mock-up . . . . .	63
6.5	Worker application dashboard mock-up . . . . .	64
6.6	Worker application occurrence description and possible actions mock-up . . . . .	64
6.7	Docker vs Virtual machines . . . . .	66
6.8	Django application dockerfile . . . . .	67
6.9	React application dockerfile . . . . .	68
6.10	Django application service in docker-compose file . . . . .	69
6.11	Camunda workflow engine service in docker-compose file . . . . .	69
7.1	Mock request exemplification in test case . . . . .	72
8.1	Log in page of the front end application . . . . .	76



8.2	Dashboard of the front end application . . . . .	76
8.3	Details of the selected occurrence and possible actions . . . . .	77
8.4	Dashboard of the front end application with tasks claimed . . . . .	77
8.5	Notifications in the dashboard of the front end application . . . . .	77
8.6	Django admin main page . . . . .	78
8.7	Occurrences represented in the Django admin . . . . .	78
8.8	Creation of a model in the Camunda modeler . . . . .	79
8.9	BPMN model represented in the workflow engine with several instances . . . . .	79
8.10	Grafana dashboard with multiple metrics . . . . .	80
10.1	Detailed relationships between the models of the back end application . . . . .	94

This page is intentionally left blank.

# List of Tables

- 2.1 Workflow Engines comparison table . . . . . 24
  
- 3.1 Risk Number 1: Lack of Experience . . . . . 32
- 3.2 Risk Number 2: Pandemic Situation . . . . . 32
- 3.3 Risk Number 3: Scope of the project not clear . . . . . 32
- 3.4 Risk Number 4: Predicted tasks take longer than expected . . . . . 33
- 3.5 Risk Number 5: Scope of the project too wide . . . . . 33
  
- 4.1 User Stories Overview . . . . . 39
- 4.2 Scenario 1 - Usability . . . . . 48
- 4.3 Scenario 2 - Performance . . . . . 48
- 4.4 Scenario 3 - Performance . . . . . 49
- 4.5 Scenario 4 - Maintainability . . . . . 49
- 4.6 Scenario 5 - Security . . . . . 49
- 4.7 Scenario 6 - Security . . . . . 49
  
- 7.2 Implemented requirements . . . . . 74

This page is intentionally left blank.

# Chapter 1

## Introduction

### 1.1 Context and Motivation

An occurrence can be any incident related to the city, which usually does not influence just one individual [7]. Thus, an occurrence can be anything that happens in a community that bothers the ordinary citizen such as potholes, necessary maintenance in public buildings, or simpler things like full waste bins. Urban occurrences are a topic that concerns many municipal employees and other entities that see their day to day affected and since a significant part of people is concentrated in urban areas, the subject is gaining more importance because, in these areas, a single occurrence can disturb not only one but multiple people increasing the level of importance of each occurrence.

Citizens usually report the occurrences in the cities to their local officials to see them fixed, but sometimes this process can be very time demanding, and with the types of lives that nowadays the families live, it becomes impossible for some people to report these issues. Considering this, some city officials implemented methods to facilitate submitting these problems by using other channels such as online pages or mobile applications. This example of citizen engagement helps city officials keep track of the existing issues to improve these situations.

Although an excellent step forward, these applications only solve part of a much bigger problem. In many cases, this creates a bottleneck in another step of the process because of poorly-designed methods to organize the information once it reaches the right entities sometimes forcing someone to rewrite reports into a database or another repository where the information will be later stored and transmitted to the entities and the workers that will resolve these occurrences. In order to resolve this, protocols like the open311 and open511 by OpenPlans[113] propose standardized APIs to allow connecting these applications of ticket submission directly to the city management database, standardizing the data through open-source data models. These solutions were innovative and helped increase the number of reported occurrences, but once more revealing other situations since many of the tasks of these static processes were still processed manually. This internship plans to tackle these tasks that can become autonomous, and help to automatically provide information to the workers or entities that will fix the man-required jobs making the process of resolving the occurrences more autonomous and efficient.

## 1.2 Objectives

This internship's objective is to create a platform where occurrences can be gathered and managed according to specific workflows, which may depend on a number of factors such as the type of occurrence, urgency and others. The platform will manage these occurrences using an existing workflow engine. A Workflow Engine can automate defined processes and follow their logic until the end of the process obtaining information and performing actions each step of the way. This provides efficiency to the system since the processes to fix the occurrences are often confusing and lengthy. Thus a workflow engine will automate the flow in the way that every step of the process is described in it and there is a coherent direction from one step to another, making the processes more autonomous.

The platform must communicate with the people responsible to verify the occurrences and manage their logistic via a user interface to inform what is being done and show information about the occurrences and the workers that are responsible for each of the occurrences. Some standards will be implemented, in specific the open311 to allow communication with other applications such as Ubiwhere's Urban platform, and others already in use by municipalities where the solution could be applied. The system will consist of an application to communicate with users for them to report the occurrences, a backend application that with the assistance of an existing workflow engine will automate the processes of resolving the multiple types of occurrences that will be described by the system admin and integrated into the workflow engine, a worker application will be developed for them to claim and mark as completed the tasks that they will resolve. This internship was followed by Ubiwhere, a software company based in Aveiro with offices in Coimbra and São João da Madeira, Aveiro. The company specializes in research and innovation in Smart Cities, IoT, and the Internet of the Future

## 1.3 Document Structure

This thesis is organized into nine chapters. The introduction, chapter 1, is where context is given for the problem in hand and the value of this thesis and its proposed solution are explained.

State of the art, chapter 2, the existing literature is stated and tools that served as inspiration for the project. The literature and tools explored gave some knowledge of what exists in the market to solve the problem.

The chapter named Planning and Methodologies, chapter 3, is where the plan of activities is made for the duration of the internship and the tools that the intern used.

The chapter of requirements, chapter 4, is where the gathered requirements are explained using user stories. This chapter is also where the non-functional requirements are exposed.

The chapter of Architecture and Technologies, chapter 5, is where the architecture of the system that was built is defined as well as the technologies that were used to accomplish the final system.

The chapter of Development, chapter 6, is where the development of the proposed system is described and the decisions made to obtain the final product. The development environment is also here described, showing how the system operates all together.

The chapter of Testing, chapter 7, The validation of the system is present and is showed how the validation process was made.

The chapter of Final Product, chapter 8, provides an overview of the system as a whole and of its main functionalities

The last chapter, chapter 9, is where a conclusion about the work is made and a scope is given about the Future work that is possible from the created system.

This page is intentionally left blank.



## Chapter 2

# State of the Art

This chapter results from research done to better understand what was done until this moment in the topics relevant to this work. It starts by introducing smart cities and analyzing what this term means and how it can be perceived in the community giving a context of where our approach would fit. After this, different solutions that address the problem of occurrence management in cities are presented and discussed. The reason to mention these specific cases is that they already explore some solutions that are similar to the goal of this project, making them good examples of what will be made and how the concept could be implemented.

Different standards to represent occurrences have been studied as well as their potential value offered to the platform in the communication with the existing applications allowing the collection of data from different points and sources.

In the section on workflow engines, various products, and their different approaches and features are presented, discussed, and evaluated to make the better choice possible for the problem stated. In the last section, a summary will be made of all the referred topics, and a conclusion will also be drawn about the referred workflow engines leading to a decision to what tool will be used.

### 2.1 Smart Cities

The concept of Smart Cities is not trivial to define because there is more than one approach to this concept. One definition is that a "Smart city is a high-tech intensive and advanced city that connects people, information and city elements using new technologies to create a sustainable, greener city, competitive and innovative commerce, and an increased life quality" [48]. Another definition of a smart city is that it is a city that monitors and integrates pieces of information of all of its critical infrastructures can better optimize its resources, plan its preventive maintenance activities, and monitor security aspects while maximizing its citizens' services[49]. Although it is a broad concept and plenty of definitions exist [47], the common element between these definitions and many others is that it implies the "use of information" [1].

With the passing years, this concept transformed itself to mean "any form of technology-based innovation in the planning, development, and operation of cities"[1]. The need for technology does not imply that every technology used in the smart cities approaches should be state of the art since the concept mentioned here does not expect it. Some new

technologies are not so reliable as technologies that have been around for some years and have some base users working with them. No matter what kind of technology is used, the common objective in every methodology of smart cities is “to improve the quality of city services and the lives of its citizens”[5].

With the growth of smart cities initiatives globally, large amounts of data need to be analyzed and “processed in a meaningful way according to the given requirement” [4] to extract meaningful information. With this information about the current condition of certain parts of the cities like traffic or other infrastructures, it can be possible to predict the need for repairs and maintenance, improving public authorities’ effectiveness. [6]

Responding and resolving occurrences that occur in cities is essential. For this, the smart cities initiatives usually implement applications to facilitate the send of reports. Citizen engagement is a form of interaction between citizens and their governments [77]. The way that the reports are collected can be viewed as a form of citizen engagement where the citizens are incentivized to engage to have their needs fulfilled.

“Cities should be based on a network of multiple systems, all of them being closely connected with citizens’ needs”[6]. For this reason, there must be systems in place that allow ordinary citizens to speak up more easily about their day to day problems that many times pass as unknown for cities management mechanisms. In some countries, like Portugal, there “are projects to start managing occurrences between the police, firemen and civil protection”[7], which shows a growing concern in this subject by smart cities initiatives. Since these systems are usually part of Smart Cities initiatives with larger scopes, it is relevant to study these initiatives in the context of this project.

### 2.1.1 Smart Cities Solutions

Smart cities initiatives are expanding and helping city officials to improve the quality of life of their citizens. Some examples of this are Barcelona and Lisbon and so many others that become examples of future cities. Many smart cities solutions integrate some form of occurrence management , since the problem is applicable to all of the world’s cities. The initiatives mentioned here are good examples of where the proposed solution could fit in.

#### 2.1.1.1 Barcelona Smart City

Barcelona’s urban transformation dates back to the 1980s. A significant change occurred in these years that took the city from an economic crisis to become a leading metropolis [50]. The main aspects of Barcelona’s smart city strategy are six: Smart Districts, Living Labs Initiatives, Infrastructures, and New Services for the Citizen, Management of Smart city initiative, and Open Data.

Barcelona transformed itself from an industrial area to the house of new innovative companies. [51] In this process, the city came up with a concept of smart cities itself that was "a self-sufficient city of productive neighborhoods at human speed, inside a hyper-connected zero-emissions metropolitan area" [52]. Barcelona chose to use new technologies and infrastructures to improve economic growth and increase its citizens’ quality of life. To make this change, Barcelona relied on partnerships with some businesses and universities, which were proved highly effective.[52] In this case, the cooperation with other cities in Spain and the European Union support was essential in collaboration and funding.

Software such as the City OS, which consists in a decoupling layer between data sources and

smart cities solutions [53], emerged in Barcelona,. Smart Citizen is an open data platform that aims to generate participatory processes in the city.[53] A project called the Urban Mobility Plan is set to ensure “safe, sustainable, equitable, and efficient mobility”[54].

Barcelona’s Dab Labs are classrooms where citizens can learn about the technologies that are being developed. These labs allow citizens to access the information and the necessary tools to innovate technologically and enable their engagement in the smart city projects present in the city [54].

Fed by IoT devices, other initiatives were also created benefiting from the real-time information generated by them. An example of this is the parking spaces. Barcelona implemented a system where sensors were placed underneath the asphalt to identify the free spots. The program reduced emissions and congestion by directing drivers to available parking spaces. This measure, complemented by an application called L’apparkB[116], allowed that 4000 permits for parking were issued per day.

### 2.1.1.2 Lisbon

Another example of a smart city initiative is Lisbon’s project “Lisboa inteligente” that is composed of multiple scopes.

In the year 2020, Lisbon became the European Green Capital Award 2020 [12] and was considered a reference to other cities in the European Union[14] to demonstrate that a smart city can also be low tech[15]. To continue their work and keep improving the city’s quality of life, a project called "Lisboa inteligente" emerged. This project joined the different ways of enhancing this city by digitizing it and making it easier for citizens to access what is happening in real-time.

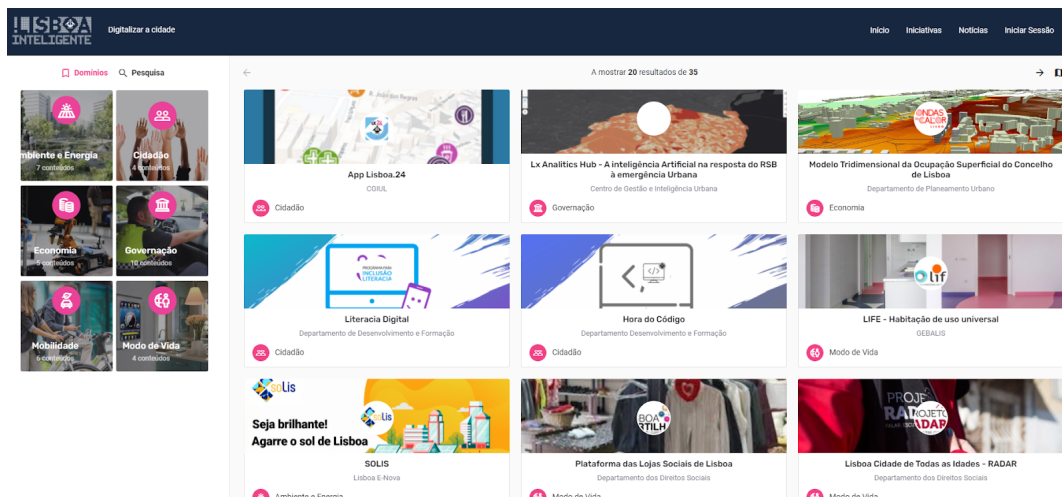


Figure 2.1: Lisboa Inteligente project home page

This initiative had the objective of improving the city in six ways: Environment and Energy, Citizen, Economics, governance, mobility, and way of life [10]. As demonstrated on their website[43], the initiatives presented tend to be technology and human-focused because one of their main objectives is, as mentioned before, improving the quality of life of their citizens.

Lisbon implemented an urban data platform to deal with the aspects of urban governance. [16] This project is connected with IoT devices to collect data and has developed the Smart Management Platform of Lisbon to integrate it[16]. This project is the result of

the partnership that the city has developed with NEC [17]. This project uses FIWARE open source building blocks and other data available through open standards. [16] To report occurrences, the city came up with an application called NaMinhaRuaLx that will be discussed further ahead.

### 2.1.1.3 Singapore

The southeast Asia city-state is the world's second-most densely populated in the world.[91] To maintain and evolve this very populated city a smart initiative was developed. This initiative is divided, like the ones previously mentioned, into six domains: Strategic Nacional Projects, Urban Living, Transport, Health, Digital Government Services, and Startups and Business. Each of these domains has a specific focus and object. To assist these projects, several applications were developed to provide more contact and interaction with the population.

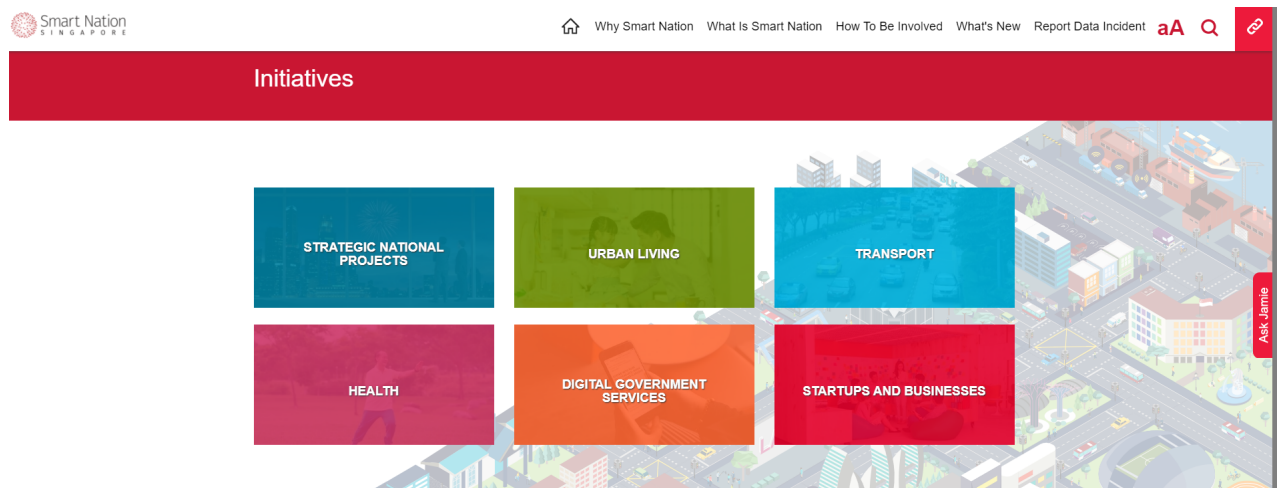


Figure 2.2: Smart cities Initiatives in Singapore

This evolution has plenty of benefits for enterprises as well as for the common citizens, these benefits are specified in the website of the Singapore Nation Initiative [92] and specified in the following image.

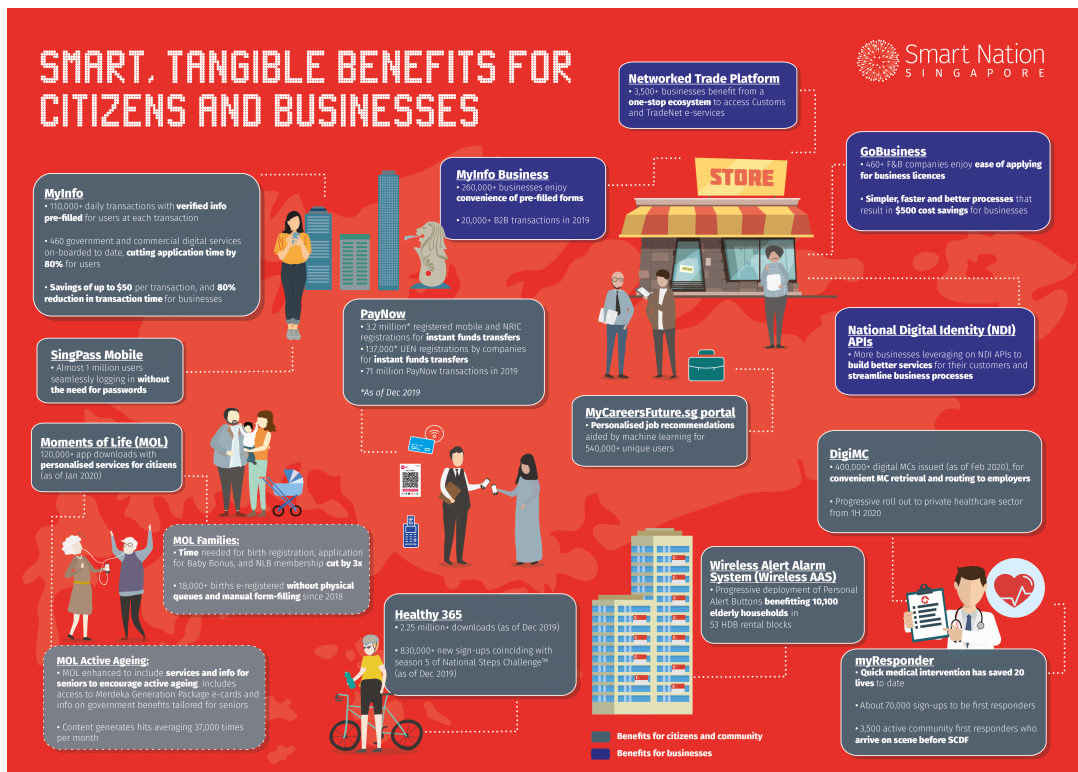


Figure 2.3: Benefits for citizens and businesses in the Singapore Smart Nation Project

This transformation is not complete and Singapore has more ideas and solutions to achieve its objectives and transform the city in order to make it smarter. Since the majority of the population lives in public housing, government agencies are working with private firms to test smart home technologies, such as home energy and water management systems and monitoring systems for the elderly.[91] By 2022, the government plans to implement intelligent, energy-efficient lighting for all public roads, and have solar panels installed on rooftops of 6,000 buildings.

#### 2.1.1.4 Amsterdam

Amsterdam is thought to be a world leader in smart city development [107]. This project came to action in 2008 and has grown to be one of the largest in Europe and it became 2016 the European Capital of Innovation by the European Commission. Amsterdam smart city project has an online platform that consists of a partnership of twelve public, private, and universities partners that acts as a forum to share and communicate ideas where later the idea can match up with some project initiator which can be companies, start-ups, government agencies, universities, research institutions or private citizens. [107] This makes it possible to verify some ideas and convert them into real-life projects. The project is divided into six different categories: Infrastructure and Technology, Energy, Water & Waste, Mobility, Circular City, Governance & Education, and Citizens& Living each one of them with different projects with more than 4000 active members and 240 in various stages of development.

One of these projects is meant to register and sort occurrences. The system used to be a collection of drop-down menus where the user registered the category where occurrence belonged to [108]. Although a clever solution that allowed for more occurrences to be registered it did not resolve the problem due to the municipality being a complex organi-

zation, many times the occurrence would be wrongly categorized. To resolve this situation a software that automatically sorts the occurrences by keywords was developed to prevent that the situation would repeat which decreased the time to process an occurrence due to being sorted correctly at the first attempt [108].

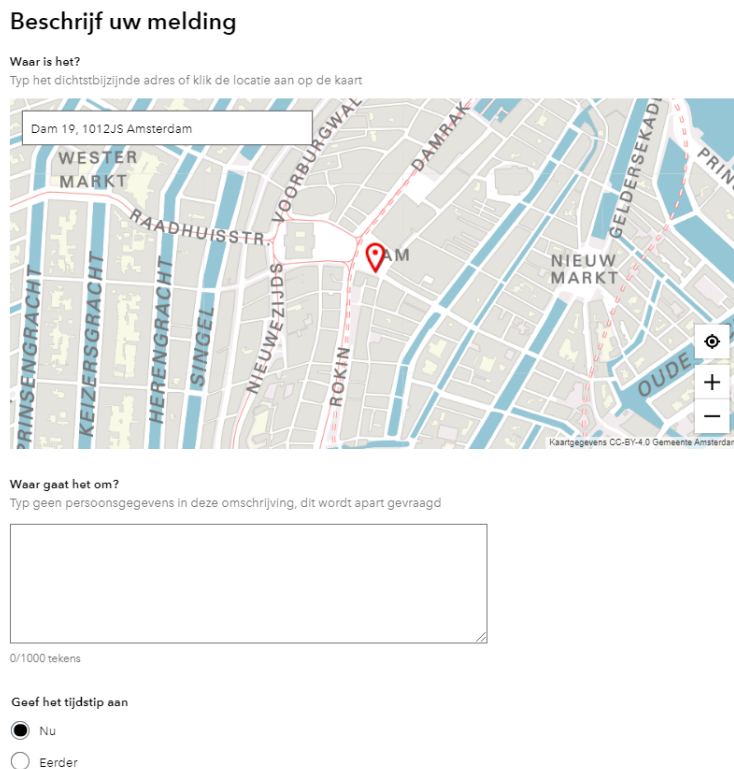


Figure 2.4: Main page of Application to register occurrences in Amsterdam

Also thanks to the system that detects which occurrence category the reported incident belongs to, the process of registering an occurrence also became easier without dashboards to identify the type and category which provides a better end user experience to the citizens.

### 2.1.1.5 City of Bloomington, Indiana

Although Bloomington is not seen as a one of the leading smart cities in the world, it has some interesting programs and also produces data that can be used as data sources for other projects. This city implements the standard open311 to various situations in specific using the project uReport to report occurrences that occur in the city.

Many of the data produced by the city programs is available to the public in the form of open data [109]. This is not only good for citizens once it is provided information on what the city is doing and how it is doing it, but as well as other entities that require data to test and verify their applications. This makes Bloomington a small smart city that produces data and uses some standards and applications to simplify their processes and in that way also simplify the life of their citizens.



Figure 2.5: Bloomington city services web page

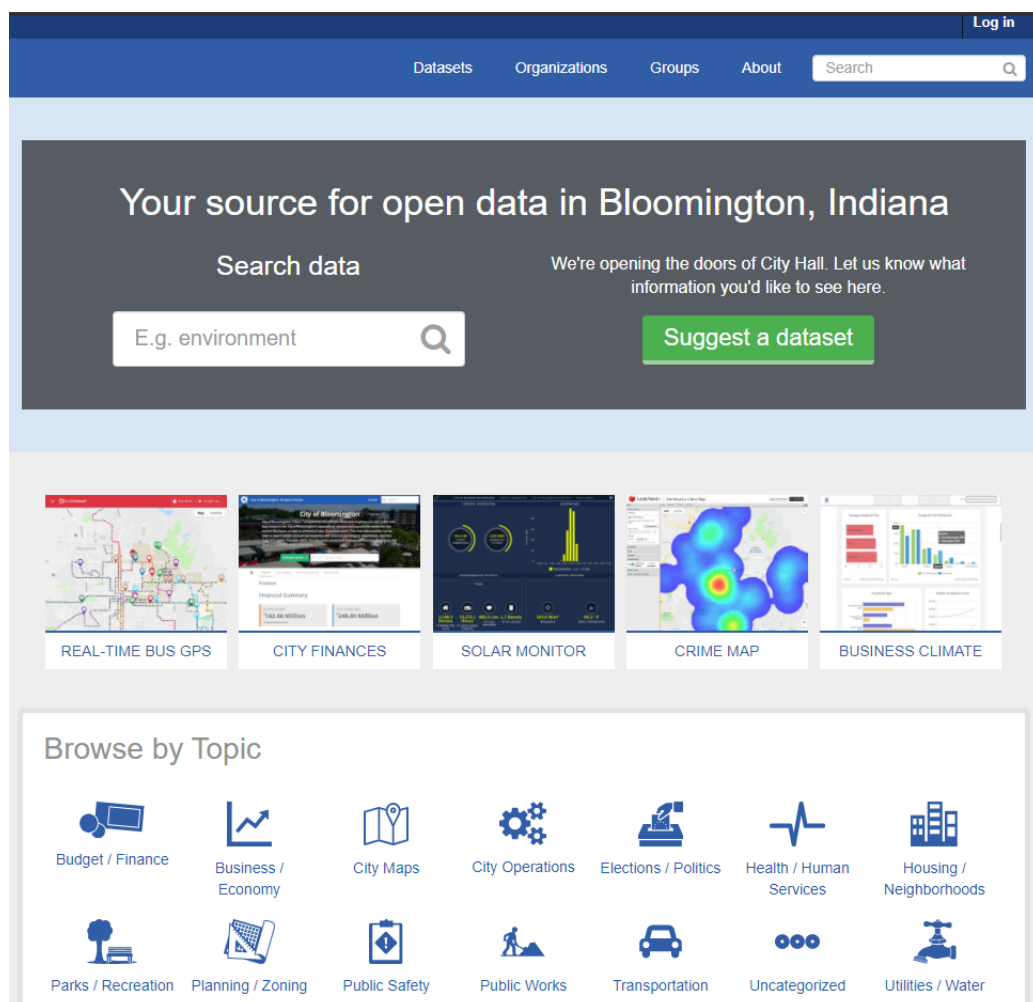


Figure 2.6: Bloomington datasets webpage

As seen in the image above, image 2.6, the produced data is categorized in topics to make it easier to search for the intended dataset. Once a topic is open, the website shows how many datasets are available in it and allows users to download the information.



### 2.1.1.6 InterScity

InterSCity is a collaborative research project hosted by the National Science and Technology Institute (INCT) of the Future Internet for Smart Cities. This project is composed of 9 Brazilian institutions as well as other international partners. InterScity project plans to develop an integrated open-source platform that contains the primary building blocks for future smart cities. [18].

One of the developments in this project was creating an app using machine learning and HPC (High-Performance Computing) techniques to predict the times of bus arrivals in over 20,000 bus stops in São Paulo, Brazil. The same technology was extended as a Real-Time Bus Dashboard system to manage the 15,000 bus fleet. [19]

Like the Lisbon project, InterScity also uses IoT devices for obtaining data in the cities where it is allocated [21]. To establish the link between the IoT devices and smart city applications, a platform, The InterScity platform, relies on APIs to provide the information necessary.[22] This project aims to provide a series of open-source final products [20] to be a basis for other projects that focus on smart cities or the Future Internet and datasets to test and serve as experimentation for other implementation of smart cities solutions.



Figure 2.7: demonstration of InterSCity Platform. Bus Application and Dashboard

## 2.2 Open Standards

As can be seen by the number of solutions that have been explored in this State of the Art, many systems are trying to solve the problem of resolving occurrences. This means that it is very important to have different systems and applications communicating in a standardized way. To make it easier to integrate and centralize information regarding occurrences, some standards like Open311 and Open511[113] were developed.

A standard is a repeatable, harmonized, agreed, and documented way of doing something [76]. Standards are necessary when discussing the integration of an application that gathers information from multiple data sources. To perform this communication, the software must communicate via the same standard to understand the messages traded between them. Some standards are already established to share occurrences, such as Open311, Open511, and FIWARE. These standards standardize communication channels via their APIs, including ways to standardize how the information is stored.



## 2.2.1 Open311 and Open511

Open311 standardizes the way that occurrences get to city officials. Applications can communicate to APIs implementing this protocol once the requirements are fulfilled [30][31][32]. Thanks to standardizing the communication between channels, the cost of future integration can be minimized. It can also allow third-party applications to send information to one system that centralizes all the requests.

Open311 works with a client and server model available through HTTP(S) requests [29], where the server is connected to a database to centralize such requests. The big picture idea is not just to be able to report incidents but also do a series of county tasks like paying some bills and renew parking licenses [34][35][36]. The information obtained by this method is sometimes used to create some datasets to study specific topics about city management systems[33] or to generate useful data to determine what are the weak and robust features of the city workforce.

The same way of thinking also led to the creation of Open 511, which applies the same concept more specifically to road issues. The standard can inform and collect several pieces of information and integrate third party applications like Waze, an Global Positioning System (GPS) mobile application that allows for its users to report road occurrences[82], to ensure the last to date information.

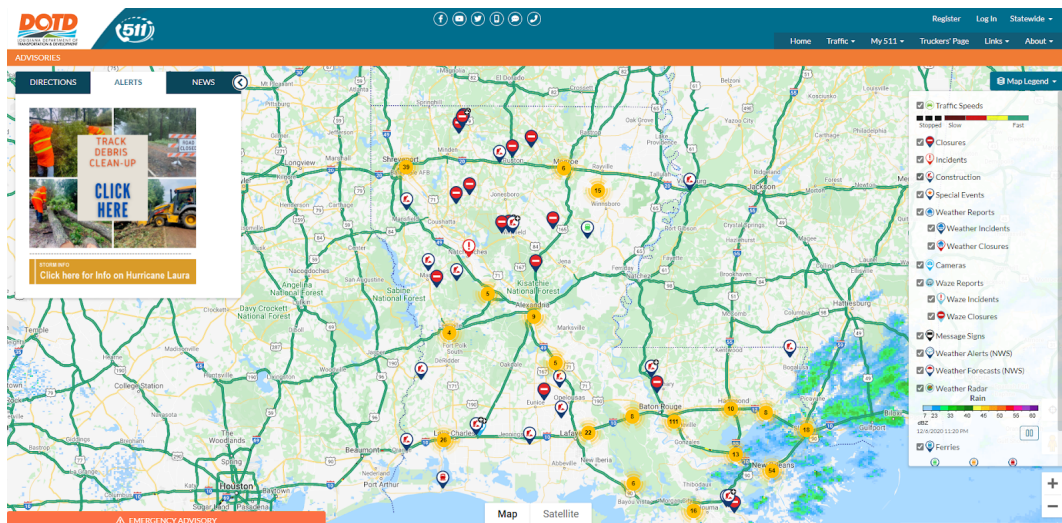


Figure 2.8: Open511 Standard applied in Louisiana city

As demonstrated in figure 5, this standard can help collect the information that should be of public knowledge and centralize it. The core components of the Open511 are the existence of a single format that allows for multiple applications to send data into it as long as it fulfills its requirements such as API to communicate to the exterior, and a web app or mobile application to show and receive the events [37]. The application or web app can be anything that can send information, including social media applications or others used to guide our way like Waze and google maps that many times people already pinpoint those problems.

## 2.2.2 FIWARE

FIWARE is a European project that evolves multiple ideas and concepts that aim to improve the quality of life of citizens in the scope of smart cities. Part of these ideas and concepts are the standards and APIs to allow integration with several applications. The Next Generation Service Interface (NGSI) data format is used to transform data from running platforms and transform it into readable data to others applications.

Data Models play a crucial role once they define the harmonized representation formats and semantics used by applications both to consume and to publish data. [78] The data models present in NGSI to represent civic issues were designed to enable trivial interoperability with Open311 and include more information in each object. The FIWARE NGSI civic issue tracking data model defines two entity types: Open311:ServiceType and Open311:ServiceRequest. The Open311:ServiceType is a type of service a citizen can request. This request encompasses data from the Open311 GET Service List and Get Service Definition. The Open311:ServiceRequest specifies a service request for a service type made by a citizen [79]. The way the data models are defined is described in the requests documentation[80][81].

Since this standard is really the unique solution of a standardized data model to represent occurrences, this is the standard that will be used in this thesis project to represent such occurrences as well as to be able to integrate the created system with other existing applications.

## 2.3 Occurrence Management

In order to respond to the growth of the number of occurrences and the number of people that report them, it was necessary to implement tools that could facilitate the process such for city employees as for the citizens. For that, cities implemented solutions as the ones that are going to be mentioned to facilitate the process for both parties.

### 2.3.1 Current Solutions

Some applications are already in use to complement old methods like going physically to city hall, provided by city counties. For example, the city of Lisbon already has a platform that allows to manage the state of occurrences and provides information about it, which also provides for "citizens to enjoy more and better information about the city" [8] from trustworthy sources. Like Lisbon, other cities in Portugal and abroad have software that allows citizens to communicate some occurrences and check the status of them. Examples of this are the applications that are going to be talked about next: FixMyStreet and SIGOc.

#### 2.3.1.1 NaMinhaRuaLX

The first solution presented is the one implemented in Lisbon. This solution is available in the APP STORE and GOOGLE PLAY. It allows citizens to inform the city officials information about incidents that they spot at any time of the day.

The application in question resides in the field of smart governance of the Lisbon initiative, with the objective of this app being to "centralize all requests in one single portal"[11].

NaMinhaRuaLX is a product that is being implemented in Lisbon to help the process of reporting occurrences from concerned citizens. This application allows citizens to “report incidents, which need to be solved, related to green spaces, public illumination, mobility, underground works, and others” [9]. All of these problems can be reported in the application, and then it will arrive at city officials that will define priority to the requests.

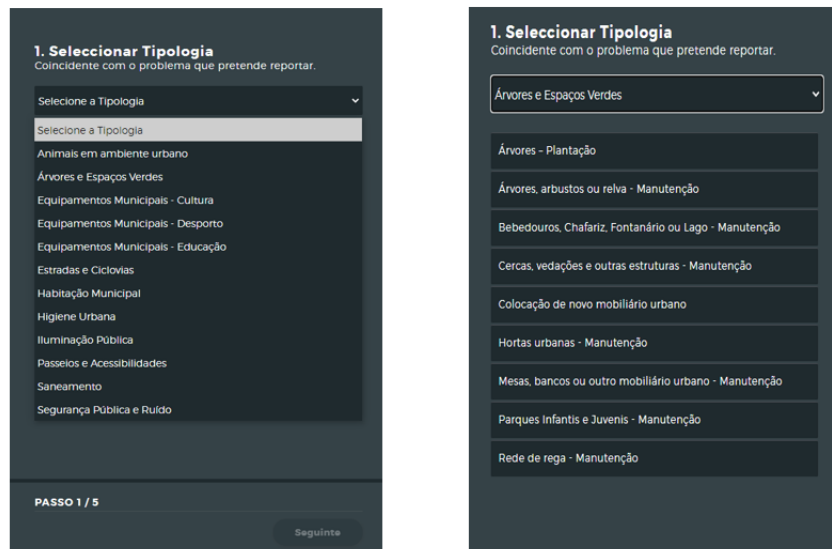


Figure 2.9: Submission report example NaMinhaRuaLX

Even though the types of occurrence are static, and the user cannot go beyond that, the selection of circumstances is extensive, covering almost all the situations. After categorizing, the person who registers the occurrence must indicate where it is located by choosing the desired location on a map or giving the name of the street, describing it by text, and adding photographs. At the end of this process, the reported event is categorized as “Registered to resolution,” where it will be analyzed. The possibility of searching for occurrences allows to search for area and type of incidents, as shown below.

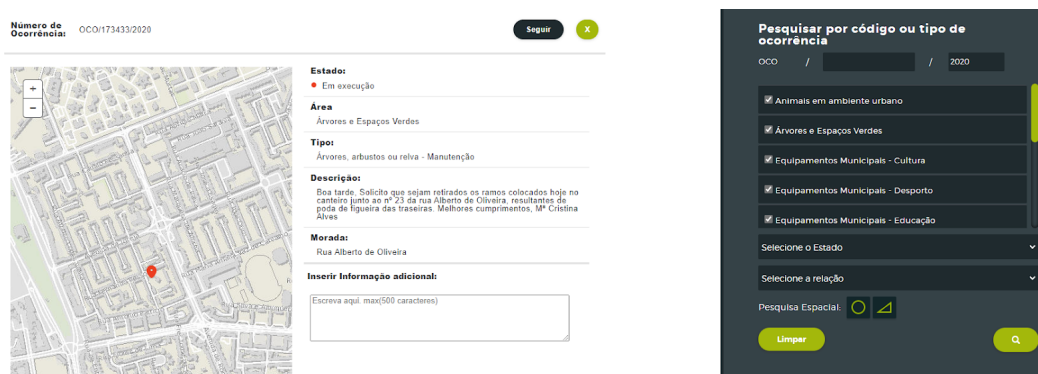


Figure 2.10: Occurrence Search NaMinhaRuaLX

This application makes use of API management from Azure services that allows external communications with the application [23]. Since the implementation of this application, citizens’ number of incidents has increased, as reported in Lisbon county’s official report. [12]

### 2.3.1.2 FixMyStreet

Another application with some relevance is FixMyStreet. FixMyStreet is a platform where cities can quickly launch a website that helps their citizens to report street problems.[24]

This Open-source platform was implemented in all of the United Kingdom by mySociety, a registered charity in England and Wales. The application's goal is to report the issues that occur in the streets, in this case, in the country. The application allows users to follow a reported occurrence from other users in an attempt to try to minimize the number of repeated reports. Between 2007 and 2017, more than 1.1 million reports were submitted in the FixMyStreet platform to local authorities [27]. About half of these reports are minor incidents like potholes and some environmental health issues like rubbish on the streets and dog fouling[27].



Figure 2.11: Reports on the FixMyStreet application, Manchester

The front-end application has a lot in common with the NaMinhaRuaLX application as the process of reporting an occurrence is almost the same. The rest of the application differs from the previous solution because FixMyStreet is an open-source platform that any person with the necessary knowledge of the technologies can use and create their own solution to fix their street. This implementation is explained on their website [25] and has a guide [26] to assist in more detail with this process.

With this application's main objective being to encourage citizens to report issues in their streets and neighborhood, the application must be directly connected to the platforms already in use by cities to have a smooth integration with the citizens. In the early days of FixMyStreet (FMS), this platform used an email-based approach as the standard way to redirect the incidents reported [28]. Still, with the passing years, they started to encourage city councils to implement an open311 endpoint by explaining its benefits to their services.

### 2.3.1.3 SIGOc

Another platform made for a smaller sample of users, a university campus, was SIGOc. This application allows for the Federal University of Rio Grande do Norte (UFRN) staff to register occurrences on the campus. The UFRN has 123 hectares, nearly 50,000 students, and around 5,000 professors and technical staff [56].

The SIGOc is an application that aims to improve public safety on the UFRN campus. The SIGOc is composed of three main parts: Campus Seguro, Vigilante UFRN, and Campus

Seguro Dashboard. The first part, Campus Seguro, is an application made for registering occurrences by staff and students. The second part, Vigilante UFRN, is another mobile application, but this one is aimed at the patrolling guards from campus. The last one is a web interface that manages the occurrences and guards and creates report documents.

These three applications do not communicate between them but through an API that standardizes requests. The architecture of the systems looks something like the image below.

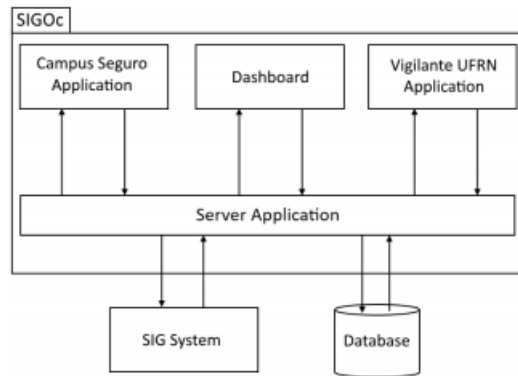


Figure 2.12: SIGOc Simplified Architecture

As seen in the image, the API receives and sends requests from applications to communicate between them. The application Campus Seguro resolves a communication problem by allowing that the reports are sent in a much faster way to the authorities and receive more pertinent information from the reports. This type of information would be lost or poorly understood by the supervisors when provided by a phone call.

The Dashboard application can be used to create more occurrences and assign teams to handle them. More actions that can be made using this application are tracking the position of security guards, viewing reports with different colors depending on their type, and heatmaps representing the density of occurrences in certain areas.

The last application, Vigilante UFRN, was made to support the patrol guards and their operations, making handling occurrences easier and faster. These applications made the use of radio communication not required except for cases of extreme necessity. Once the occurrence is registered, a notification appears in the guard's Vigilante application and then is provided with the necessary information regarding the event. These changes made a difference in how the occurrences were resolved, but new requirements were added with the passing time.

In the new version of the SIGOc, the system should show the gathered information from two IoT devices in the dashboard to improve supervisors' ability to understand emergencies and conditions such as fire events. One of those devices can collect means of cameras and sensors, the number of students in a classroom, its temperature, humidity, and motion. The other device was designed to measure the classroom's electricity consumption. To do it, the SIGOc uses services from the FIWARE European platform: Orion Context Broker and Cygnus.

### 2.3.2 Conclusion

These existing systems are examples of what is planned to do, although not meeting all of the planned features or integration with tools that could automate some static tasks. The one that comes near the proposed solution in terms of functionalities is the SIGOc application. Although it comes near, the project's scope is somewhat different since the application is only used on a college campus, restraining its size and requirements. The other applications, NaMinhaRuaLX and the FixMyStreet, are good examples of what a possible solution for a citizen application would be expected to do. The existence of many applications like these to report occurrences makes the need to create another application for citizens to report occurrences less relevant than creating an application for workers to interact with the occurrences or manage the upcoming requests since the protocols used to describe the requests can be implemented into the application to handle them proposed in this internship.

The SIGOc and the FixMyStreet projects provided more information about their operation since one was the target of a paper and the other was an Open-Source application so it was possible to gather more information. The NaMinhaRuaLX application did not have so much information available since it is an application used by the city of Lisbon and information such as standards used can be considered sensitive information. All of these applications are custom made for the solution that is going to be presented and so the way that the occurrences are managed can vary such as the way that the occurrences are described or the way that they are handled. Ubiwhere intends to build an application that can be applied in multiple counties customizing the applications to be created.

## 2.4 Workflow Engines

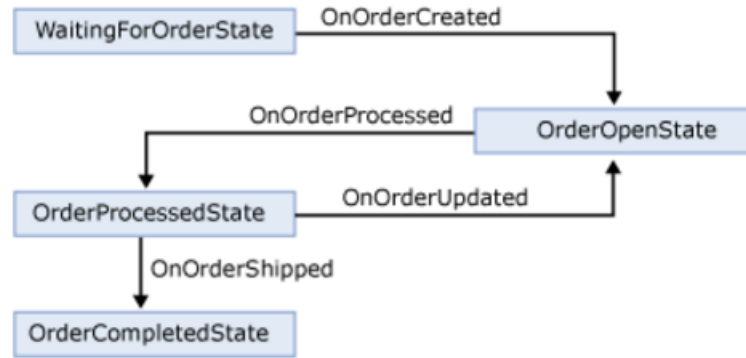
A workflow is a predefined set of steps to resolve a problem or complete a task, and unlike a process, the final goal is the delivery of a product or service. Workflows are an important concept for this project since they can be applied to determine a set of steps to apply in resolving an occurrence. An example of this would be the process of onboarding in a company where the company has several predefined steps to complete the process.

There are three categories: Sequential Workflow, Rules-driven workflows, and state machine workflows [38]. Each of these categories has some characteristics that make them different, including how the workflows are designed or how the task passes to another.

In Sequential workflow, the possibility is always to move forward and, like a flowchart, moves step-by-step having the forward step to depend on the previous one. This kind of workflow is useful for situations like production lines and other situations where the same sequence is always required. The Rules-driven workflow is based on the Sequential one with the addition of rules that make the final result vary according to the user's choices along the way. Unlike the previous methods, the state machine workflow thinks about the current state it is in and does not have any problem with going back if necessary, with the restriction of at any given time the only fulfills one state. An example of a state machine workflow can work is presented in the next image.

In the Figure above, the initial state, `WaitingForOrderState`, is changed when something triggers it. When triggered, a change of state occurs to the state `OrderOpenState`, which changes when triggered by another event. The state `OrderProcessedState` has possible events, the `OnOrderUpdated` and the `OnOrderShipper`. If the `OnOrderUpdated` occurs, a cycle will exist in the system that is interrupted by an `OnOrderShipped` event that





Source: Microsoft Developer Network

Figure 2.13: State Machine Workflow Example

concludes the process.

Workflow engines are tools designed to manage and monitor the state of activities in a certain workflow. The concept of a workflow engine consists of 6 logical components. [39]

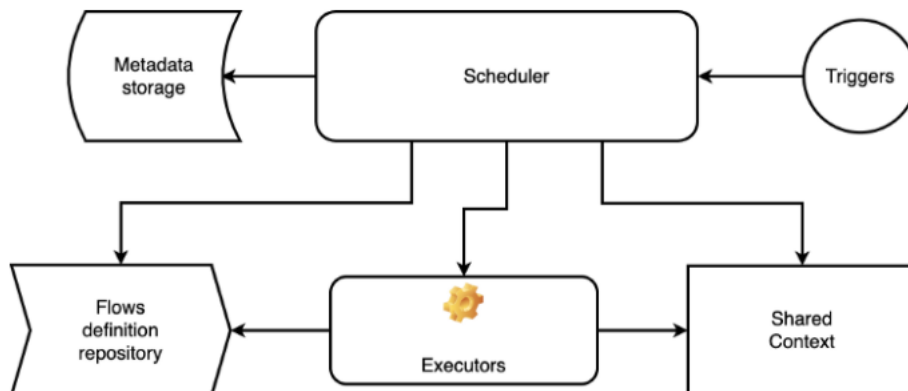


Figure 2.14: Components of a Workflow Engine

The central part of these engines is the scheduler, where the heavy work is done. This scheduler can be triggered by an event (API, timer, sensors, and others) and obtains the necessary data in the metadata storage where such information is stored. After this, the scheduler will order the executors to start the defined tasks. Whenever different tasks need context from each other, they will use the shared context to communicate. The Flows definition repository is the place where the definition of the workflow is stored. There are many ways to define workflows and each engine as its own.

Some of the engines use programming languages [44], others use languages that are used to describe business processes like Unified Modeling Language (UML), Business Process Model and Notation (BPMN), and Business Process Execution Language (BPEL)[45][46], and others have Extensible Markup Language (XML), Yet Another Markup Language (YAML) or JavaScript Object Notation (JSON) files [42] that describe the flow of the process.

### 2.4.1 Existing Market and Solutions

Ubiwhere imposed a restriction that dictates that the chosen technology must be open-source. With this restriction set, was taken into consideration a list of open-source workflow engines [40] to select the technology. Despite this restriction, some technologies used for the same purpose as the one proposed were explored [70] to see what features could be beneficial. Only one non-open source solution was explored since it was not possible to find relevant information about other products, due to the fact that they were too costly and did not have sufficient information available. The descriptions given for the products that will be approached were collected from surveys or official websites/ documentation of such products.

#### 2.4.1.1 Airflow

Apache Airflow[58], is a platform that allows programmatically author, schedule, and monitor workflows. The workflows in this platform are described through code, using the Python language. The Airflow team claims that this makes the workflows more maintainable, versionable, testable, and collaborative. The principles of this tool are Dynamic, Extensible, Elegant, and Scalable.[57]

This Open-source tool has some features available like a User Interface (UI) that where users can monitor, schedule, and manage the created workflows, integration with other third-party services, like Microsoft Azure and Google Cloud Platform, to execute the required tasks. [58] Although this feature is not yet stable, Airflow exposes a REST API that allows it to make several actions like return lists of directed acyclic graphs (DAGs) Runs for specific DAG IDs and others represented in their website. [59]

The ecosystem of the application is one of the significant advantages of the engine. This ecosystem comprises tools that can integrate into airflow [60], learning resources, and already implemented services that use airflow like Google Cloud Composer.

#### 2.4.1.2 jBPM

Another workflow engine, but this one using java, which is worth mentioning, is jBPM[46]. jBPM is a very flexible workflow management system. This workflow can be used as a standalone service or inside a custom service. It allows modeling business goals by describing the steps necessary to achieve a goal using a flow chart. The core of jBPM is written in java and enables users to execute processes using Business Process Model and Notation (BPMN) 2.0 specification.

jBPM can interact with other applications[61], which represents great value for this application. This integration can be made through an API that can connect directly to the engine. This API can load processes and execute them.

The web-based tools allow modeling, simulation, and deployment of the process and related artifacts such as data models, form, and rules. A web-based management console allows users to manage their runtime, manage task lists, perform Business Activity Monitoring, and check reports. If the IDE Eclipse is used, an extension can also allow the creation of business processes using a drag-and-drop mechanism and test and debug cycles.[62]



### 2.4.1.3 Kogito

Kogito[86] is a cloud-native application that relies on jBPMN. This application, like jBPM, belongs to the KIE group, group behind business automation projects. Kogito is designed to run and scale in cloud infrastructure, and it can be used with other cloud-based technologies such as Quarkus[83], Knative[84], and Kafka[85].

Like jBPM, Kogito can also communicate using APIs using domain-specific JSON data. The workflows can be edited using, for example, the Kogito Bundle VSCode extension. This extension allows users to edit the BPMN 2.0 processes and Decision Model and Notation (DMN) decision models in the IDE.

To deploy this service into the cloud, exists the Kogito Operator. This service is based on the Operator SDK and automates many deployment steps for the user. When given a link to a git repository with the desired application, this Operator can build the project and deploy the services that result from them. In line with this, Kogito also offers a command-line interface (CLI) to simplify some deployment tasks.

Kogito has two primary frameworks: Quarkus and Spring Boot. Quarkus is a Kubernetes-native Java framework. This framework optimizes Java specifically for Kubernetes by reducing the size of both the Java application and container image footprint, eliminating some of the Java programming workloads from previous generations, and reducing the amount of memory required to run those images. Spring boot is a java-based framework to build standalone Web Applications. This framework enables developers to develop applications with minimal configurations and without an entire Spring configuration setup.[63]

### 2.4.1.4 Camunda

Camunda[87] is a Java-based framework workflow engine that can be integrated with the spring framework. This framework uses BPMN 2.0 to automate processes and workflows.[64] The multiple Symbols and Events are configured using Java code or Javascript and XML notation to achieve this. Camunda also implements other standards like Case Management Model and Notation (CMMN) and Decision Model and Notation (DMN)[65]. These standards complement the use of BPMN 2.0. The BPMN is suitable to model well-structured and highly predictable work. Simultaneously, CMMN is better suited when talking of less predictable outcomes that require the active involvement of knowledge workers making decisions and planning during run-time[66]. At the same time, DMN helps to decouple business rules from both languages[67]. The next image, Figure 2.15, shows the essential components of the workflow engine.

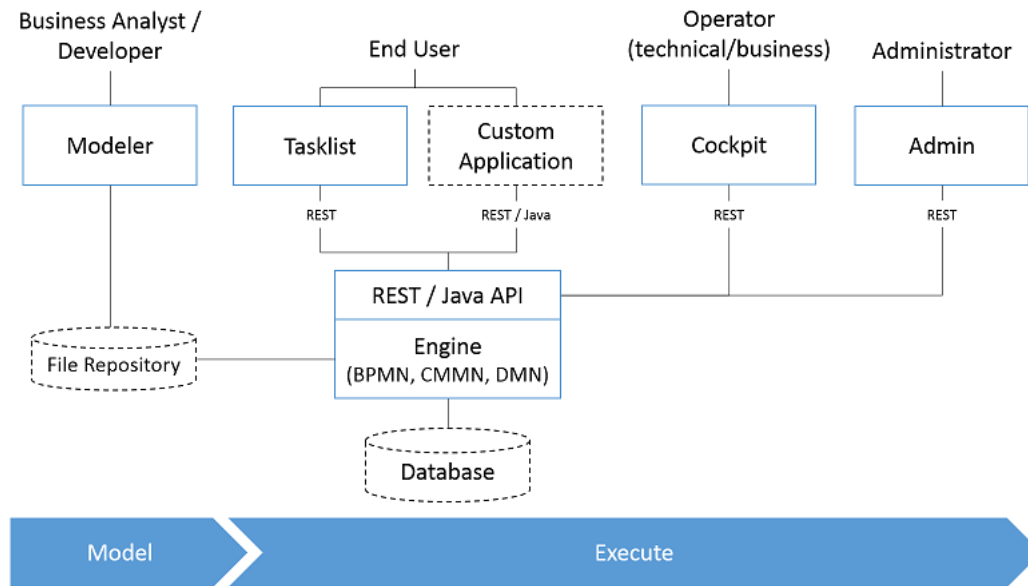


Figure 2.15: Camunda Workflow engine and Integration Representation

As seen, Camunda can communicate to custom applications through the REST API, which can contribute to integrate different applications to communicate with the End User. Other components described in the figure are systems to the user such as the Modeler where it is possible to describe the BPMN models. The Tasklist, the Cockpit and the Admin are components of the workflow engine that provide important features that can be accessible through the REST API or accessing the workflow engine Admin application through the web browser.

#### 2.4.1.5 Activiti

Activiti [88] is an open-source light-weight workflow and Business Process Management (BPM) that runs on Java. This workflow can run in any Java application on a server, on a cluster, or in the cloud. To model the workflows, it uses BPMN 2.0, and it supports external tools like Drools rule engine. When talking about programming languages, Activiti, as does Camunda, supports Javascript and Java. In terms of communicating with other systems, a REST API can be used. Despite this, the lack of monitoring capabilities in the community edition can be considered a down point for the work in question.

This workflow has a set of services that deal with the multiple processes available. These Services are RepositoryService, RuntimeService, TaskService, FormService, IdentityService, HistoryService, ManagementService, DynamicBpmnService. The RepositoryService helps manipulate the deployment of the process definitions and their static data. The RuntimeService manages the processes that are running at the time and the variables associated with them. The TaskService keeps track of the UserTask, that means, the tasks that need to be carried out manually. FormServices, a non-obligatory service, is used to define the start and task form in a process. IdentityService is the service that manages the Users and Groups created. HistoryService keeps track of the history of the Activiti Engine. ManagementService is many times not required when making an application, but it relates to the metadata. DynamicBpmnService provides a way to change anything in a process without having to redeploy it.[68]

#### 2.4.1.6 k2 Platform

K2 Platform [70] is a product from Nintex used to automate workflows. The K2 platform can be divided into three components: Data provider, smartObject, Data Consumer.[69] The first component, SmartObject, is a data integration/data management framework that serves as a bridge between those who provide data to those who consume the data by homogenizing and manipulating the data. K2 smartForms allows building user interfaces by dragging and dropping components into User interface (UI) blocks. And the third one, Data consumer, is the workflows, which allows to visualize business processes and allocate roles, participants, and required activities. The workflows can be designed graphically, and users can connect with custom applications using a javascript-based custom broker.[70] K2 also provides reporting and analytics tools to track tasks and variables. This platform can not be embedded in an application but can be run on-premise or on the cloud. Since this is a non-open-source solution, its pricing starts under 2000per month for 100 users.

#### 2.4.2 Bonita BPM

This BPM is composed of two parts, the Bonita BPM Studio and the platform. The Studio is a graphical environment for creating processes and Web application forms. The platform is formed by the BPM engine and Bonita Portal, which provides visualization of tasks and performs actions. The Bonita BPM [41] supports BPMN 2.0 as a modeling language and allows users to import processes defined in BPMN 2.0, XPDL 1.0, and jBPM 3.2. Bonita has Java as its primary programming language. The Bonita software has more than 80 connectors to integrate with other applications and services. A simple log and API are available in order to access the Engine and infrastructure data. After the validation of diagrams, the Bonita BPM can display suggestions to fix errors.

##### 2.4.2.1 Zeebe

Unlike other Workflow Engines, Zeebe is a Workflow Engine for microservices orchestration that provides high throughput, low latency, and horizontal scalability[71]. Zeebe uses a client/server model where the server (broker) is a remote engine that runs as its program on a Java Virtual Machine (JVM). The way that Zeebe transmits information to its clients works like Apache Kafka, via a publish-subscribe system to expose workflows event streams to clients. When creating topics, there is the possibility of configuring the replication factor. The replication factor determines how many “hot standby” copies of a partition are stored on other brokers. With this, if one goes down, there is always another possibility of replacing it, providing fault tolerance and high availability, without an external database. But like other Workflow engines, the workflows are designed using BPMN 2.0. Zeebe is language-agnostic because it can have clients in every language that supports gRPC, although it has out-of-the-box clients for Java and Go [71]. It is also available a way to export information using Zeebes exporter. Exporters provide an entry point to process every record that is written on a stream. With them, it is possible to persist data in an external database or to a visualization tool[72]. However, exporters will only be loaded when configured in the main Zeebe YAML configuration file. However, Zeebe covers fewer BPMN symbols than the more mature workflow engines such as Camunda BPM [71].

The summary of the main points considered for this project are described in the table below for each of the studied workflows.

	Airflow	jBPM	Kogito	Camunda	Activiti	K2 platform	Zeebe
Design of workflows	(Python)	BPMN2.0	BPMN2.0	BPMN2.0	BPMN2.0	Design graphically	BPMN 2.0
Open Source	Yes	Yes	Yes	Yes	Yes	No	Yes
Deployment	Python application	Standalone or custom application	Cloud	Hosted on cloud Or as a Java Spring application	Local Server, Cluster or Cloud	Server or Cloud	Docker
Communication with other applications	API (BETA)	API	API	API	API	Javascript broker	Exporters API

Table 2.1: Workflow Engines comparison table

## 2.5 Final thoughts

In this chapter, research was made about the themes relevant for this work. We started to explore solutions and initiatives about smart cities checking what solutions they implement and what problems they aim to resolve. These applications of occurrence management are included in a larger section of smart cities solutions in the means that they gather information and provide them to the services providers in an attempt to make them more efficient. Standards that could be relevant to the problem of standardizing the requests and the data that comes associated with them were studied such as examples where such standards are implemented to have a perspective of how this could be done in the work that is proposed.

Some methods of occurrence management already used as a solution in some smart cities initiatives were studied and their main characteristics analyzed since the scope of the project requires some way for the citizens to report their occurrences. Even though building a citizen-centered application to report occurrences is not the main goal of the project, it is a relevant part of the system, so that the platform to be developed has input from another point and with this gathers more information.

The concept of using workflow engines is already present in smart city initiatives for purposes such as collecting data. They can be used to create and manage flows. These flows can be triggered from established actions such as receiving an email. The objective is to have a solution that can be used as an independent module for communication with other applications and that is capable of making a connection to multiple data sources to collect as much data as possible. This solution will be connected to a workflow engine to resolve occurrences reports using an established flow and making it more efficient.

It is intended for the final solution for the back-end application to be as integrable as possible so that it can receive requests from third-party applications such as Ubiwhere's Urban Platform. Since the users of the system will not have, most of the time, any experience with programming and other skills associated, it is proposed to have an application to manage the system using a UI (User Interface) minimizing the need for trained personnel to manage and run the application.

The solution chosen as the Workflow Engine to be used was the Camunda. The reason to choose this engine rather than the other was that it fulfills all of the requirements, the size of its community that remains very active and the support given through the engine documentation and video tutorials that teach on how to use their tools at the beginner level. Another valuable characteristic was the inclusion of other standards to create flows to complement the use of BPMN2.0. The BPMN2.0 models that will be created must use the specific tool designed for Camunda to integrate them into the engine, but this is not a setback because many of the observed tools require the same.

This page is intentionally left blank.

## Chapter 3

# Planning and Methodology

In this chapter, the methodology implemented throughout the internship is explained. Here, the planning and deviations are shown as well as the tools used to manage and orchestrate the work and communication between teams.

### 3.1 Process Management

During the duration of this internship, an Agile methodology was adopted, more specifically the scrum framework. The agile methodology promotes continuous iteration of development and testing during the development lifecycle of the project [74].

In this methodology, the development and testing activities are concurrent [74]. Unlike a more traditional approach, Waterfall, where the development of the software flows from a starting point and sequentially goes to a final point, in the Agile methodology exists an incremental and iterative approach to the software design which allow teams to prioritize the value of the product as well as having more feedback about the developed product.

Scrum, a framework from Agile, concentrates specifically on how to manage tasks within a team-based development environment [74]. In the environment of this methodology, it is normal to encounter small teams of 7 to 9 elements. This framework focuses on the deliveries from the sprints which are periods that go from one week to two months. Before each sprint, the team gathers to review the work done until that moment in time and prioritize the subsequent requirements to implement. The Product Backlog, i.e, the requirements, or in this case, the user stories, can be the target of change to add more value to the product, once this is the team's final goal.

At the beginning of the sprints, assignments are planned, as well as the results expected for it. After the sprints, the results are discussed with every stakeholder involved to have feedback about the work done and verify what went well and what needs to have more effort put into it in a review meeting. This is possible for Scrum once in this methodology the client has more involvement that provides valuable feedback to the development allowing for him to give value to the product and its features.

In the specific case of this internship, were considered sprints of one week. To signalize the beginning and ending of each sprint. Weekly the intern would gather with his advisor where the advisor would behave as a client of the solution, representing Ubiwhere. In these weekly meetings, the advisor would see the progress made in the previous week and define the next steps for the forthcoming week as well as objectives that need to be met by the intern.

Later, in the requirement gathering phase, the intern worked together with the client in order to define the requirements and validate them according to the client's view. When problems regarding the requirements or in the development of the project would occur, those were also discussed with the intern adviser that would give recommendations about what to do and sometimes what kind of solutions would be possible to make. Sometimes when the intern encountered a problem, he would find various solutions and would also discuss with the advisor about what was the best solution for the use case in order to maintain the client's view of the project.

## 3.2 Tools

The use of project management tools is important when developing software in an organized way. These tools are often used to provide communication and organization within teams that allow for a better flow of work. During this internship, the following tools were adopted to allow such behavior and attributes.

- **Clockify** - Is a free time-tracking software that is used by both interns and employees of Ubiwhere, that allows to track the time spent on each task.
- **GitLab** - Is an open-source Git-repository manager. It is hosted in the company headquarters and provides functionalities such as a version management system to all artifacts generated, this can be source code, configuration files, and ordinary documents and reports that are necessary for the normal operation of the company. This software is very important for development once it allows to share code, keep the multiple version of it, and merge new functionalities into the source code.
- **Slack** - is a messaging system mainly for enterprise teams. This software allows better-organized communication by having chat rooms and direct messaging. In the hosting company, it is where most of the communication is done.
- **Docker** - is an Open-Source tool that offers lightweight virtualization of the application and all its dependencies. The main objective of this tool is to increase the compatibility with the environments where the application will run, reducing the link of the application and where it was made. Moreover, this tool also decreases the application deployment time.

## 3.3 First Semester

The first semester was mainly for research and planning for the work to be done in the second semester. The tasks involved in the semester were the following:

### 1. Concept Study

This task involved research in the themes addressed in the State of the Art Smart cities, Open standards, Occurrence management, and Workflow Engines. Each of the themes was made a study of the market to try to observe the different approaches to similar problems. The standards available were studied to understand how communication between applications and how the information is stored. In the case of Workflow Engines to have a general idea of what was possible to do with them and

the associated features available. The solutions studied were low-level, working in the code perspective and others were very high level allowing for a complete abstraction with the code involved to the features.

## 2. Requirements Specification

For this task, meetings were held with the intern and Ubiwhere's advisor. The intern gathered the requirements of the platform from those meetings and, in an afterward meeting, were reviewed and discussed.

## 3. Architecture and Technologies

In this task, the requirements of the platform were taken into account in the process of drawing an initial architecture.

## 4. Intermediate Internship Report

This task was made alongside all the other tasks with the help and feedback of both the advisors.

# 3.4 Second Semester Planning

The main objective of the Second semester was the development of the proposed system to satisfy the defined needs and requirements. The tasks involved in the second semester where the following:

## 1. Training and testing of solutions

The intern studied several types of technologies in order to be prepared to initiate the project. The first thing to do was assess whether the workflow engine would meet every requirement or most of them and if so proceed with its implementation and integration in docker. After this the intern gained some experience in the other technologies that would be required to the project like the Django REST framework and React.

## 2. Product Development

This task was the task that occupied most of the time of the intern, and, as predicted, was divided into weekly sprints with defined objectives stated in the beginning of each one. In the end of the sprint the work would be revised by the adviser that would assess the nexts objectives.

## 3. Product Testing

At this internship the scrum framework was used so the product testing was done alongside the product development, specifically the unit tests.

## 4. Final Internship Report

This task was made alongside with the previous ones, however with more effort into it in the late phases of the project.



### 3.5 Gantt Diagram

The Gantt diagram summarizes the planned work for this internship. This diagram shows the planned tasks for the entire internship and its duration for the first and second semesters. The tasks represented in the diagram are very high level once they were programmed in the beginning of the project.

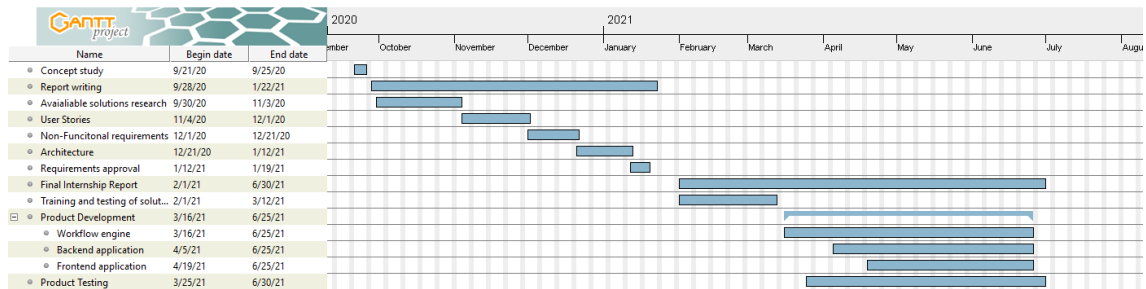


Figure 3.1: Representation of the planned tasks through a Gantt Diagram

### 3.6 Process Plan and Deviations

Even though the inicial Gantt diagram showed the high level tasks, that initial plan suffered some changes and deviations due to circumstances that occurred during the product development. All of the deviations and some delays that occurred forced the internship delay, in a major part the addition of new software to the system, that required new configuration and more effort in the intern. Despite the delay of the internship the intern did not have enough time to materialize the citizen application due to the integration of the different systems implemented and the addition of new systems that were not initially thought of having.

A second Gantt diagram that shows the real time duration of the tasks with more granularity can be seen in the figure 3.2. When compared to the initial Gantt diagram it is possible to verify that the tasks took longer than expected as well as their integration, where the lines of the graphics cross, also require a lot of time increasing the development time.

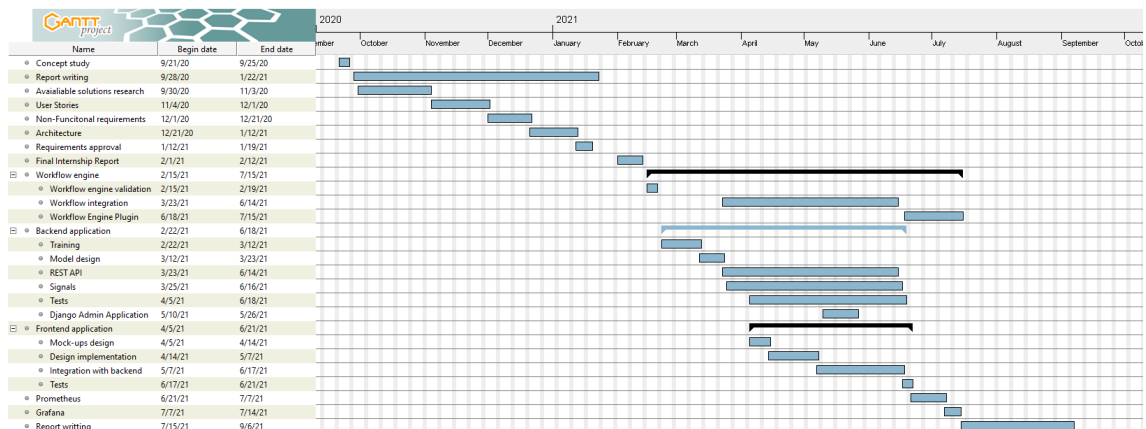


Figure 3.2: Tasks duration represented through a Gantt Diagram

## 3.7 Risk Management

In this section the factors that could lead to the success of the internship are enumerated and analyzed.

The activity of Risk Management is important for developers and managers. This helps the stakeholders to plan methods to overcome risks that can influence the outcome of the project. The management of risks helps to control and mitigate them, but for that, it is important to identify potential problems in the development of the product and establish plans to mitigate them in case they happen. The management control is not a static process that is done only once, it needs to be a continuous process where risks are identified, analyzed, and evaluated the probability of it happens and, in cases where the probability is high and the impact of the risk is also high, elaborate a mitigation plan to reduce the consequences of such risks. One of the revisions was made after the first semester were the risk number 5 was identified and added to the matrix.

To plan the risks, it is important to establish the criteria for the success of the internship and how the risks are gonna be measured to identify what and how these could damage the internship's success.

### 3.7.1 Internship Success Criteria

The success of the internship is dependent on the defined criteria at the beginning of the internship. The established criteria that were defined and were not subject to change were the following:

- The requirements and the Architecture must be reviewed and approved by Ubiwhere.
- The requirements with the priority “must-have” must be successfully implemented.
- The delivered product must ensure the quality requirements that were specified.
- The internship must end within the planned time with all the previews goals achieved.

### 3.7.2 Impact Scale

The impact of a risk is measured by the damage that it can have on the project if not appropriately dealt with. For this, a scale was used to best describe such impacts:

- **Catastrophic** - The success of the project could not be achieved.
- **Critical** - With significant cost or effort, the success criteria could be achieved.
- **Significant** - With some effort or cost, the success criteria will be achieved.
- **Marginal** - With minimal impact on the success criteria, the success criteria would be achieved with no additional cost.

### 3.7.3 Probability Scale

The probability Scale indicates how likely it is for a risk to occur. For that the following scale was used:

- **Very Likely** - it is almost certain that the risk will occur.
- **Likely** - there is a high probability of the risk occurring.
- **Not Very Likely** - it is not probable that the risk will occur, but it is possible.
- **Unlikely** - it is assumed that the risk is nearly impossible to happen.

### 3.7.4 Risk Management Analysis

In the following tables are expressed the risk identification and analysis. The risks are categorized with the scales that were previously described.

ID	R1
Condition	The lack of experience of the intern in the technologies and tools that are going to be used.
Consequence	The development of the final product can take more time than originally expected.
Impact	Significant
Probability	Very likely

Table 3.1: Risk Number 1: Lack of Experience

ID	R2
Condition	The situation of the pandemic that is currently lived affects the intern's productivity.
Consequence	The development of the final product can take more time than originally expected.
Impact	Critical
Probability	Not Very Likely

Table 3.2: Risk Number 2: Pandemic Situation

ID	R3
Condition	The scope of the project is not very clear in the proposal made by the hosting company.
Consequence	The requirements and Architecture of the final product suffer changes that could increase the complexity of the internship.
Impact	Critical
Probability	Not Very Likely

Table 3.3: Risk Number 3: Scope of the project not clear

ID	R4
Condition	The predicted tasks take more time than expected due to the project complexity.
Consequence	Cause a delay in the subsequent tasks that could affect the final delivery
Impact	Critical
Probability	Likely

Table 3.4: Risk Number 4: Predicted tasks take longer than expected

ID	R5
Condition	The project scope be too wide for the time to developed it
Consequence	The threshold of success is not completed.
Impact	Catastrophic
Probability	Not very Likely

Table 3.5: Risk Number 5: Scope of the project too wide

### 3.7.4.1 Risk Exposure Matrix

The identified risks were labeled according to their impact on the project in case they happen and the probability of them to happen. In the next figure, figure x, is shown in a matrix where the two criteria are represented in the axes. To better understand the ratio of the identified risks, they were identified in the matrix to give it more perspective and to better understand their importance.

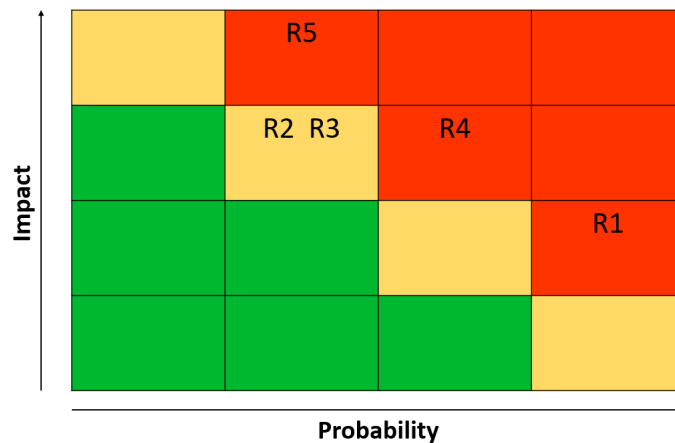


Figure 3.3: Risk Exposure Matrix

### 3.7.4.2 Mitigation Plans

Once the risks are detected, it is important to establish plans to mitigate those in the “danger zone” (represented with the red color), since this means that those risks represent a danger to the internship.

- **R1**

To mitigate this risk the intern will allocate sufficient time to get familiar with the

necessary technologies before starting the project.

- **R4**

The intern must communicate these delays to the advisor for them to establish a plan to minimize the impact of the delay and reorganize the requirements in case there is a more valuable feature that the intern should focus on.

- **R5**

To mitigate this risk is necessary to reduce the scope of the project and probably redefine some requirements and establish priorities between them.

This page is intentionally left blank.

# Chapter 4

## Requirements

In this chapter, the requirements that were gathered for the development of the product are presented. First, the functional requirements will be exposed and then the non-functional requirements that the application must meet.

The desired system will consist of a set of three applications. One application for reporting occurrences and providing feedback for the citizens, another to manage the occurrences, the users and the teams that are going to be in charge of responding to the requests and fix them and the last application that left mentioning would be an application to inform the workers what occurrences to tackle and resolve.

Therefore three different types of users were identified: citizens, admin, and workers. Each one of them will be presented with a set of requirements for the application that they are going to use. This internship's intended work is the management of occurrences with the assistance of a workflow engine to automate the flows and certain tasks. This means that the main part of the system is the admin and workers view, and the functionalities associated with the management and resolve occurrences.

The application required by the workers will have a great deal of importance because it is vital for the realization of the activities that only them can resolve. This application will ensure that workers know what tasks are assigned to them most effectively and allow them to be able to respond quicker to the occurrences. This application could also reduce the costs of operation since it can reduce the number of travels of the workers or the number of phone calls needed to inform them what task to do next.

Although it is important to show the platform's value directly to the people and collect feedback more effectively from the normal citizen, the citizens' application is not considered a must-have in this internship. The motive that this happens is because it is possible to obtain information from other sources and therefore do not restrict the functionalities of the system. In this case a dataset from the city of Bloomington, USA was used to provide the application with occurrences.

### 4.1 Terminology

User stories were the chosen methodology in order to describe the requirements of the system that will be developed. Users Stories are a description of the functionalities expected to have in the system from the perspective of an end-user. These stories are made using small sentences that use non-technical language while this means the requirements will not



specify most of the technical details, it is a representation that is easily understandable by every stakeholder

The acceptance criteria of each User Story were written using Gherkin language, to standardize and make them more understandable. Acceptance criteria are the conditions that a product must meet to be accepted by a user, customer or other system[89]. This language is mostly used in software testing and its structure is gonna be exploited in the next section. [73]

This language helps stakeholders communicate and create a better understanding of what is expected from the system that is going to be built.

## 4.2 User Stories Structure

The requirements gathered for this project adopt the following structure:

US#ID

- **Description:** As a <role>, I want <feature> so that <reason>.
- **Acceptance Criteria:**
  - **Scenario:** Some situation
    - \* **Given** Some precondition  
**And** Some other precondition
    - \* **When** some action by the actor  
**And** some other action  
**And** yet another action
    - \* **Then** Some testable outcome is achieved  
**And** something else we can check happens too
- **Dependencies:** US#ID
- **Priority:** Must Have, Should have, Nice to Have

### 4.3 User Stories

Before the definitions of the User Stories, is presented a table that summarizes the User Stories providing their ID, definition and the level of priority, Table 4.1.

ID	Description	Priority
<b>Citizen</b>		
1	As an unregistered user, I want to report an occurrence so that city officials can resolve it.	Nice to Have
2	As an unregistered user, I want to authenticate in the application so that I can access features that unauthenticated users cannot.	Nice to Have
3	As a registered user, I want to see the submitted reports on a given radius from a given location so that I can see the occurrences registered by other users in that area.	Nice to Have
4	As a registered user, I want to filter the reported occurrences so that I can search more precisely for occurrences.	Nice to Have
5	As a registered user, I want to view the information associated with the occurrences and their current state so that I can be informed about those occurrences.	Nice to Have
6	As a registered user, I want to submit a report so that I can expose my problem.	Nice to Have
7	As a registered user I want to logout from the application so that my account is logged off in that session	Nice to Have
<b>Admin</b>		
8	As an admin, I want to add a new workflow so that a new occurrence type can be automated and described in the application.	Must Have
9	As an admin, I want to edit an existing workflow so that the efficiency of the system will improve	Must Have
12	As an admin, I want to create groups of workers so that they would be represented in the application.	Must Have
13	As an admin, I want to add new employees to the platform so that they will be represented.	Must Have
14	As an admin, I want to see the statistics about the individual tasks of the workflows so that it is possible to make conclusions from them and see how they can be improved.	Must Have
18	As an admin, I Want to eliminate existing teams so that it is possible to erase them from the platform.	Should Have
10	As an admin, I want to add observations to the submitted report so that it can be more easily understood.	Should Have
15	As an admin, I want to see statistics about the different teams so that it is possible to evaluate their work.	Should Have
16	As an admin, I want to see the Users and the tasks associated with them so that I can keep track of what they are doing.	Should Have
17	As an admin, I Want to edit the existing teams so that I can modify the teams.	Should Have

11	As an admin, I want to validate occurrences that were automatically generated from third party applications so that we ensure that all information entered in the system is accurate	Nice to Have
<b>Workers</b>		
19	As a worker, I want to see the occurrences that can be attributed to me or my team.	Must Have
20	As a worker, I want to select the occurrence that I am going to resolve and remove it from the list when it is resolved.	Must Have
21	As a worker, I want to change the current status of the occurrence to inform the citizens and other entities.	Must Have
22	As a worker, I want to ask for human resources to resolve the occurrence to solve it better or faster.	Must Have
23	As a worker, I want to receive notifications about the new occurrence that I was appointed to have that information.	Must Have

Table 4.1: User Stories Overview

### 4.3.1 As a Citizen

US#01

- **Description:** As an unregistered user, I want to report an occurrence so that city officials can resolve it.
- **Acceptance Criteria:**
  - **Scenario:** The user wants to report an occurrence.
    - \* **Given** that am I am an unregistered user
    - \* **When** I provide the location  
**And** the occurrence typology according to a premade list  
**And** description of the occurrence
    - \* **Then** My report is sent.
  - **Scenario:** The user does not provide the necessary information
    - \* **Given** that am I am an unregistered user
    - \* **When** I don't provide the necessary information
    - \* **Then** My report is not sent and error information appears on my screen
- **Dependencies:** None
- **Priority:** Nice to Have

US#02

- **Description:** As an unregistered user, I want to authenticate in the application so that I can access features that unauthenticated users cannot.
- **Acceptance Criteria:**

- **Scenario:** The user provides a valid set of credentials.
  - \* **Given** that I am unauthenticated
  - \* **When** I provide a valid set of username and password
  - \* **Then** I am redirected to the main page of the application.
- **Scenario:** The user provides an invalid set of credentials.
  - \* **Given** that I am unauthenticated
  - \* **When** I provide an invalid set of username and password
  - \* **Then** I should receive an error message.
- **Dependencies:** None
- **Priority:** Nice to Have

#### US#03

- **Description:** As a registered user, I want to see the submitted reports on a given radius from a given location so that I can see the occurrences registered by other users in that area.
- **Acceptance Criteria:**
  - **Scenario:** The User wants to observe the submitted reports and shares their location.
    - \* **Given** that the user is logged in to the application  
**And** is on the main page
    - \* **When** the user gives a location  
**And** checks what types of reports want to see
    - \* **Then** the reports appear on the map depending on the user location.
  - **Scenario:** The User wants to observe the submitted reports and does not share their location.
    - \* **Given** that the user is logged in to the application  
**And** is on the main page  
**And** does not give his location
    - \* **When** checks what types of reports want to see
    - \* **Then** the reports appear on the map by a default location.
- **Dependencies:** US#02
- **Priority:** Nice to Have

#### US#04

- **Description:** As a registered user, I want to filter the reported occurrences so that I can search more precisely for occurrences.
- **Acceptance Criteria:**
  - **Scenario:** The user wants to filter the occurrences by their name
    - \* **Given** that the user insert the name of the pretended occurrence
    - \* **When** the user is writing the name of the occurrence

- \* **Then** the results will be automatically triaged.
- **Scenario:** The user wants to filter the occurrences by their category
  - \* **Given** That the user checkboxes the types of occurrences that he wants to see
  - \* **When** he hits the search button
  - \* **Then** the results will be triaged.
- **Dependencies:** US#02
- **Priority:**Nice to Have

## US#05

- **Description:** As a registered user, I want to view the information associated with the occurrences and their current state so that I can be informed about those occurrences.
- **Acceptance Criteria:**
  - **Scenario:** The user wants to see the information about an occurrence
    - \* **Given** that the user is logged in
    - \* **When** it clicks on occurrences  
**And** searched for occurrences
    - \* **Then** a pop-up with the details of the occurrence appears.
- **Dependencies:** US#02,US#03
- **Priority:**Nice to Have

## US#06

- **Description:** As a registered user, I want to submit a report so that I can expose my problem.
- **Acceptance Criteria:**
  - **Scenario:** The user wants to report an occurrence providing his personal information.
    - \* **Given** that am I am a registered user
    - \* **When** I provide the location  
**And** the occurrence typology according to a premade list  
**And** description of the occurrence
    - \* **Then** My report is sent providing my personal information.
  - **Scenario:** The user wants to report an occurrence without his information
    - \* **Given** that am I am a registered user
    - \* **When** I provide the location  
**And** the occurrence typology according to a premade list  
**And** description of the occurrence  
**And** check a box to send my report anonymously
    - \* **Then** My report is sent without my personal information
- **Dependencies:** US#02

- **Priority:**Nice to Have

US#07

- **Description:** As a registered user I want to logout from the application so that my account is logged off in that session
- **Acceptance Criteria:**
  - **Scenario:** The user wants to log out from the application
    - \* **Given** that has a session open
    - \* **When** Clicks on the log out button
    - \* **Then** the presentation page of the application is showed to the user
- **Dependencies:** US#02
- **Priority:**Nice to Have

### 4.3.2 As an admin

Some of the requirements such as the login and logout are similar between the different applications and therefore only expressed once.

US#08

- **Description:**As an admin, I want to add a new workflow so that a new occurrence type can be automated and described in the application.
- **Acceptance Criteria:**
  - **Scenario:** The user wants to add a workflow to the system.
    - \* **Given** that the user has the diagram described
    - \* **When** describes the service type associated **And** insert in the application the diagram variables
    - \* **Then** a new workflow will be added to the system
- **Dependencies:** US#02
- **Priority:**Must Have

US#9

- **Description:**As an admin, I want to edit an existing workflow so that the efficiency of the system will improve.
- **Acceptance Criteria:**
  - **Scenario:** The admin wants to substitute the original diagram from another.
    - \* **Given** that a diagram is already associated with a service type
    - \* **When** The user replaces the preview diagram for the new one
    - \* **Then** a new way to resolve the occurrence is implemented

- 
- **Scenario:** The admin wants to modify variables to the already deployed diagram
    - \* **Given** that a diagram is already associated with a service type
    - \* **When** a user modifies the object he can add, delete or modify the variables onto the system.
    - \* **Then** other variables will be represented in the system.
  - **Dependencies:** US#02
  - **Priority:**Must Have

## US#10

- **Description:**As an admin, I want to add observations to the submitted report so that it can be more easily understood.
- **Acceptance Criteria:**
  - **Scenario:** The user wants to provide additional information to the request.
    - \* **Given** that a request already has been submitted  
**And** some information is invalid or incorrect
    - \* **When** the admin selects the edit option in the report
    - \* **Then** the report becomes editable allowing the admin to change its information.
- **Dependencies:** US#02
- **Priority:**should have

## US#11

- **Description:** As an admin, I want to validate occurrences that were automatically generated from third party applications so that we ensure that all information entered in the system is accurate
- **Acceptance Criteria:**
  - **Scenario:** The user wants to allow for some occurrences to be added to the system.
    - \* **Given** That the user is on the automatically generated page  
**And** wants to add some occurrence
    - \* **When** The user clicks on the add button present on the event
    - \* **Then** The occurrence is represented in the platform.
- **Dependencies:** US#02
- **Priority:**Nice to Have

## US#12

- **Description:** As an admin, I want to create groups of workers so that they would be represented in the application.

- **Acceptance Criteria:**

- **Scenario:** The user wants to create a groups of workers in the application
  - \* **Given** the employees are registered in the system
  - \* **When** the admin goes to the main page
    - And** clicks on the new group button
    - And** adds at least one worker to the group
  - \* **Then** a group of workers is added

- **Dependencies:** US#02

- **Priority:**Must Have

US#13

- **Description:** As an admin, I want to add new employees to the platform so that he will be represented.

- **Acceptance Criteria:**

- **Scenario:** The user wants to add a worker to the system
  - \* **Given** the employee is not registered in the system
  - \* **When** the admin goes to the add employee panel
    - And** provides the employee information
    - And** click on the add button
  - \* **Then** the new employee is represented in the system
    - And** an username and password is sent to the employee email.

- **Dependencies:** US#02

- **Priority:**Must Have

US#14

- **Description:** As an admin, I want to see the statistics about the individual tasks of the workflows so that it is possible to take conclusions from them and see how they can be improved.

- **Acceptance Criteria:**

- **Scenario:** The admin wants to see statistics about the different steps of a workflow.
  - \* **Given** that the admin selects the workflow witch plans to see information
  - \* **When** selects the details menu
    - And** clicks of a task
  - \* **Then** statistics about the task will be shown.

- **Dependencies:** US#02

- **Priority:**Must Have

US#15



- 
- **Description:** As an admin, I want to see statistics about the different teams so that it is possible to evaluate their work.
  - **Acceptance Criteria:**
    - **Scenario:** The admin wants to see statistics associated with current and past work.
      - \* **Given** That the user is on the page of management of the teams
      - \* **When** The user explores the team details
      - \* **Then** graphics about the team shows to the user
  - **Dependencies:** US#02
  - **Priority:**should Have

US#16

- **Description:** As an admin, I want to see the Users and the tasks associated with them so that I can keep track of what they are doing.
- **Acceptance Criteria:**
  - **Scenario:** The admin wants to see users and the tasks associated with them.
    - \* **Given** That the user is on the main page
    - \* **When** he clicks on the tasks section  
**And** clicks in the task
    - \* **Then** information about it including who handles it is shown.
- **Dependencies:** US#02
- **Priority:**should Have

US#17

- **Description:** As an admin, I want to edit the existing teams so that it can modify the teams.
- **Acceptance Criteria:**
  - **Scenario:** The admin wants to change the existing teams
    - \* **Given** The user is on the teams' page  
**And** Goes to the detail page
    - \* **When** It goes to the details section on the team  
**And** goes to the edit team
    - \* **Then** The admin can choose to delete or add team members.
- **Dependencies:** US#02
- **Priority:**should Have

US#18

- **Description:** As an admin, I want to eliminate existing teams so that it is possible to erase them from the platform.
- **Acceptance Criteria:**
  - **Scenario:** The admin wants to delete an existing team
    - \* **Given** The user is on the teams' page
    - And** Goes to the detail page
    - \* **When** It goes to the details section on the team
    - And** choose to delete the team
    - \* **Then** the admin is asked if he is sure of his choice. If yes the team is deleted from the system.
- **Dependencies:** US#02
- **Priority:**should Have

### 4.3.3 As a worker

US#19

- **Description:** As a worker, I want to see the occurrences that can be attributed to me or my team.
- **Acceptance Criteria:**
  - **Scenario:** The worker wants to see the occurrences left to be assigned
    - \* **Given** The worker is logged in
    - \* **When** the user selects the option to see what assignments are available
    - \* **Then** A map with the assignments pinpointed is shown.
- **Dependencies:** US#02
- **Priority:**Must Have

US#20

- **Description:** As a worker, I want to select the occurrence that I am going to resolve and remove it from the list when it is resolved.
- **Acceptance Criteria:**
  - **Scenario:** The worker wants to select a task to claim
    - \* **Given** The user is on the page to select the task
    - \* **When** selects a task
    - \* **Then** It can select it as its next job.
- **Dependencies:** US#02
- **Priority:**Must Have

US#21

- **Description:**As a worker, I want to change the current status of the occurrence to inform the citizens and other entities.
- **Acceptance Criteria:**
  - **Scenario:** The worker wants to change the state of the occurrence.
    - \* **Given** The worker has a task assigned to him
    - \* **When** goes to the details of the task  
**And** change its state
    - \* **Then** the state of the task is changed.
- **Dependencies:** US#02
- **Priority:**Must Have

US#22

- **Description:**As a worker, I want to ask for human resources to resolve the occurrence to solve it better or faster.
- **Acceptance Criteria:**
  - **Scenario:** The worker needs the support of other teams to complete a task.
    - \* **Given** The worker has a task assign to him
    - \* **When** Goes to the details  
**And** press the support button
    - \* **Then** a new occurrence of support is created.
- **Dependencies:** US#02
- **Priority:**Must Have

US#23

- **Description:** As a worker, I want to receive notifications about the new occurrence that I was appointed to have that information.
- **Acceptance Criteria:**
  - **Scenario:** The worker needs to know what occurrence was appointed to him.
    - \* **Given** the worker is using his device
    - \* **When** a new assignment is appointed to him
    - \* **Then** a notification will appear with the task assigned to him.
- **Dependencies:** US#02
- **Priority:**Must Have

In light of a deeper analysis of the requirements and some orientation from the internship adviser that also is the client of the project, some requirements suffered some changes regarding to what was firstly presented in the first semester. Despite this, these changes were not very impactful in the final product and some of them only to correct ambiguous language that was used that could cause problems of interpretation.

## 4.4 Non-Functional Requirements

The Non-Functional Requirements ensure the quality of the system and define it. They are important to establish limits and constraints in the system and provide value to the product from knowing what are the system's boundaries, providing insurance and room for negotiation when trying to sell it. These requirements also describe the way that the system is expected to behave, which is described in this section that results from the qualities here enumerated.

- **Usability** - The system should be represented similarly in a variety of devices and should be focused on the end-user experience so that every user that is used to working with apps can easily understand the application's functionalities.
- **Performance** - The system should store all the requests and allow them to have a workflow in place to resolve each one of them.
- **Maintainability** - The developed product should follow good practices so that in the future, it can be easily modified by the hosting company allowing it to grow and add more functionalities.
- **Security** - The system should be capable of denying unauthorized access from users that are not registered in the system, as well as verify the level of access of the users to check if they have access to certain, more private, endpoints.

Scenario 1	
Quality Attribute	Usability
Source of stimulus	Internet Browser
Stimulus	User access the platform using a browser
Environmental Conditions	Normal conditions
Artifacts	Platform
Response	The system can adapt its interface to fit the interface
Response Measurement	The user can perceive the necessary aspects to interact with the application understanding what the application is supposed to do

Table 4.2: Scenario 1 - Usability

Scenario 2	
Quality Attribute	Performance
Source of stimulus	applications admin
Stimulus	User create a new workflow
Environmental Conditions	Normal conditions
Artifacts	System
Response	The system processes the order and responds to the user's output giving a message verifying that the workflow was added
Response Measurement	The user should see a positive or negative response in no longer than 1 second.

Table 4.3: Scenario 2 - Performance

Scenario 3	
Quality Attribute	Performance
Source of stimulus	applications admin
Stimulus	User adds multiple requests
Environmental Conditions	Normal conditions
Artifacts	System
Response	The system will make pre-visualisation of the created objects and then if no error were detected they can be added to the system.
Response Measurement	The system should be able to process the request and give a response in no more than 3 seconds for datasets with less than 100 requests.

Table 4.4: Scenario 3 - Performance

Scenario 4	
Quality Attribute	Maintainability
Source of stimulus	Maintenance Developer
Stimulus	a developer without any experience in the project wants to add a new feature.
Environmental Conditions	Development Branches
Artifacts	system
Response	The system should maintain the normal operation, having a new feature integrated.
Response Measurement	The feature should be developed, tested and integrated in the time that was established for it.

Table 4.5: Scenario 4 - Maintainability

Scenario 5	
Quality Attribute	Security
Source of stimulus	Unregistered Users
Stimulus	An unregistered user wants to access information that must only be visible to registered users.
Environmental Conditions	Normal conditions
Artifacts	System
Response	The user must get an error message saying that he does not meet the requirements to make that request.
Response Measurement	The system rejects the requests from these users, and can not access any private data.

Table 4.6: Scenario 5 - Security

Scenario 6	
Quality Attribute	Security
Source of stimulus	Registered citizen user without the necessary clearance.
Stimulus	An user wants to access information that must only be visible to admin users.
Environmental Conditions	Normal conditions
Artifacts	System
Response	The user must get an error message saying that he does not meet the requirements to make that request.
Response Measurement	The system rejects the requests from these users, and can not access any private data.

Table 4.7: Scenario 6 - Security

## 4.5 Restrictions

The only restriction here involved is the use of an open-source solution for the workflow engine so that it can be modified and used at will. This is due to business reasons so that Ubiwhere does not rely on other companies to upgrade or customize the solutions and allows them to have more freedom in those aspects.

This page is intentionally left blank.

## Chapter 5

# Architecture And Technology

The architecture of the software is important to represent the way the software is going to be made and shows the different parts of the software that are going to be implemented and the ones that already exist. This chapter explains the terminology used to describe the architecture as well as design. For the architecture definition, the elicited requirements and the established quality attributes will be taken into account.

### 5.1 Terminology

One of the objectives of describing a software architecture is for it to be easy to understand among all the stakeholders, whether they are developers or not, but also provides valuable content for the developers that are going to build the application. With that in mind, to describe the proposed architecture the C4 model [75] is going to be used.

The C4 model purpose is to resolve the not standardized notation and separate the architecture into different levels of detail [75]. This model allows, in maximum, four diagrams of depth.

### 5.2 Architecture

The platform that will be developed is expected to communicate with final users: citizens that need to Report their issues and need information about the existing problems, Admin users that are going to maintain the normal operation of the application, and city work crews that need to know the location and the information about the next job. Although the application needs to communicate with different end-users, it is also important that the application can gather information from other sources such as other platforms where the citizens can report their problems. In the first diagram of the C4 model, the context diagram, these relationships are explained, and how it is intended for them to communicate, showing an overview of the whole system.

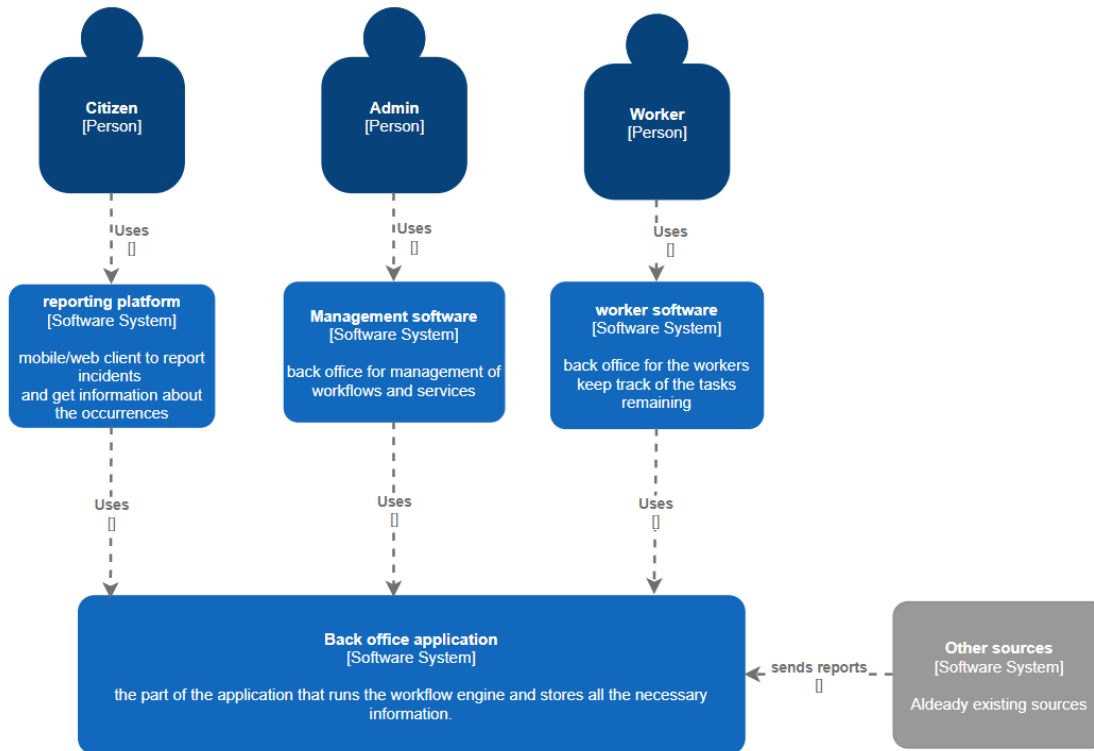


Figure 5.1: Context Diagram

There are two architectural approaches that can be used: The monolithic and the microservices methodology. The monolithic solution is normally built as one single component that is developed, tested, and deployed as a whole, while a microservices approach is split into multiple different and independent components that interact with each other.

The older and traditional way to produce software is using monolithic architecture. These systems bring advantages such as easier development, deployment, and testing, and fewer problems in terms of integration and dependencies since they work as a whole. The downside of these types of architectures is that the components in them become tightly coupled and entangled that influence management and scalability[110]. This downside leads to prolonged development time and more extended downtimes of the product when making changes or repairs.

Microservices are a way of breaking these Monolithic approaches into loosely coupled modules that communicate with each other through APIs[90]. This makes the system more adaptable to changes and more fault-tolerant since if one component fails, it will not break the whole system. Because of the decoupled services, developers can easily understand the specific service's function, reducing the time needed to make changes or implement new features, on the other hand this significantly increases the complexity of the system and the maintenance of the communication layer. Although the need for maintenance of the communication layer, as long as that layer continues to remain unchanged the services that depend on it will not suffer any changes. Testing and deploying the application is also more challenging since more components need to be tested and deployed.

For this internship, a Microservices approach was chosen, where the different components were divided in a logical way for them to work independently from each other. This choice was made so that in the future if the hosting company wants to replace or add applications that require communication with the application created in this internship it can do it



easily once the different components of the system are decoupled from each other.

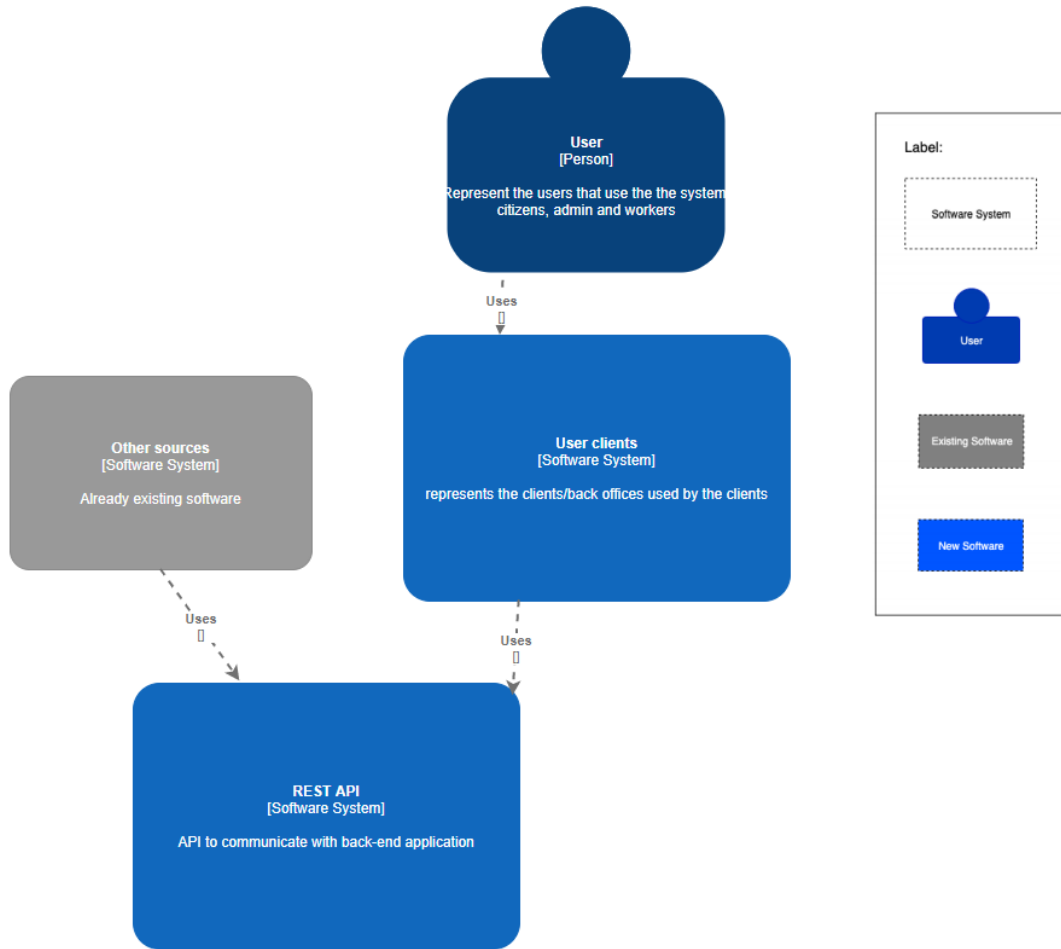


Figure 5.2: User Integration Architecture

In this figure, Figure 5.2, the communications between the users and their applications is shown as well as eventual sources of information that communicate with the REST API of the back-end application.

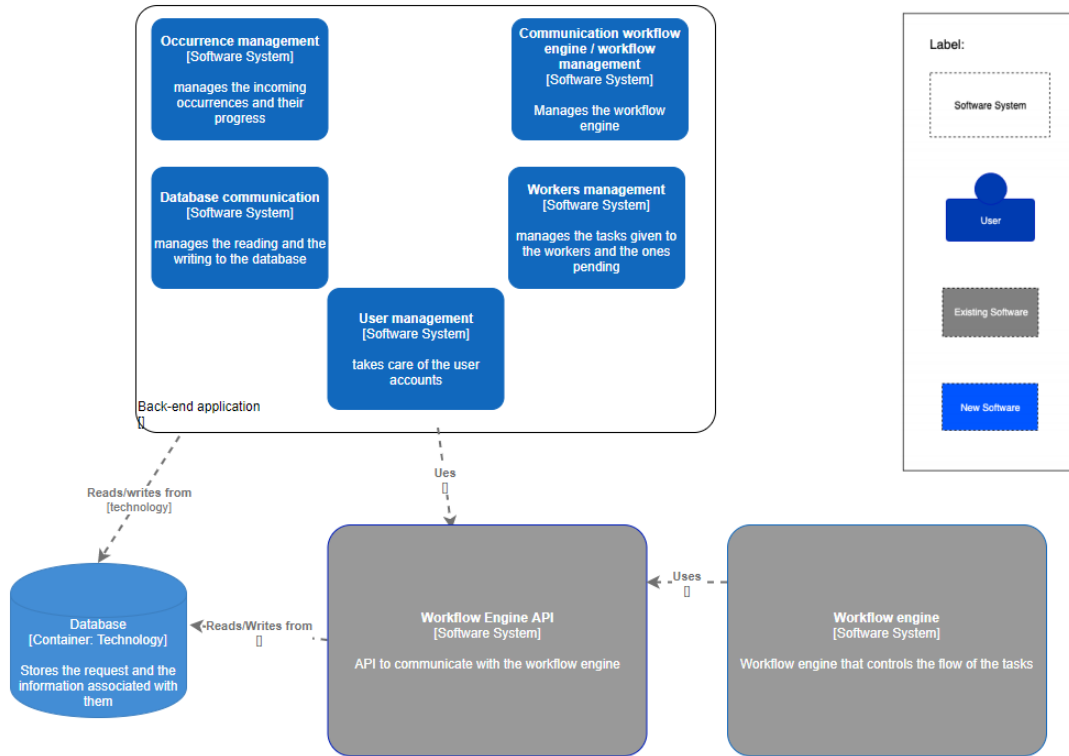


Figure 5.3: Back end detailed proposed Architecture

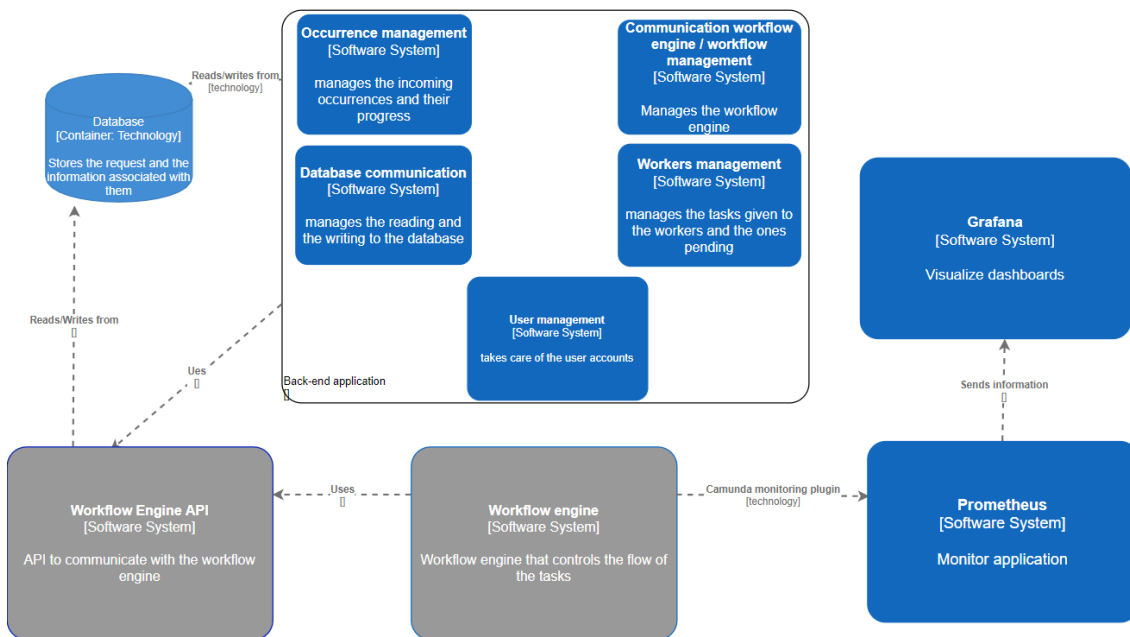


Figure 5.4: Back end detailed final Architecture

in image 5.4, the Back End application detailed systems are represented as well as other systems that interact with it like the workflow engine. The Prometheus and Grafana applications that communicate with the workflow engine to represent the engine metrics are also here represented. The proposed Architecture, image 5.3, just varies from the final Architecture because of the added systems to monitor the workflow engine, the Prometheus,

and the Grafana.

The added systems, Prometheus and Grafana, were necessary to implement the requirement expressed in the user stories number 14 and 15 due to limitations in the chosen external system, the Camunda Workflow engine, that were only discovered after the development process had started.

### 5.3 Technology

To determine what technologies to use in this project were considered some factors like the restriction made by Ubiwhere that the software must be open source. In this case, the product that was affected by the limitation more directly was the workflow engine.

For the **web application**, the chosen technology was reactJS. This choice was made because it is a technology that is easy to learn and, thanks to the optimization of the rendering methods, a high-speed technology. Another reason that had value for selecting this technology was the fact that it has excellent support from its maintainers, Facebook, and by its community. Besides that, it is a framework that Ubiwhere's employees have experience with, making it easier for the intern to learn and for them to maintain in the future. Other technologies such as Angular were taken into consideration as well.

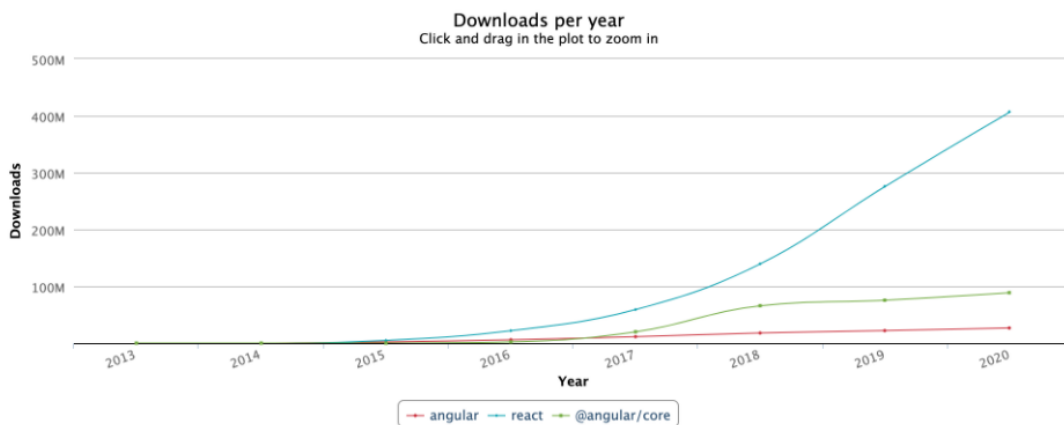


Figure 5.5: Number of npm downloads angular and react

As seen in the image above, the number of npm downloads per year in react tend to increase faster than angular. Another reason that took part in this decision was the fact that react has faster performance due to the virtual DOM that reduces the overload of the browser, it has a HTML-like syntax which allows templating and highly detailed documentation and is overall easier to learn than Angular [111]. Another big advantage of react is the support from the hosting company once experienced users can assist with questions that the intern may have in the development.

The hard part when looking for the right **workflow engine** was not the limitation imposed by Ubiwhere that states that the workflow engine must be open source, but the fact that there exist multiple open source workflow engines and which one of them have its pros and cons that must be verified before choosing a solution. This work was partly expressed in the chapter on workflow engines, where only the technologies that were considered relevant for this work were analyzed. The final choice was to use the workflow engine Camunda, which should fulfill all the expected requirements. The choice was made considering the surveys

i.e, documents that compare multiple solutions in this case workflow engines, analyzed and the official websites of the technologies.

The use of a plugin to obtain the metrics for the workflow engine involved the use of two more technologies. These two technologies were a technology to monitor the workflow engine and another to visualize the gathered metrics. The chosen technologies were **Prometheus** and **Grafana**, respectively. These two technologies were chosen by recommendation of the plugin makers and, in the case of Grafana, the one that actually had some support in the plugin page. In the case of Prometheus, the plugin makers recommended its use although it is said that any monitor tool should work as well, but as this was the recommended tool and therefore the intern decided to use them.

For the **database**, PostgreSQL was chosen to serve the services of the system. PostgreSQL has an active community and high performance and it is a database engine that the intern was already comfortable with. Since the intern already had some experience working with this database engine and it was a supported solution for both the workflow engine and the as well as the back-end application, it was decided to use it.

To develop the **REST API** and the associated back-end application, Django REST framework was chosen. The choice of using Django REST framework was due to many variables. To begin, Django makes use of a powerful Object-relational-mapping (ORM), which simplifies dealing with the database. Without it, the developer would have to create tables and queries that could become quite complex. Besides, Ubiwhere has professionals with years of experience in the framework, allowing the intern to learn and develop among side trained personnel. For developing the API, Django Rest Framework provides a Web Browsable API and extensive documentation that can be useful to resolve upcoming issues. To choose this technology there was a process of selection and other frameworks came into question, and one big competitor was Flask. Flask is a micro framework that offers positive features such as performance due to it being so lightweight. This is a good feature and for it being more lightweight than Django REST framework makes a lot of sense using it in a microservices architecture, but despite this, Django offers Django REST framework's browsable API, versioning support, and other features such as the admin page that was used as an admin user page. These features make Django REST framework heavier, but more useful for the final product and therefore the best choice for this project.

## Chapter 6

# Development

The project development was divided into three modules: The workflow engine, the back-end application and the front-end application. The technologies used to develop these modules were the Camunda Engine as the chosen workflow engine, django REST framework, a python framework to develop REST frameworks, and React for the front-end development.

The back-end application makes the bridge between the front-end application and the workflow engine providing it with information from what tasks require human work and the ones that do not. The ones that do are described as tasks that the worker can claim. Once this task is claimed by a worker only an admin user can remove that task from that user by disassociating it from the user that claimed it and making it claimable again. After a worker claims the task, the task will appear in the complete task section of the front-end application where the user can click and complete the task that he has committed to.

### 6.1 Workflow Engine

A workflow engine is a software application that helps organizations initiate and automate tasks. It manages, monitors and determines the next task depending on the process definition. It can also be useful for organizations to streamline processes by allocating tasks and communicating data to both employees and management. [102] As previously mentioned, the chosen workflow engine was the Camunda Engine.

To deploy the workflow engine a image of Camunda was used that allowed it to have an instance of the engine running. This instance allows the REST API to be accessible to the users. On top of that, is also available a user that is automatically created with the credentials demo for the username and password that can be used to access an interface. Another user that is automatically created is the first admin user in the Django application. Both of these users can access Camunda's interface that allows for some interaction and visualization of the processes while they are running, giving an admin view of the processes that are running. Despite this, the users created in the Django application are not created with the intention of directly interacting with Camunda's interface, but rather to claim and complete tasks via the front-end application.

### 6.1.1 Process Definition

To define the processes in Camunda, it is required to use software made available by Camunda's team, Camunda Modeler. In the Camunda Modeler it is possible to describe the processes using BPMN 2.0. A process is defined by a set of activities that represent a task. These activities can be of many kinds, but for this project two were mainly used and worked on that were the User Activity and the Service Activity. These two types of activities alone can be responsible to describe very complex processes in as much as they represent a broad number of actions. The User Activity describes an activity that requires a user to resolve, in the context of the project, this would mean an intervention from a worker and, in other hand, the Service Activity is an activity that describes an automatic event that can be anything the user wants it to be. However, the use of automatic events implies a javascript worker that is integrated with the activity through a topic that is defined in the activity. Once the diagram is deployed, the javascript worker can subscribe to the topic and wait for any activity. It is also possible to handle errors that occur in the Service Activities in the javascript workers and associate tasks when these errors exist. It is not possible to verify that an error has occurred in the applications developed by the intern, but it is possible to see it in the workflow engine visual tool. This is not a problem since if the error exists in the diagram it was an error that was expected to occur and therefore the necessary precautions have already been taken place.

The process of creating these processes using BPMN 2.0 is not straightforward and implies a know-how from the user that is performing this job. The major obstacle that involves the creation of these diagrams in the service tasks once they require a custom javascript script to be executed in order for them to work. A positive side of this is that since the javascript script is man made sometimes specifically for that service task, the service task can do anything that can be coded in javascript and for other tasks that are recurrent in a service activity a list of pre-made set of javascript workers can be made and deployed and the same topic applied several times. The output generated from the service tasks can also differ from instance to instance once it is possible to have variables in the diagrams which make it a positive feature.

To deploy the process created is possible to do it though the modeler application, but for the sake of the project and for the model created to be recognized by the back-end application, the deployment of the model must be done though the back-end application or the REST API. After this the model is sended to the Workflow engine and it is possible to create a new instance of this model that would represent a new occurrence.

### 6.1.2 Workflow Engine metrics

To access the metrics of the workflow engine, and since the REST API from the Camunda Engine did not meet the requirements needed, a plug in for the camunda engine was used, the Camunda Monitoring[93]. Camunda Monitoring is a plugin that provides the Camunda BPM engine with configurable and script-based metric data. The data produced can be consumed by numerous monitoring tools, but for this project the Prometheus time-series database [94] was chosen. This plugin is very recent and for that it is only possible to get certain default metrics. This plugin sends the gathered metrics to the Prometheus database and then another tool, Grafana[95], comes to use. Grafana allows for the visualization of the collected metrics by a series of graphs. In this case one graph is used for each of the metrics. These graphs are only visible on the grafana dashboard, however it is one of the things that can be talked about for future work to integrate on a platform that is not the

Django admin.

## 6.2 Django Application

The Django application can be divided into two major sections. The Django admin application was used to implement the admin user platform and the REST API to provide information and access to other applications such as the workers' application that was created. In this case the REST API was used to obtain authorization and to collect information about the occurrences.

### 6.2.1 Models

The models created to represent the occurrences and the types of occurrences were the ones established by the open311 standard, Service Type, and Service Request. The Service request object represents a request made by a user and the service type represents a type of issue that can occur. Since the objective of the project is to integrate the application with the workflow engine, it was necessary to add more fields to the standards to achieve that goal. The models that support these integrations are **VariableWithValue**, **VariableBpmn**, **Attribute**, **Value**, **BpmnDescription**. The model **VariableBpmn** specifies what variables are present in the BPMN model and the model **VariableWithValue** specifies that variable with the value. This means that the **VariableBpmn** has a relationship with the model **BpmnDescription** which has a relationship with **ServiceType**. The Value also has a relationship with the object **Attribute**, this represents the multiple values that the attribute can have, the attributes then also have a relationship with the **ServiceType**. These attributes are required by the open311 standard and this was the way chosen to implement the standard. The model **VariableWithValue** has a connection with the **ServiceRequest** in case of it having necessary Variables. The diagram with all of these relations can be seen in the next image.

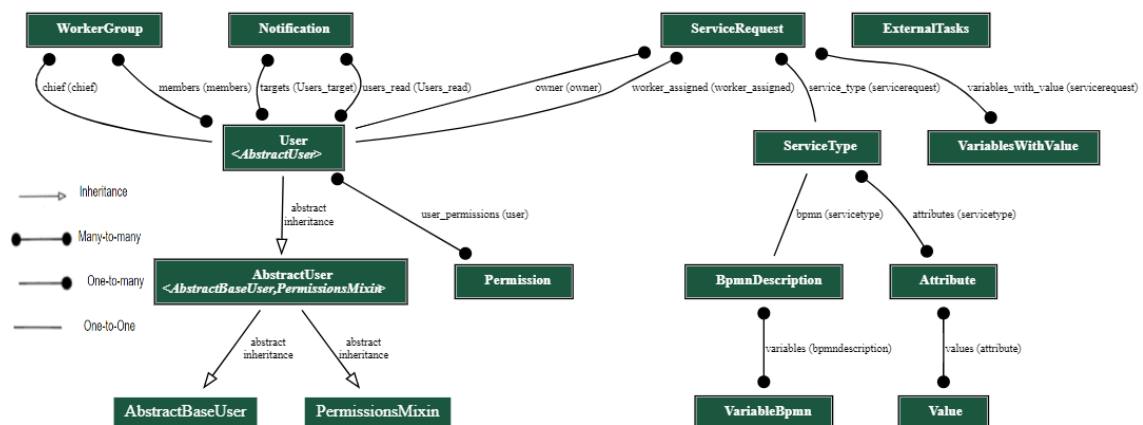


Figure 6.1: Relationships between the models of the back end application

In this diagram it is also possible to see the permissions sections. The permissions that are referenced in the diagram were not created by the intern but they were auto-generated by the Django framework. However, these permissions are used to check if the user can access the API endpoints. The users can have three security levels: normal user, Staff user and Superuser. The normal user is equal to the Citizen user described in the requirements,

Staff user is the Worker and Superuser the Admin. A full version of the diagram with all Django automatically created models can be seen in the annexes, chapter 10.

### 6.2.1.1 Model Signals

To establish a unique implementation of the logic in order to facilitate the process of future modifications of the code, Django signals were used. These signals were very useful in cases where changes or creations of objects could be made through different channels, in this case the possible channels are the API and the Django admin web page. Django signals allow certain senders to notify a set of receivers that some action has taken place involving the model that the signal is associated with. These actions can be in the form of creation or changes in some object of that model. The signals were implemented only for the models that required some form of modification or integration with the workflow engine such as the Service Request, User and Service Type.

In order to represent the instance of the occurrence that runs on the workflow engine an ID generated by the workflow engine is stored in the Django application. This ID is obtained from the workflow engine when the instance of the model is created. When created a signal is triggered that sends a request to the Camunda API which responds with the said ID. This integration can be seen in the image below.

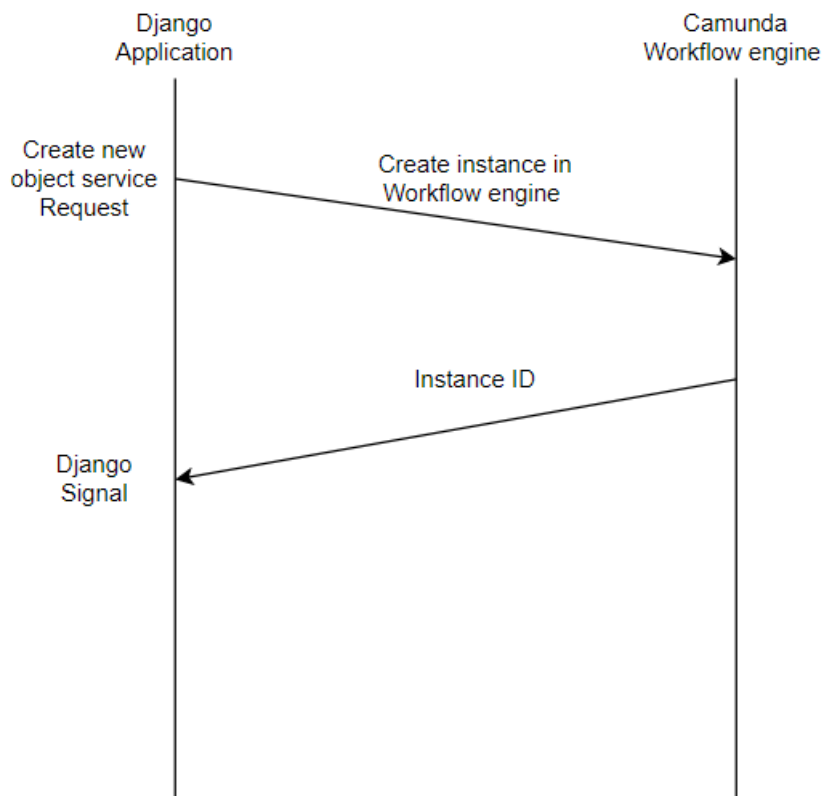


Figure 6.2: Diagram representing the process of obtaining the process id from Camunda

### 6.2.2 REST API

One of the major objectives of this project was the development of a REST API so that it would be possible to integrate the application with other applications. An example of



this objective is the developed frontend application that makes use of the REST API in order to obtain a session for users and the information necessary so that the workers could resolve the occurrences.

Some requests in the REST API do not make simple requests like ones seen previously in the Model Signals to the workflow engine and require more requests to achieve the wanted result. An example of this are the requests to claim and complete tasks that are available to the workers via their frontend application. These tasks require a second ID to be obtained, the activity ID, in order to be able to make actions on them.

The reason that it is not possible to keep the ID required for these tasks, instead of the one that is kept, is because the ID required is only created when the instance passes through a user activity. Since not all the processes are required to have a user activity, it would be impossible to store that ID and still have access to the instance in the engine. The reason that this approach is possible is that the instance is only in one activity at any given time, since the same instance was on two activities at the same time other steps would be necessary to make sure that the right activity ID is being retrieved. This process can be viewed in the image below.

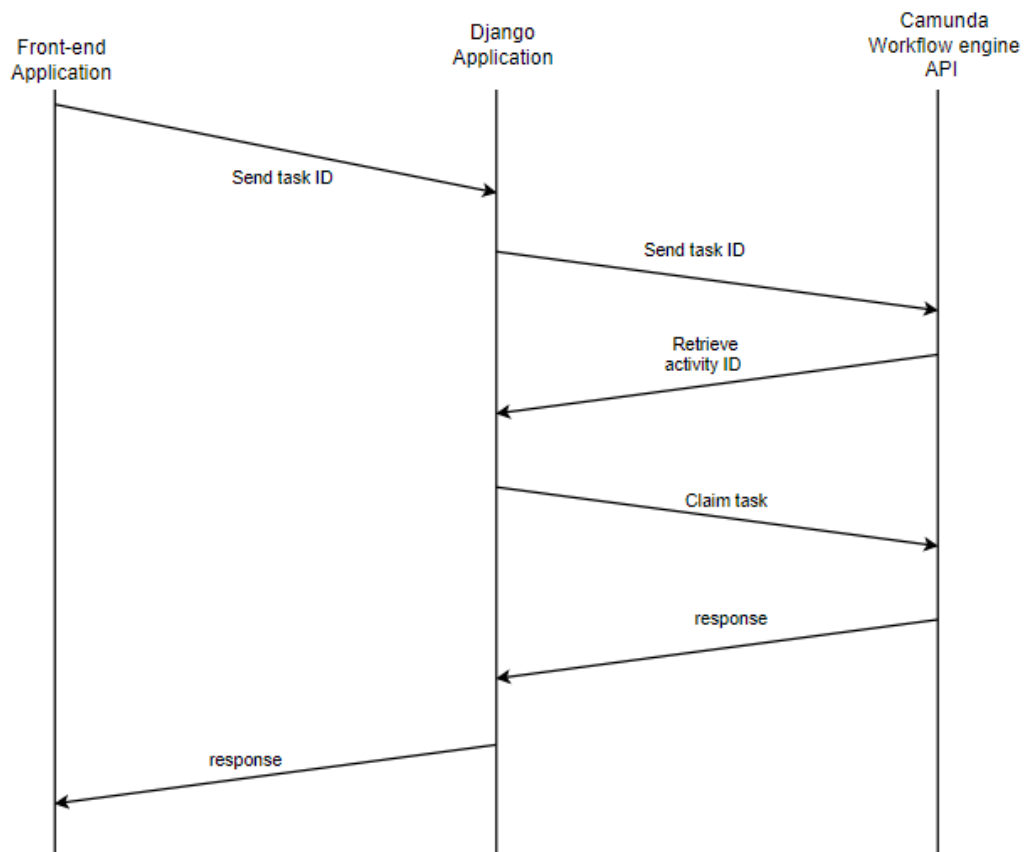


Figure 6.3: Diagram representing the process of obtaining the activity id using the process id

### 6.2.2.1 Permissions And authentication

This API has implemented permissions to make sure that some endpoints were not accessible just knowing the correct endpoint. These permissions ensure that the user that is

making the request has the necessary credentials (citizen, worker or superuser) in order to make the request that he is attempting to make. To verify the permissions it is necessary for the user to provide a JWT in the request. JWT (JSON Web Tokens) provides a token to guarantee the user can have access to the requests. This token expires within every hour upon getting that forces the user to login again in the application to get another valid code. The JWT token provides an easy way to make the authentication between the two applications and once Django REST framework has packages that implement this solution, the integration with the backend application was very smooth.

### 6.2.2.2 Swagger

To provide documentation for the API was created a swagger that provides the necessary information about the endpoints so that the application can be integrated with others. By having a swagger for the application is possible to learn several things about the API [115] :

- What operations are supported by the API
- What are the API parameters and endpoints
- What each endpoint returns.

All of this information can be useful when integrating with a new application or for testing purposes. The swagger generated by the backend application can be seen in the root page of the Django application and it can also be downloaded to share with other developers that do not have access to the swagger first hand. The swagger provides information about the endpoints and what methods are available in those endpoints and the information required to make requests. This was one requirement that was not functional, but necessary for future upgrades in the Django application as well as integration with other applications that can be made or that can be connected with the backend application.

### 6.2.3 Django Admin

In the django admin page it is possible to make several tasks such as *creating processes*, *add users*, *create groups* and *create requests*. The django admin is used mainly to create and change objects. To assist with this and to recreate scenarios or just to create multiple requests at once because the admin user had the occurrences in a file it was implemented an import and export package. This import and export package gives the user an option that allows them to select a file and import that file with the object's descriptions. This functionality is also used to store the information locally once it is possible to export the information out of the application.

## 6.3 Front End

Front-end applications can have many forms and shapes, but essentially a Front-end application is a graphical interface that the user sees and interacts with. [103] As mentioned, it can have several forms, but one of the most common, and the one that was implemented in this project, was in the form of a website. To write a website three types of technologies are usually used: Text Markup Language (HTML), Cascading Style Sheets (CSS) and

JavaScript code. To simplify their development javascript frameworks are used that make this process much easier. In the case of this project the chosen framework was React. React was first created by Jordan Walke on Facebook. React first deployed on Facebook's newsfeed in 2011 and on Instagram.com in 2012 [104]. There are many upsides of using react, some of them being without a doubt the use of reusable UI components that contribute to an easier way to maintain the project by simplifying the process required to make changes in the UI, and the possibility to create large web applications that can change data, without reloading the page.

### 6.3.1 Mock-ups

The front-end application created in this internship made use of the previously mentioned features with the objective of making the best use of the chosen framework. The created application went through a process of making and verifying the design out of the system requirements, in this case the user stories. This process deeply impacted the final result that was obtained since it would be an application that would be used by users that do not have any knowledge of how the system actually works. This result can be seen in the following images in the form of mocks-ups.

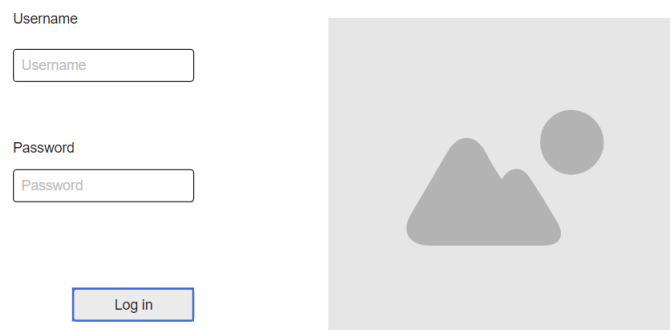


Figure 6.4: Worker application login page mock-up

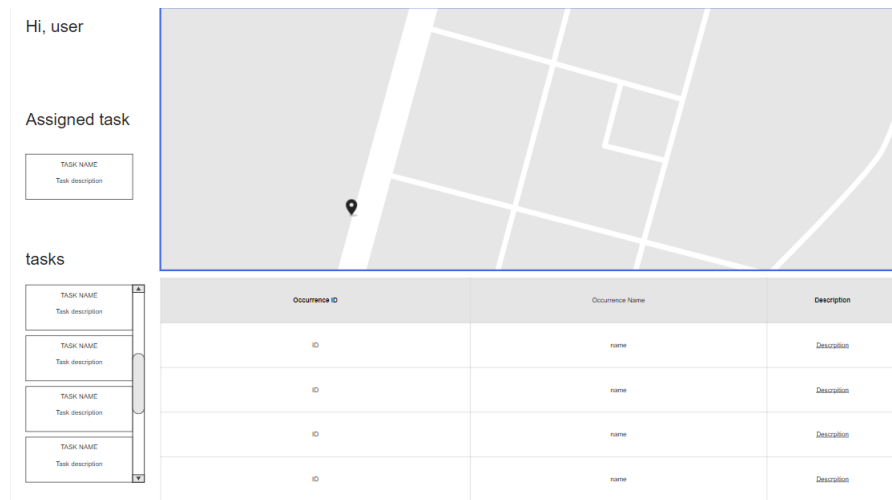


Figure 6.5: Worker application dashboard mock-up

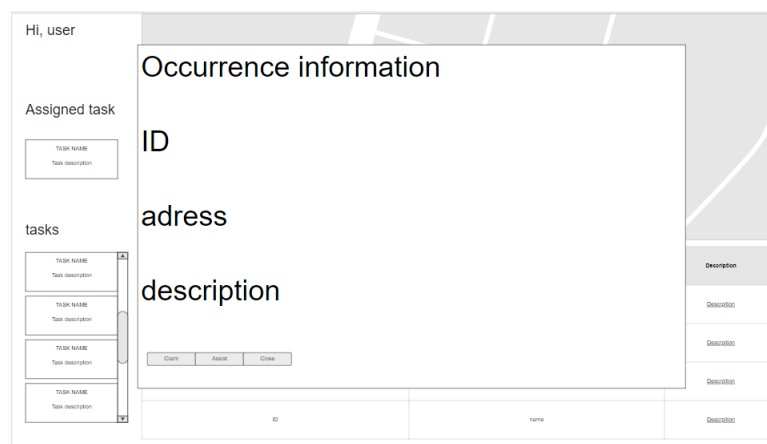


Figure 6.6: Worker application occurrence description and possible actions mock-up

Many of the designs and features present in the mock-ups were present in the final product, sometimes with some differences from the initial. These differences are more impactful in the design than the features, although some did end up with some differences as well. Some of these differences were to obtain gain from the application by changing some functionalities and others were to obtain a better visual experience for the end users.

### 6.3.2 Application Overview

To make the operations that the users will perform, the user must be logged on which can only be done if he has the permissions to do so. To ensure this, the application is composed of two pages, the **login page** and the **dashboard page**. The login page requires a user to input his username and password and, once the user presses the button “log in”, a request is made to the back-end application to ensure that the user exists in the system and has the permissions required to access the application. If the information is verified, a JWT token is sent to the front-end application, which ensures the authentication for the remaining requests that are possible to make in the dashboard page.

The information displayed for the user in the dashboard is fetched from the back-end application created by the intern. This information is obtained from the backend through

requests that require authentication since the information and the possible actions that the user can do are sensitive and is necessary to have extra security. This security is implemented throughout all of the application using the JWT Token.

The relevant information in the front-end application is present in the map, the table and the side bar. The map has pinned down every occurrence registered in the system where it can be claimable or not. The ones that can be claimable are pinned down in a green tone pin and the ones that can not are pinned down with a red tone pinned. This information can help the workers get to place where they need to be or provide more information about the occurrence in hands.

To claim and complete tasks the user must click in the tasks present in the two lists present in the side bar. In the list with the name "tasks" are represented all of the tasks that can be claimed by the user, and in the list with the name "Claimed tasks" are the tasks that the user already claimed and can mark as complete. Each one of these functionalities require communication between the workflow engine and the workflow engine and this application works as a trigger for this communication.

Another information that in gathered from the backend application is the notifications for each specific user. These notification can either be custom made from the admin user in the admin application or automatics each time a user claims or completes a task.

## 6.4 Environment

In order to start the process of development it is necessary to build the environment required for the tools that are going to be used. In architectures like the ones seen in this project (microservices), it is often a challenge to achieve integration between the multiple components that compose the final system [96] especially when third-party applications are in the mix. To install the components of the application directly on the system would be a task prone to failure once it would be required to change the system where the application was developed on, making it hard to reproduce in another environment.

### 6.4.1 Virtualization

One way to overcome this situation is through the usage of virtualization technologies. Virtualization is the process of running a virtual instance of a computer system in a layer abstracted from the actual hardware [98]. In terms of virtualization, there are two major approaches: a Virtual Machine and a container.

Both of the approaches allow for running applications in an isolated system although a container is not a fully independent machine [98]. In a virtual machine, it is possible to install all of the services and dependencies required and maintain them once it is offered in an isolated environment where it is possible to do so. However, a virtual machine can become very resource-intensive since it not only runs a copy of the operating system but a virtual copy of the hardware that the operating system needs to run. On the other hand, in a container the hardware is not virtualized, just the OS is. [97]

A tool that is being used more and more in the software community as well as in the company, is docker. Docker is an open-source tool that is used to build, run and deploy Linux containers. Instead of replicating all of the operating system, the application within containers share both binaries and libraries [99] making them more lightweight. These

differences can be seen in the following image.

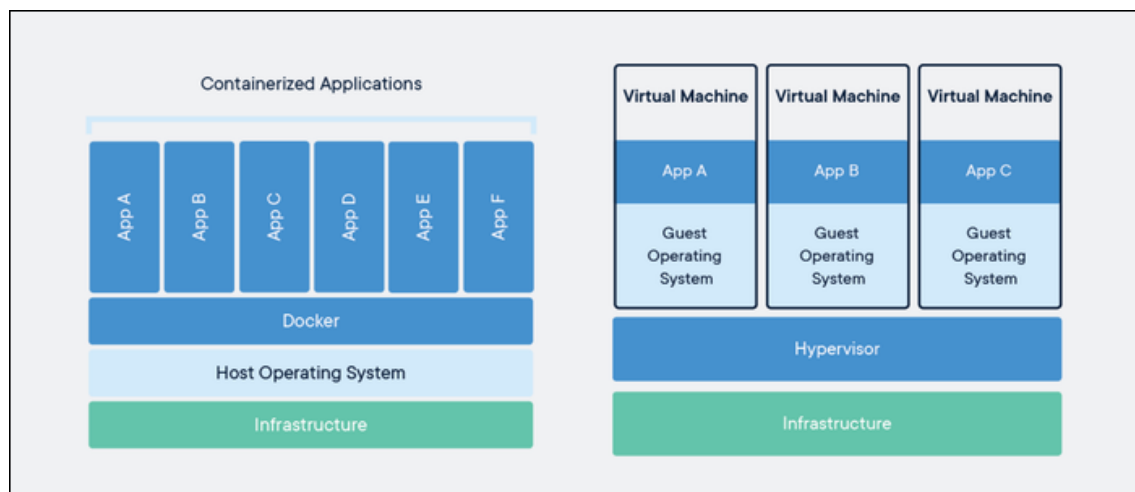
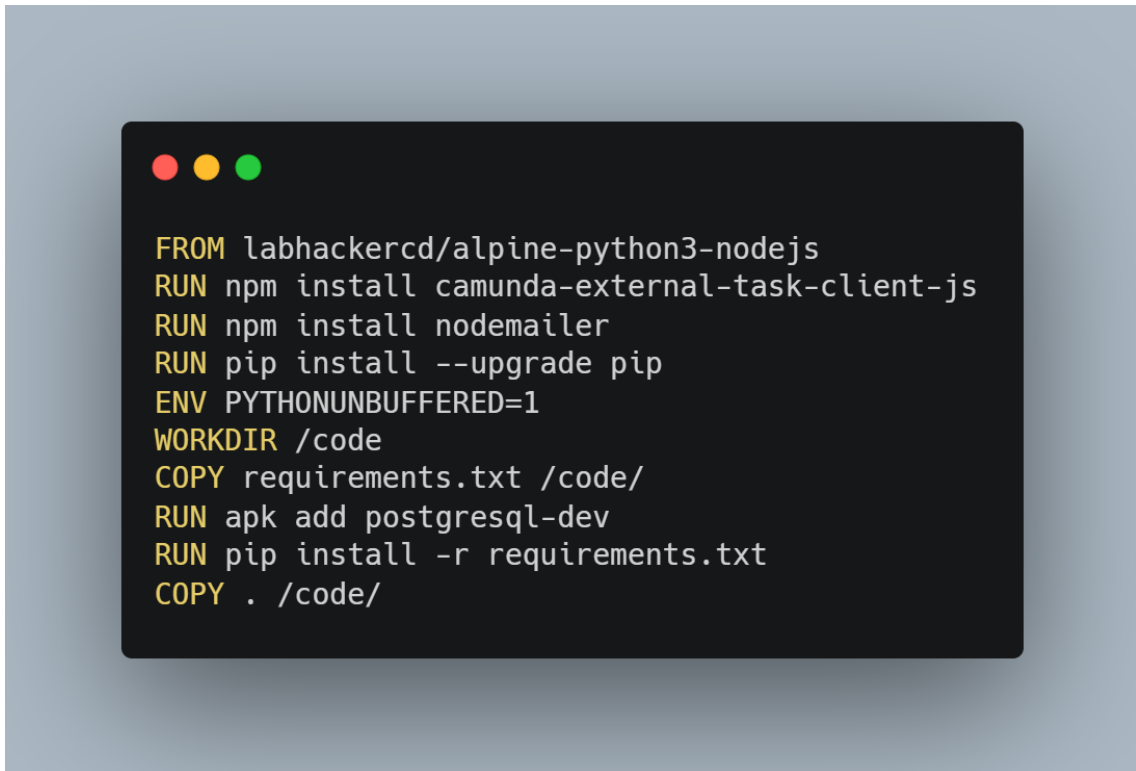


Figure 6.7: Docker vs Virtual machines

Since it is the most advantageous option and opens a window to improve the system by adding other tools such as Kubernetes to provide high availability, all of the services in this project will run in docker containers. Some of these containers are provided with a specification, a Dockerfile, and others use an image that is already ready to use. The ones that do not require this specification are systems where it wasn't necessary to touch the source code, like the camunda or the postgres instance, and therefore all of the necessary changes and tweaks could be made making resources to config files. The ones where the source code was elaborated by the intern, like the backend application and frontend application, a Dockerfile is required. A Dockerfile is a text document that contains all the commands a user could call on the command line to assemble an image [100]. in the following image is possible to see the Dockerfile of the Django application, image 6.8.



```
FROM labhackercd/alpine-python3-nodejs
RUN npm install camunda-external-task-client-js
RUN npm install nodemailer
RUN pip install --upgrade pip
ENV PYTHONUNBUFFERED=1
WORKDIR /code
COPY requirements.txt /code/
RUN apk add postgresql-dev
RUN pip install -r requirements.txt
COPY . /code/
```

Figure 6.8: Django application dockerfile

As is possible to see in the image 6.8, the process begins by getting a base image with the Alpine Linux OS, with Python 3 and NodeJS installed. The usage of an image with python and nodejs is due to the backend being developed in Django, i.e Python, and the integration of the BPMN models activities be though javascript clients that require NodeJS. Since it is the Django application that calls the NodeJS clients it is necessary to have an image of node in here for them to work. In the following lines, the necessary packages are installed. In this case we have the Camunda external task client that is used by the workers. nodemailer, that was also used in the javascript client, was also installed. After that, the rest of the requirements are installed and the source code is copied.

The other Dockerfile present in this system is for the react application that can be seen in the following image, image 6.9.

```
#NGINX CONFIGURATION
# build environment
# stage1 as builder
FROM node:10-alpine as builder

# copy the package.json to install dependencies
COPY package.json package-lock.json ./

# Install the dependencies and make the folder
RUN npm install && mkdir /react-ui && mv ./node_modules ./react-ui

WORKDIR /react-ui

COPY . .

# Build the project and copy the files
RUN npm run build

FROM nginx:alpine

#!/bin/sh

COPY ./nginx/nginx.conf /etc/nginx/nginx.conf

## Remove default nginx index page
RUN rm -rf /usr/share/nginx/html/*

# Copy from the stahg 1
COPY --from=builder /react-ui/build /usr/share/nginx/html

EXPOSE 3000 80

ENTRYPOINT ["nginx", "-g", "daemon off;"]
```

Figure 6.9: React application dockerfile

In this Dockerfile the process is the same except for the use of a nginx server to provide more efficiently to the application. This is due to the react application only using static content and thus is more efficient to use a faster server such as nginx.

## 6.4.2 Deployment

To deploy these Dockerfiles and some other containers docker compose was used. Compose is a tool for defining and running multi-container Docker applications.[101] The docker-compose file uses YAML, a human-readable data-serialization language, to describe the configuration of the system services and, with this, reducing the effort required to run docker images. In the image 6.10, it is possible to see an example of how a service is



declared in this file, in this case the example is the Django application.

```
web:
  container_name: django_Backend
  build: ./autonomous-occurrences
  volumes:
    - ../autonomous-occurrences/code
    - autoOccu:/home/web/autonomousOccurrences
  command:
    bash -c "
    sleep 25
    && python3 manage.py makemigrations occurrences
    && python3 manage.py makemigrations
    && python3 manage.py migrate
    && python3 manage.py test
    && python3 manage.py ensure_adminuser --username=tomas --email=tomasgabriel1998@gmail.com
    --password=password --first_name=tomas --last_name=lopes
    && python3 manage.py runserver 0.0.0.0:8000"
  links:
    - camunda
    - db
  depends_on:
    - db
    - camunda
  ports:
    - "8000:8000"
```

Figure 6.10: Django application service in docker-compose file

Here it is possible to see what commands are used to run the application, what are the application dependencies, i.e. other services that need to be running before this one, the volumes used by it and the ports that are going to be used. Volumes are the preferred mechanism for persisting data generated by and used by Docker containers and are completely managed by docker[112]. A more simplistic use of the docker-compose file is when it is not necessary to use commands to run commands which implies that a pre-made image of the service is going to be used. An example of these is the Camunda workflow engine service, image 6.11.

```
camunda:
  container_name: camundaEngine
  image: camunda/camunda-bpm-platform:run-latest
  environment:
    - JMX_PROMETHEUS=true
  ports:
    - "8080:8080"
  expose:
    - 8080
  volumes:
    - engineVolume:/home/web/workflowEngine
    - ../autonomous-occurrences/Camunda-Monitoring.jar:/camunda/configuration/userlib/Camunda-Monitoring.jar
    - ../autonomous-occurrences/metrics:/metrics
    - ../autonomous-occurrences/camunda-monitoring-beans.xml:/camunda/configuration/camunda-monitoring-beans.xml
    - ../autonomous-occurrences/default.yml:/camunda/configuration/default.yml
  networks:
    - default
  restart: unless-stopped
```

Figure 6.11: Camunda workflow engine service in docker-compose file

In this service it is possible to see that commands are not used but rather volumes that contain configuration files. This configuration files are used to extract metrics from the Camunda engine to Prometheus and as is possible to verify an environment variable is also set for this service to run.

The final system is composed of six containers that include the front-end application, the back-end application, the Camunda workflow engine, Grafana, Prometheus and also a Postgres database all described in the docker-compose file to allow for a more concise deployment.

# Chapter 7

## Testing

Software Testing is a method to check whether the actual software product matches expected requirements and to ensure that software product is defect free [105]. This process can be either automated or manual depending on the type of product and rather it is pertinent depending on the deadlines since many of the tools to automate tests have high learning curves. With any of these methods, properly tested software ensures reliability and security which further results in time saving, cost effectiveness and customer satisfaction. Due to the increasing complexity of newly made applications, especially the ones settled in microservices, testing is even more important in software development so that end users do not encounter failure when using the final product.

A big portion of the tests, especially the Unit tests, is meant to verify the developed REST API that ensures the communication between other systems, in this case the front end application, and in the future other applications that can be developed making resource to it. The reason this is the most tested part of the system is because it represents a big part of the architecture that was chosen.

With the objective of isolating the system while testing it in order to prevent that third parties application, such as the workflow engine, could negatively affect the tests and also to prevent the necessity of later undoing the test case, mock-requests were used to simulate the integration while testing.

### 7.1 Unit Testing

Unit testing has the objective of isolating small portions of the system and verify that their functionalities are what is expected of them. Is considered a small portion of the system small enough parts of the system that can work isolated from the rest of the system. To verify this a set of predefined inputs and outputs are made following the logic of each component and then the results are crossovered with the ones predefined to assess if the system is working properly.

In this project, the intern composed a series of tests that describes the behavior of the created system, in this case these types of tests were mainly applied to the back-end application made in django. The goal of writing these tests was to assess during the development if the application maintained its normal functionality when changes were made.

Mock requests were used to eliminate the dependence of the application with the workflow

engine in testing. The mock requests allow the developers to make sure that their application is independent from third party software when testing their application but without changing the code in the application every time. An example of this can be seen in the image below, image 7.1, where a mock request is applied in a test case.

```
@mock.patch('requests.get', side_effect=mocked_requests_get)
def test_occurrences(self, mock_get):
    print("Test 10")
    client = APIClient()
    response = client.get('/occurrence/')
    self.assertEqual(response.status_code, status.HTTP_200_OK)
```

Figure 7.1: Mock request exemplification in test case

Once the request is made, using the `requests.get` function anywhere within the function, the request is redirected to the `mocked_requests_get`, where a predictable response is sent to the request, making it more predictable to the developer once there is no contact with third party applications.

The scenarios tested as the development of the system are listed in the Annexes. The goal of the scenarios is to verify if the access of the resources is only given to the users that have the right to do so according to application requirements. The resources tested were a combination of:

- Credentials
  - Without credentials
  - with wrong credentials
  - with right credentials
- Resources
  - Occurrence
  - Claim Occurrence
  - Notification
  - Service type

The actions that are applied to the resources vary from each other depending on the type of resource, but the possibilities are: create, list, retrieve. None of the actions to any of the Resources is deleting, since it was not a requirement of the system due to its pertinence. The test coverage is 88% of the back-end application. The coverage percentage should not be taken as an absolute figure in terms of how well the system is tested, since it only tells us the percentage of code that is run during the tests, which is not a guarantee of the quality of those tests. However, it still indicates, in this case, that a large percentage of the produced code is tested.

Every change implemented in the project had to comply with the tests to make sure that the requirements were met. When applying new features to the application, these tests also make sure it did not negatively impact the work previously done.

## 7.2 Functional Testing

Functional tests are used to confirm that functionalities of a system are behaving as expected [106]. These kinds of tests can be automated, but due to the complexity of modern application because of screen sizes and the general complex UIs, is often not doable in useful time due to the learning curve of tools to do it. this is what ended up happening in the internship, and it would not be possible to include them in the useful time of this internship. These tests validate the software to see if it matches the requirements by giving appropriate input and verify if it matches the acceptance criteria.

To perform functional tests a step-by-step plan with the following steps should be followed [114]:

- Understand the functional requirements
- Identify what is the input required for that test
- Have an expected output of the given input
- Execute the test case
- Verify if the output matches the expected output

If all of these steps are successful, it is possible to say that the test passes, otherwise the test fails and that functionality should be once again sent to the developers team to be reviewed.

The functional testing done in this internship was validated by the acceptance criteria in the user stories that matched the requirement and then tested to see if the requirement was developed as intended. For each of the scenarios, after the preconditions were satisfied, the intended actions were performed following what was described in each one. The results of the tests can be seen in the annexes. Most of the requirements that were assigned with the priority of Must have were accomplished with the exception of two involving the statistics. These requirements were not met due to restrictions in the workflow engine that only later, with deeper knowledge of the system, were discovered. The following list shows the information regarding the requirements, their priority and if they were implemented or not.

User story	Priority	implemented
<b>Citizens</b>		
US#1	Nice to Have	No
US#2	Nice to Have	Yes
US#3	Nice to Have	No
US#4	Nice to Have	No
US#5	Nice to Have	No
US#6	Nice to Have	No
US#7	Nice to Have	No
<b>Admin</b>		
US#8	Must Have	Yes
US#9	Must Have	Yes
US#10	Should Have	Yes
US#11	Nice to Have	No

US#12	Must Have	Yes
US#13	Must Have	Yes
US#14	Must Have	No
US#15	Should Have	No
US#16	Should Have	Yes
US#17	Should Have	Yes
US#18	Should Have	Yes
<b>Worker</b>		
US#19	Must Have	Yes
US#20	Must Have	Yes
US#21	Must Have	Yes
US#22	Must Have	Yes
US#23	Must Have	Yes

Table 7.2: Implemented requirements

These results suggest that the system has most of the functionalities expected to have. As mentioned before, all of these must-have requirements have been successfully implemented except the ones that require metrics that ended-up with a version of what was expected to be that later can be improved depending on the plugin updates.

## Chapter 8

# Final Product

In this chapter, the product developed during this internship is going to be presented. The interfaces shown here are the final interface of the developed system. Here is going to be shown the process that came out of the software development process during the time of the internship. Most of the design and decisions were made in line with what was planned in the first half of the internship, as others involved some decision-making and making the most of the application for the end-user. To verify the normal function of the project and assess the system a dataset of open 311 requests was used. This dataset from the city of Bloomington, USA had to be changed to be used in the project since some data, like the user information, was not available. Other changes that were made to the dataset were because of the integration with the workflow engine and the need of associating an instance of the engine process with each occurrence. After this, the intern created a BPMN model with two automatic tasks and one user task that requires a user to conclude the task to verify the frontend application's correct behavior. This dataset allowed to verify the normal function of the system and produce evidence of what the system could accomplish and produce an output to show the final product.

The frontend application is composed of two pages that are accessible to the users described as Workers. The first page is a simple login page composed of an username field and a password field and a button that when clicked sends the information to the backend application. Once the information is verified, the backend application sends a JWT token to maintain a session for that user.

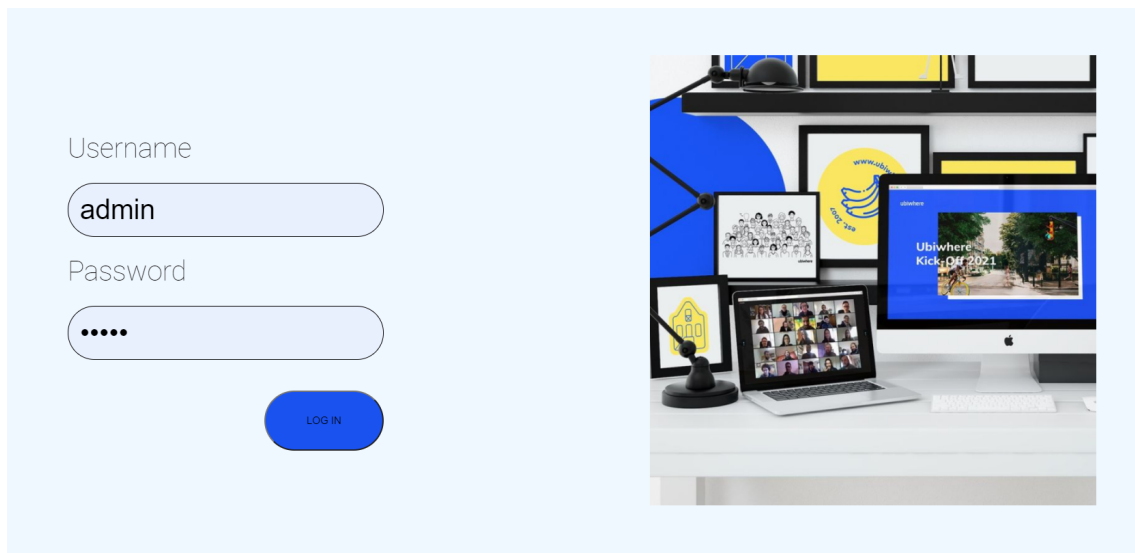


Figure 8.1: Log in page of the front end application

Once the user successfully logs in into the application, he is redirected to the Dashboard where the information present in the backend application is shown. Every task that exists in the system can be searched in the table with the title “Tasks” and the users can claim only the ones that are in the sidebar with the name “Tasks”. There is also present in this application a map pinpointing every occurrence. The green dots in the map mark the occurrences that can be claimed by workers and the red ones the ones that can not be claimed.

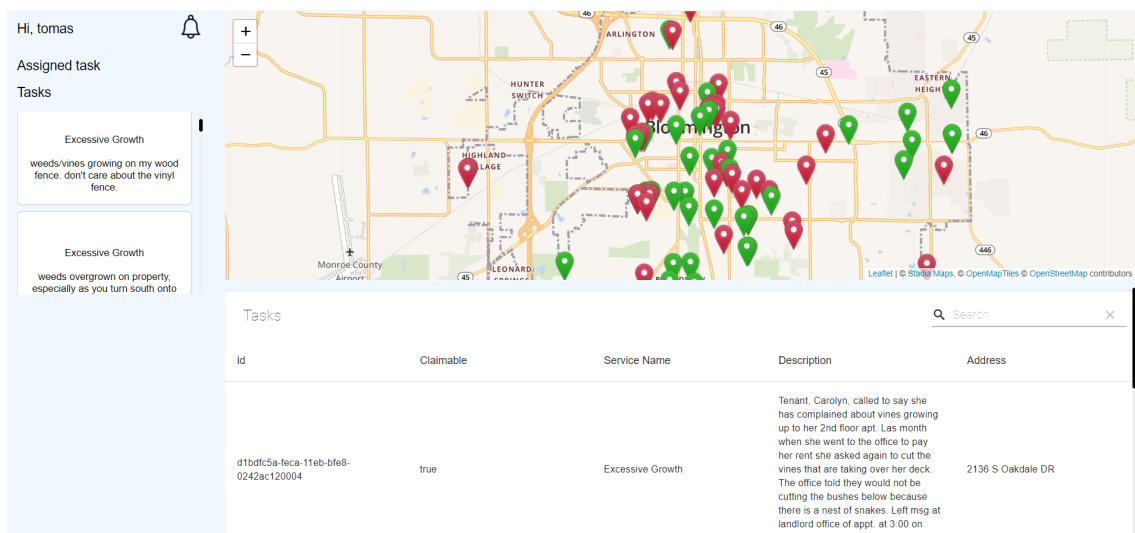


Figure 8.2: Dashboard of the front end application

When the user wants to claim a task, he selects a task in the left slider and a pop-up opens with information about the task and two buttons.



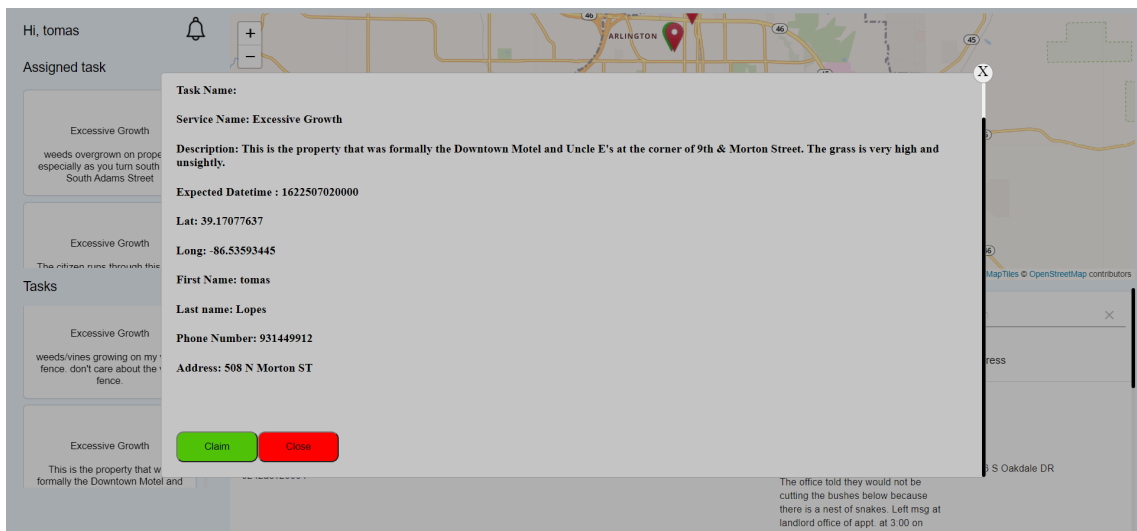


Figure 8.3: Details of the selected occurrence and possible actions

After claiming the task, the task will move to the assigned task making the application look like the following image.

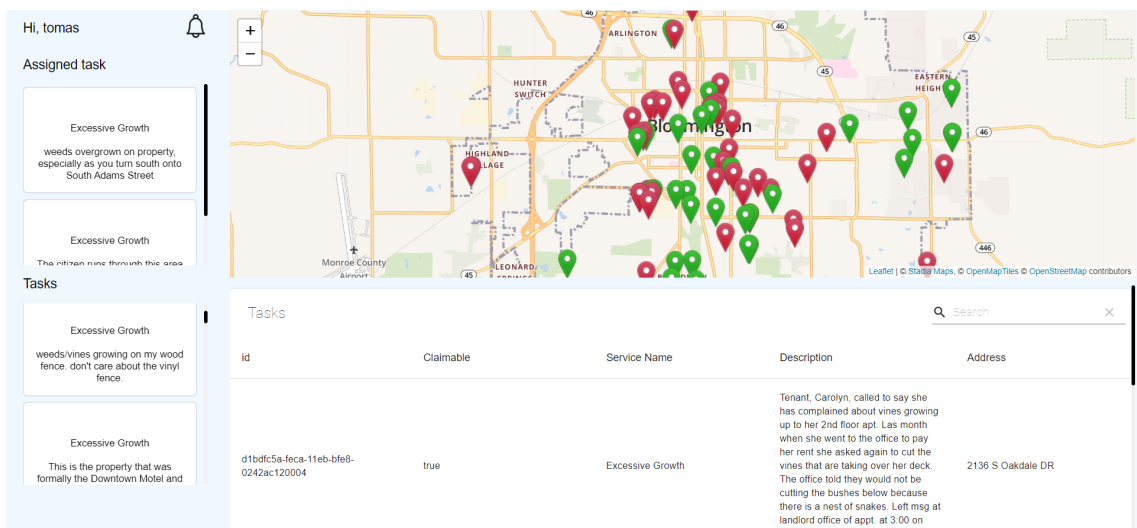


Figure 8.4: Dashboard of the front end application with tasks claimed

These actions generate notifications for the other users to inform and to create a log of claimed and completed tasks. For the user to read the notification, the user must pass in the notification bell and a section will appear where the user can select the notification.

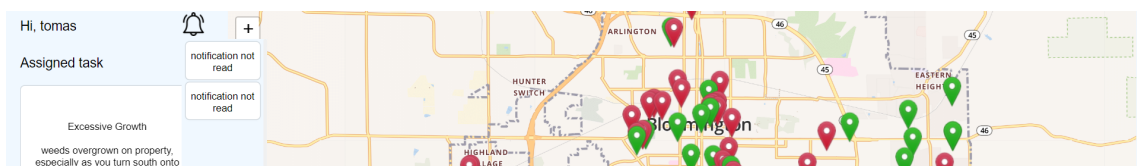


Figure 8.5: Notifications in the dashboard of the front end application

These notifications will not appear automatically when it is created because it was not a requirement of the system, however, when the user reloads the page it shows up.

Another major component of the system is the django admin application. This application allows for admin users to create and edit objects and interact with the workflow engine invisibly through signals that were previously discussed. This makes crucial integration with the workflow engine making it possible for these two components to communicate with each other. Since the integration is made through signals if later is decided to integrate a new application for admins through the REST API this would make the process of integration easier. Other useful functionalities present in this application are the import and export functionality that allows the user to import several objects to the application and export them in various formats.

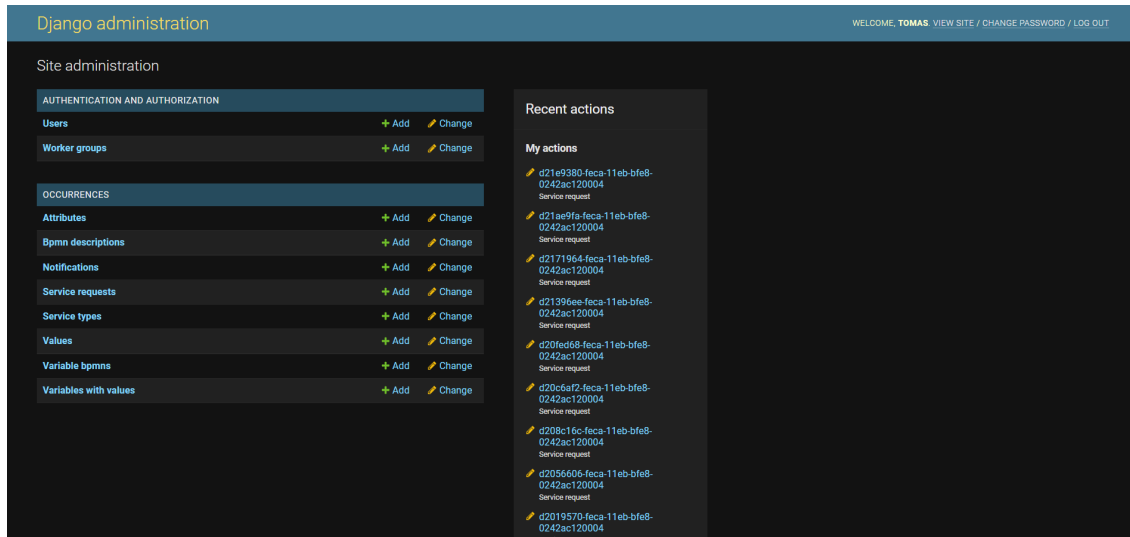


Figure 8.6: Django admin main page

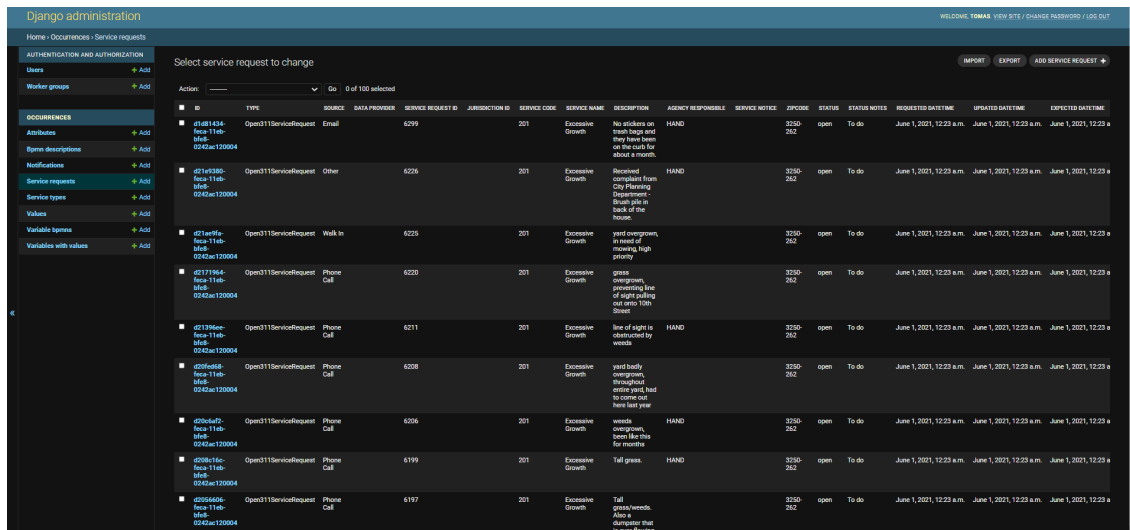


Figure 8.7: Occurrences represented in the Django admin

Another crucial part is the application to create the BPMN models used for the workflow engine, the Camunda modeler. Although not developed by the intern, without this application it would not be possible to describe the process required to handle the occurrences and the Camunda workflow engine does not work with any other BPMN modeler because of its proprietary activities, the steps of the model, which include a lot of information to

the engine, including the topics that allow to perform any kind of task and so brings value to the solution.

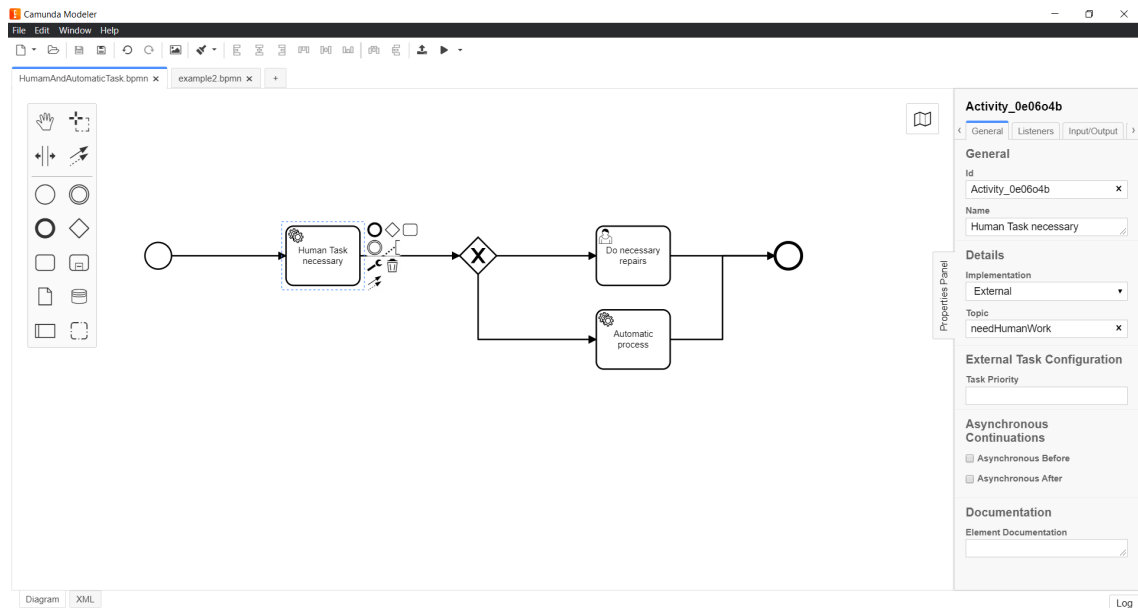


Figure 8.8: Creation of a model in the Camunda modeler

Although not a direct part of the system, a visual tool is also available for the workflow engine that helps debug and visualize the tasks in the workflow engine. This was very helpful in the developing process to make a double check of the requests made from the backend application.

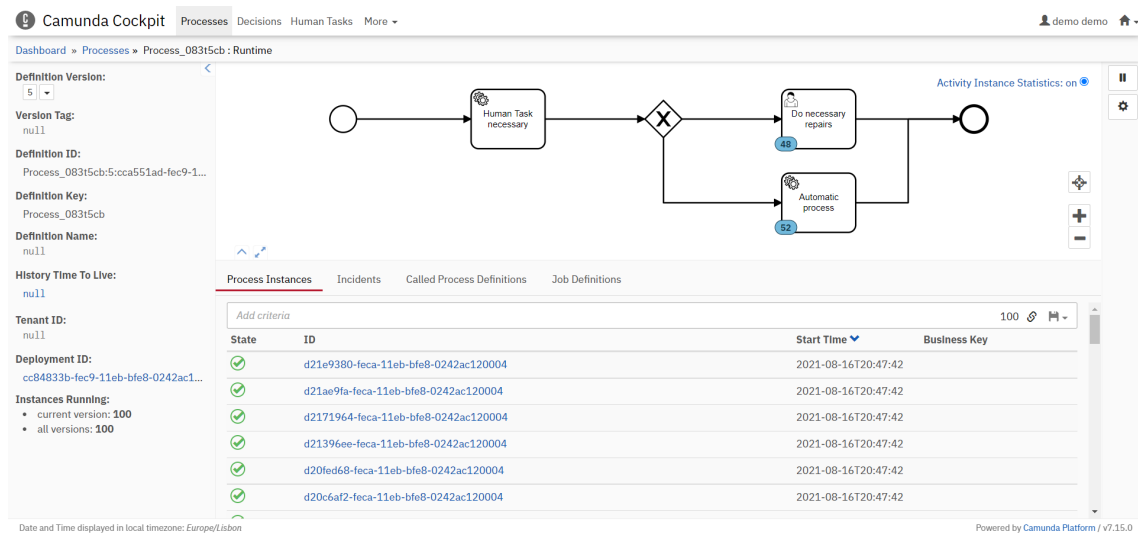


Figure 8.9: BPMN model represented in the workflow engine with several instances

This system has one component that was not initially thought of having or at least not in the way that it was implemented, the statistical part. The statistical part of the system that was initially thought to be easily integrated with the backend application making use of the REST API of the engine was an idea that after a deeper review to the situation it was noticed that it would not be possible to do in the way that was initially thought of doing. Since this would not be possible to make, but the engine met every other requirement, it

began the search for other solutions. The best solution found was a newly made plug in that met the requirement and had the potential of improving the application in the future. This plugin used Prometheus and Grafana to show some metrics from the workflow engine.



Figure 8.10: Grafana dashboard with multiple metrics

The metrics collect do not have granularity within each type of occurrence instead the collected metrics are of the aggregation of each type present in the workflow engine. The plugin also provides good integration with the workflow engine and later, after upgrades to the plugin, be redesigned to meet the requirement as it should. Other parts that made the use of the plugin even more logic was it could make the integration and deployment through Docker, which can later be scalable along with the application.

## Chapter 9

# Conclusion & Future Work

This chapter presents the conclusions taken by the intern during the internship about the process and the developed product. It is also mentioned some of the future work that can be done to improve the system that was created later in the future, and the new features that can be implemented.

### 9.1 Work done

Considering the success criteria defined for this internship, it can be considered that the internship was a success. Most of the requirements defined as must-have have been implemented successfully with the exception of the requirements involving the statistics from the workflow engine. These requirements were not implemented in the way that it was initially predicted to be, however, a solution to implement them partially has been done. This solution involved the use of more tools that the intern was not experienced with, and therefore increased the complexity of the project overall. Despite this, the integration with the workflow engine was a success and there is a lot of potential future work to do with the exploration of all of its features.

Unfortunately it was not possible to explore all of the features and all of the characteristics especially in the Camunda modeler which is a tool that can have a tremendous impact in the efficiency of the BPMN models due to the learning curve and complexity of the application. Unfortunately there was not enough time to develop the frontend application for users in the internship due to encountered setbacks and the complexity of the tools that were used. This was already foreseeable in the first semester and was agreed with the hosting company that the requirements regarding this part of the project would be stated as nice to have as the initial plan was probably too ambitious like mentioned in the Risk represented in the table 3.5 in the in chapter 3 of Planning&Methodologies .

It was necessary to use several technologies for both backend, frontend and even for infrastructure that the intern did not feel comfortable with, which increased the complexity of the internship. Integrating these different components was also not a simple assignment since the intern did not have experience with the tool used as infrastructure for this project. At the testing level, the intern also made a set of tests that can cover a large part of the backend application, and to verify the frontend application were used functional tests that verified its correct functioning.

The non-functional requirements affected the final product and in how it was developed.

The usability requirement was fulfilled by making sure that users could use the application in different screen sizes. This was tested by increasing and decreasing the application page and certified that the use of the application was not harder. The requirement of performance was verified in two specific scenarios that were more sensitive to the application that were the insertion of BPMN files, table 4.3, and multiple occurrences to the system, table 4.4, which were also tested in the functionality tests. In terms of maintainability the requirement was more abstract to the final users since it is a requirement focused on the process of development and future developers, but good practices such as standards of RESTful architectures, standards to represent the models (OpenAPI) and documentation for the REST API were followed to verify that this would be satisfied. The security that the system must provide which ensure that users without credentials or without the clearance could not access some endpoints, like *GET /notification/* and *POST /bpmn\_description/* once they require admin clearance, was accomplished by the use of JWT.

The final product is a proof of concept of what can be achieved with the integration of these technologies. This internship can provide valuable information for future products as well as a base to implement a product similar to the one presented.

## 9.2 Challenges

While developing the system the intern faced some challenges since many of the technologies used were not known by the intern before the internship. The decisions made throughout the internship were made always thinking of what could bring more value to the project and also in the future work that was possible to do. These challenges caused the intern to delay the internship due to the increase of complexity.

One challenge that probably impacted the project the most was the research made to the workflow engine. The intern had some difficulties deciding to what degree of granularity should he stop the research of the engine and the adjacent software such as the camunda modeler and the javascript workers and focus on the integration of the systems since the observed potential was tremendous and not so trivial.

The challenges present with the chosen technologies were due to the intern inexperience with them and the learning curve associated with them. Another big challenge posed was the way that the metrics were collected from the workflow engine since the first objective had little to none similarities with what was developed. To overpose this situation the intern had to resort to a new tool with little documentation that forced the intern to spend more time in the learning process to implement it as well as new technologies, the Prometheus and the Grafana that the intern never had used as well. Another challenge related to the technologies used was the poor documentation of the tools like is the case of the Camunda modeler and the Plugin for the metrics. Although many of the tools are used by a lot of users and have great communities and documentation, the documentation required to integrate the different pieces of the system was many times too specific for the use case which forced the intern to spend more time to set up the environment necessary for all of them to work as a whole.

## 9.3 Future Work

The work done in this internship was what was expected to be. All of the must-have requirements have been completed or partly complete, and the application has the potential

to grow thanks to the REST API and the fact that it is ready to be deployed in docker.

The application is ready for other systems to be added into it, in specific the Citizen application that was previously mentioned in the document and was considered not essential to this internship due to being possible to obtain occurrences from other sources, but in the future can be an application to implement and integrate into the system as a new application or even in the workers application that was created creating new pages that were accessible depending on the user permissions. Other future work that can be foreseen is the integration of the application in Kubernetes. This would make it possible to deploy multiple instances of the system and coordinate them. The system can also suffer a visual upgrade in the front-end application made for workers since the application was intended to work as proof of concept and the visual aspect of it was not with the most importance.

## **9.4 Lessons Learned**

Plenty of lessons were learned in this internship, but one of the biggest lessons learned was that it is impossible to control every single aspect of the development moreover when working with third parties applications. However that preview statement is a reality, it is always possible to minimize it by studying and planning what is going to be made and sometimes addressing someone with more experience that has passed for many of the mistakes that inevitably will be made. More important than this is the capability to resolve these problems and address these issues.

Thankfully, the team at Ubiwhere helped to overcome these situations providing help to make decisions and provide a healthy environment for the intern to make mistakes and learn from them, and in the end assist when necessary. This internship forced the intern to learn new skills and granted him with experience on software development never before achieved prior to this internship. In order to keep up with the internship expectations the intern had to improve his capabilities in software development. All of this had a huge impact in the intern experience in multiple technologies that are among the most popular in the market, which improved the intern's skill set, enriching his future in the area.

# Bibliography

- [1] S. City, “Smart City,” *The Smart City Journal*, 17-Mar-2017. [Online]. <https://www.thesmartcityjournal.com/en/articles/1333-smart-cities-futuristic-vision>. [Accessed: Nov-2020].
- [2] M. Gorini, “What Exactly is a Smart City?,” *Blog de Bismart, Informação sobre Big Data, Artificial Intelligence, BI*. [Online]. Available: <https://blog.bismart.com/en/what-is-a-smart-city>. [Accessed: Nov-2020].
- [3] View of A Theory of Smart Cities. [Online]. Available: <https://journals.issn.org/index.php/proceedings55th/article/view/1703/572>. [Accessed: Nov-2020].
- [4] “Difference between Information and Data,” *Guru99*. [Online]. Available: <https://www.guru99.com/difference-information-data.html>. [Accessed: Nov-2020].
- [5] Santana, E. et al. “Software Platforms for Smart Cities.” *ACM Computing Surveys (CSUR)* 50 (2018): 1 - 37.
- [6] K. Kolomvatsos and C. Anagnostopoulos, “Reinforcement Learning for Predictive Analytics in Smart Cities,” *Informatics*, vol. 4, no. 3, p. 16, Jun. 2017.
- [7] Ferreira M., Ramos J., Novais P. (2019) Occurrences Management in a Smart-City Context. In: Rodríguez S. et al. (eds) *Distributed Computing and Artificial Intelligence, Special Sessions, 15th International Conference. DCAI 2018. Advances in Intelligent Systems and Computing*, vol 801. Springer, Cham. [https://doi.org/10.1007/978-3-319-99608-0\\_13](https://doi.org/10.1007/978-3-319-99608-0_13)
- [8] Eco, “Lisboa vai ter plataforma para gestão de ocorrências,” *ECO*, 14-Jul-2017. [Online]. Available: <https://eco.sapo.pt/2017/07/14/lisboa-vai-ter-plataforma-para-gestao-de-ocorrencias/>. [Accessed: Nov-2020].
- [9] S. P. Lopes, “Lisbon smart cities : perception and reality,” *RUN*, 18-Jun-2018. [Online]. Available: <https://run.unl.pt/handle/10362/40252>. [Accessed: Nov-2020].
- [10] “Iniciativas,” *Lisboa Inteligente*, 26-Oct-2020. [Online]. Available: <https://lisboainteligente.cm-lisboa.pt/iniciativas/>. [Accessed: Nov-2020].
- [11] “Na Minha Rua LX,” *Lisboa Inteligente*, 03-Feb-2020. [Online]. Available: <https://lisboainteligente.cm-lisboa.pt/lxi-iniciativas/na-minha-rua-lx/>. [Accessed: Nov-2020].
- [12] B. S. City, “Smart City Lisbon: Knowing more: Brussels Smart City,” *Knowing more | Brussels Smart City*. [Online]. Available: <https://smartcity.brussels/news-598-smart-city-lisbon>. [Accessed: Nov-2020].



- 
- [13] “Lisboa é uma das Smart Cities da UE que vão guiar a Europa,” UE – Lisboa é uma das Smart Cities que vai guiar a Europa. [Online]. Available: <http://www.lisbonne-idee.pt/p4132-lisboa-uma-das-smart-cities-que-vaoguiar-europa.html>. [Accessed: Nov-2020].
- [14] Pixelixir, “Lisbon,” SHARING CITIES. [Online]. Available: <http://www.sharingcities.eu/sharingcities/city-profiles/lisbon>. [Accessed: Nov-2020].
- [15] J. F. Gomes, “Smart Cities. O que Lisboa tem para ensinar (e a aprender),” Observador, 26-Jun-2017. [Online]. Available: <https://observador.pt/especiais/smart-cities-o-que-lisboa-tem-a-ensinar-e-a-aprender/>. [Accessed: Nov-2020].
- [16] O. A. S. C. Secretariat, “Lisbon’s Bet on Urban Data Platform,” Open & Agile Smart Cities. [Online]. Available: <https://oascities.org/lisbons-bet-on-urban-data-platform/>. [Accessed: Nov-2020].
- [17] “Intelligent Management Platform makes the city of Lisbon smarter,” NEC. [Online]. Available: <https://www.nec.com/en/global/onlinetv/en/lisbon.html>. [Accessed: Nov-2020].
- [18] “About Us: INCT,” Interscity. [Online]. Available: <https://interscity.org/about/>. [Accessed: Nov-2020].
- [19] Fabio Kon, Kelly Braghetto, Eduardo Z. Santana, Roberto Speicys, and Jorge Guerra Guerra. 2020. Toward smart and sustainable cities. *Commun. ACM* 63, 11 (November 2020), 51–52. DOI:<https://doi.org/10.1145/3416961>
- [20] D. M. Batista, A. Goldman, R. Hirata, F. Kon, F. M. Costa and M. Endler, "InterSCity: Addressing Future Internet research challenges for Smart Cities," 2016 7th International Conference on the Network of the Future (NOF), Buzios, 2016, pp. 1-6, doi: 10.1109/NOF.2016.7810114.
- [21] Bertrand-Martinez, Eddas & Feio, Phelipe & Nascimento, Vagner & Kon, Fabio & Abelém, Antônio. (2020). Classification and evaluation of IoT brokers: A methodology. *International Journal of Network Management*. 10.1002/nem.2115.
- [22] “InterSCity Platform: INCT,” Interscity. [Online]. Available: <https://interscity.org/software/interscity-platform/>. [Accessed: Nov-2020].
- [23] “Relatório Câmara Municipal de Lisboa.” [Online]. Available: <https://www.am-lisboa.pt/documentos/1529934530I0wDR8zb1Yy52HT0.pdf>. [Accessed: Nov-2020].
- [24] “FixMyStreet: Platform,” Welcome | FixMyStreet Platform | mySociety. [Online]. Available: <https://fixmystreet.org/>. [Accessed: Nov-2020].
- [25] “FixMyStreet: Platform,” How to FixMyStreet | FixMyStreet Platform | mySociety. [Online]. Available: <https://fixmystreet.org/how-it-works/>. [Accessed: Nov-2020].
- [26] “Can we fix it? .” [Online]. Available: <https://fixmystreet.org/The-FixMyStreet-Platform-DIY-Guide-v1.1.pdf>.
- [27] A. R. and E. N. (U. of S. Peter Matthews (University of Stirling), “FixMyStreet!,” mySociety. [Online]. Available: <https://research.mysociety.org/html/fms-report/>. [Accessed: Nov-2020].

- [28] Myfanwy, “Councils: Use Open311 and youll never have to re-key a problem report from FixMyStreet. And its free.,” mySociety, 09-Sep-2013. [Online]. Available: <https://www.mysociety.org/2013/09/09/councils-use-open311-and-youll-never-have-to-re-key-a-problem-report-from-fixmystreet-and-its-free/>. [Accessed: Nov-2020].
- [29] D. Whiteland, “Open311: Explained,” mySociety, 08-Jun-2016. [Online]. Available: <https://www.mysociety.org/2013/01/17/open311-explained/>. [Accessed: Nov-2020].
- [30] SF311, “Open311 Applications,” SF311, 05-Mar-2019. [Online]. Available: <https://sf311.org/help/open311-applications>. [Accessed: Nov-2020].
- [31] “Citizen Mobile Apps Build Community - SeeClickFix,” SeeClickFix.com. [Online]. Available: <https://seeclickfix.com/pages/311-app.html>. [Accessed: Nov-2020].
- [32] C. Vein, “5 Open Data Apps That Are Improving Our Cities,” Mashable, 13-Aug-2010. [Online]. Available: <https://mashable.com/2010/08/13/open311-apps/?europe=true>. [Accessed: Nov-2020].
- [33] “311 Service Requests,” Analyze Boston. [Online]. Available: <https://data.boston.gov/dataset/311-service-requests>. [Accessed: Nov-2020].
- [34] T. Steinberg, “Open311 - What is it, and why is it good news for both governments and citizens?,” mySociety, 08-Jun-2016. [Online]. Available: <https://www.mysociety.org/2013/01/10/open311-introduced/>. [Accessed: Nov-2020].
- [35] “Payments for Utilities Bill,” Article · 311 Service Portal. [Online]. Available: <https://311.southbendin.gov/knowledgecenter/article/?id=KA-04633>. [Accessed: Nov-2020].
- [36] “HOW CAN WE HELP YOU?,” Home · NYC311. [Online]. Available: <https://portal.311.nyc.gov/>. [Accessed: Nov-2020].
- [37] “Talk About an Information Highway – Open 511 meets DriveBC,” TranBC, 16-Nov-2018. [Online]. Available: <https://www.tranbc.ca/2015/09/17/talk-about-an-information-highway-open-511-meets-drivebc/>. [Accessed: Nov-2020].
- [38] Rindle, “Which Workflow Type Works for You?,” rindle. [Online]. Available: <https://rindle.com/blog/which-workflow-type-works-for-you/>. [Accessed: Dec-2020].
- [39] A. Kornev, “The concept of workflow engines,” Medium, 18-Mar-2020. [Online]. Available: <https://itnext.io/the-concept-of-workflow-engines-c14e8088283?gi=a4c075a042d>. [Accessed: Dec-2020].
- [40] Meirwah, “meirwah/awesome-workflow-engines,” GitHub. [Online]. Available: <https://github.com/meirwah/awesome-workflow-engines>. [Accessed: Dec-2020].
- [41] Bonitasoft, 01-Jan-1970. [Online]. Available: <https://www.bonitasoft.com/>. [Accessed: Dec-2020].
- [42] Elsa-Workflows, “elsa-workflows/elsa-core,” GitHub. [Online]. Available: <https://github.com/elsa-workflows/elsa-core>. [Accessed: Dec-2020].
- [43] “Início,” Lisboa Inteligente, 23-Oct-2020. [Online]. Available: <https://lisboainteligente.cm-lisboa.pt/>. [Accessed: Dec-202AD].

- 
- [44] S. C. O. O. P. S. GmbH, "Workflow made easy," COPPER. [Online]. Available: <https://copper-engine.org/>. [Accessed: Dec-2020].
- [45] R. Soika, "Workflow: Open Source Workflow Engine," Imixs. [Online]. Available: <https://www.imixs.org/>. [Accessed: Dec-2020].
- [46] "jBPM - Open Source Business Automation Toolkit - jBPM ..." [Online]. Available: <https://www.jbpm.org/>. [Accessed: Dec-2020].
- [47] Vito Albino, Umberto Berardi & Rosa Maria Dangelico (2015) Smart Cities: Definitions, Dimensions, Performance, and Initiatives, *Journal of Urban Technology*, 22:1, 3-21, DOI: 10.1080/10630732.2014.942092
- [48] T. Bakıcı, E. Almirall, and J. Wareham, "A Smart City Initiative: the Case of Barcelona," *Journal of the Knowledge Economy*. [Online]. Available: [https://econpapers.repec.org/article/sprjknowl/v\\_3a4\\_3ay\\_3a2013\\_3ai\\_3a2\\_3ap\\_3a135-148.htm](https://econpapers.repec.org/article/sprjknowl/v_3a4_3ay_3a2013_3ai_3a2_3ap_3a135-148.htm). [Accessed: Dec-2020].
- [49] Hall, R.E. (2000) The Vision of a Smart City. 2nd International Life Extension Technology Workshop, Paris, France, September.
- [50] Tim Marshall (2000) Urban Planning and Governance: Is there a Barcelona Model?, *International Planning Studies*, 5:3, 299-319, DOI: 10.1080/713672855
- [51] Bakıcı, T., Almirall, E. & Wareham, J. A Smart City Initiative: the Case of Barcelona. *J Knowl Econ* 4, 135–148 (2013). <https://doi.org/10.1007/s13132-012-0084-9>
- [52] Z. G. I. of Technology, "Barcelona Smart City: most remarkable Example of Implementation," *Engineers & Architects*, 03-Nov-2020. [Online]. Available: <https://www.e-zigurat.com/blog/en/smart-city-barcelona-experience/>. [Accessed: Dec-2020].
- [53] J.-R. Ferrer, "Barcelonas Smart City vision: an opportunity for transformation," *Field Actions Science Reports*. The journal of field actions, 01-Jun-2017. [Online]. Available: <https://journals.openedition.org/factsreports/4367>. [Accessed: Dec-2020].
- [54] L. Smith, "Smart City Portrait: Barcelona," *The Global Smart City Knowledge Center*, 24-Nov-2020. [Online]. Available: <https://hub.beesmart.city/city-portraits/smart-city-portrait-barcelona>. [Accessed: Dec-2020].
- [55] S. Ravindra, About Savaram Ravindra Savaram Ravindra was born and raised in Hyderabad, N. says, S. Clein, J. Burrows, V. Lindner, and R. Osborne, "The Technologies That Turned Barcelona Into A Smart City," *Barcinno*, 05-Jul-2018. [Online]. Available: <http://www.barcinno.com/barcelona-smart-city-technologies/>. [Accessed: Dec-2020].
- [56] Í. França et al., "SIGOc: A Smart Campus Platform to Improve Public Safety," 2020 IEEE 6th World Forum on Internet of Things (WF-IoT), New Orleans, LA, USA, 2020, pp. 1-6, doi: 10.1109/WF-IoT48130.2020.9221373.
- [57] Apache, "apache/airflow," GitHub. [Online]. Available: <https://github.com/apache/airflow>. [Accessed: Dec-2020].
- [58] "Apache Airflow Home Page," Apache Airflow. [Online]. Available: <https://airflow.apache.org/>. [Accessed: Dec-2020].

- [59] “Experimental REST API Reference,” Experimental REST API Reference - Airflow Documentation. [Online]. Available: <https://airflow.apache.org/docs/apache-airflow/stable/rest-api-ref.html>. [Accessed: Dec-2020].
- [60] “Ecosystem,” Apache Airflow. [Online]. Available: <https://airflow.apache.org/ecosystem/>. [Accessed: Dec-2020].
- [61] GARCÉS, Ricardo, et al. Open source workflow management systems: a concise survey. GWDL: A graphical workflow definition language for business workflows. Springer Berlin Heidelberg, 2009.
- [62] jBPM Documentation [Online]. Available: [https://docs.jboss.org/jbpm/release/7.21.0.Final/jbpm-docs/html\\_single/#\\_jbpmoverview](https://docs.jboss.org/jbpm/release/7.21.0.Final/jbpm-docs/html_single/#_jbpmoverview) [Accessed: Dec-2020].
- [63] Kogito Documentation [Online]. Available: [https://docs.jboss.org/kogito/release/latest/html\\_single/#con-kogito-automation\\_kogito-docs](https://docs.jboss.org/kogito/release/latest/html_single/#con-kogito-automation_kogito-docs) [Accessed: Dec-2020].
- [64] camunda B. P. M. community, Introduction. [Online]. Available: <https://docs.camunda.org/manual/7.14/introduction/>. [Accessed: Dec-2020].
- [65] camunda B. P. M. community, Implemented Standards. [Online]. Available: <https://docs.camunda.org/manual/7.14/introduction/implemented-standards/>. [Accessed: Dec-2020].
- [66] “What is Case Management Model and Notation (CMMN),” What is Case Management Model and Notation (CMMN)? [Online]. Available: <https://www.visual-paradigm.com/guide/cmmn/what-is-cmmn/>. [Accessed: Dec-2020].
- [67] Delgado, Andrea & Calegari, Daniel. (2019). Towards integrating BPMN 2.0 with CMMN and DMN standards for flexible business process modeling.
- [68] K. Bahri, “Activiti Overview,” Medium, 20-Jan-2020. [Online]. Available: <https://medium.com/@kanavbahri/activiti-overview-e6d2754687c1>. [Accessed: Dec-2020].
- [69] “K2 Platform Overview,” Techendo, 23-Jan-2019. [Online]. Available: <https://techendo.com/post/k2-platform-overview.html>. [Accessed: Dec-2020].
- [70] “Business Process Automation Platform: K2 Process Automation,” Nintex. [Online]. Available: <https://www.nintex.com/workflow-automation/k2-software/>. [Accessed: Dec-2020].
- [71] “What is Zeebe?,” Zeebe. [Online]. Available: <https://zeebe.io/what-is-zeebe/>. [Accessed: Dec-2020].
- [72] “Zeebe Engine,” Camunda Cloud Docs. [Online]. Available: <https://docs.zeebe.io/basics/exporters.html>. [Accessed: Dec-2020].
- [73] N. Werner, “Writing User Stories With Gherkin,” Medium, 11-Oct-2017. [Online]. Available: <https://medium.com/@nic/writing-user-stories-with-gherkin-dda63461b1d2>. [Accessed: Dec-2020].
- [74] “Agile Methodology: What is Agile Software Development Model?,” Guru99. [Online]. Available: <https://www.guru99.com/agile-scrum-extreme-testing.html>. [Accessed: Dec-2020].

- 
- [75] The C4 model for visualising software architecture. [Online]. Available: <https://c4model.com/>. [Accessed: Dec-2020].
- [76] “What are Standards?,” IRENA International Renewable Energy Agency. [Online]. Available: <https://www.irena.org/inspire/Standards/What-are-Standards>. [Accessed: Jan-2021].
- [77] S. Macbeth, “Citizen Engagement,” Citizen Engagement | Participatory Methods. [Online]. Available: <https://www.participatorymethods.org/glossary/citizen-engagement>. [Accessed: Jan-2021].
- [78] Smart-Data-Models, “smart-data-models/data-models,” GitHub. [Online]. Available: <https://github.com/smart-data-models/data-models>. [Accessed: Jan-2021].
- [79] “Civic Issue Tracking data models,” Fiware. [Online]. Available: <https://fiware-datamodels.readthedocs.io/en/latest/IssueTracking/doc/introduction/index.html>. [Accessed: Jan-2021].
- [80] “Open 311 Service Type,” Fiware. [Online]. Available: [https://fiware-datamodels.readthedocs.io/en/latest/IssueTracking/Open311\\_ServiceType/doc/spec/index.html](https://fiware-datamodels.readthedocs.io/en/latest/IssueTracking/Open311_ServiceType/doc/spec/index.html). [Accessed: Jan-2021].
- [81] “Open 311 Service Request,” Fiware. [Online]. Available: [https://fiware-datamodels.readthedocs.io/en/latest/IssueTracking/Open311\\_ServiceRequest/doc/spec/index.html](https://fiware-datamodels.readthedocs.io/en/latest/IssueTracking/Open311_ServiceRequest/doc/spec/index.html). [Accessed: Jan-2021].
- [82] Driving Directions, Traffic Reports & Carpool Rideshares by Waze. [Online]. Available: <https://www.waze.com/en/waze>. [Accessed: Jan-2021].
- [83] “Supersonic Subatomic Java,” Quarkus. [Online]. Available: <https://quarkus.io/>. [Accessed: Jan-2021].
- [84] Knative, 17-Dec-2020. [Online]. Available: <https://knative.dev/>. [Accessed: Jan-2021].
- [85] Apache Kafka. [Online]. Available: <https://kafka.apache.org/>. [Accessed: Jan-2021].
- [86] “Kogito ergo automate,” Kogito. [Online]. Available: <https://kogito.kie.org/>. [Accessed: Jan-2021].
- [87] “Workflow and Decision Automation Platform,” Camunda, 08-Jan-2021. [Online]. Available: <https://camunda.com/>. [Accessed: Jan-2021].
- [88] “Open Source Business Automation,” Activiti. [Online]. Available: <https://www.activiti.org/>. [Accessed: Jan-2021].
- [89] Editor, “Acceptance Criteria: Purposes, Formats, and Best Practices,” AltexSoft, 09-Dec-2019. [Online]. Available: <https://www.altexsoft.com/blog/business/acceptance-criteria-purposes-formats-and-best-practices/>. [Accessed: Jan-2021].
- [90] A. Qian, “Advantages and Disadvantages of Microservices Architecture,” Cloud Academy, 21-May-2020. [Online]. Available: <https://cloudacademy.com/blog/microservices-architecture-challenge-advantage-drawback/>. [Accessed: Jan-2021].

- [91] "Top 10 Smart Cities in the World - ASME", Asme.org, 2021. [Online]. Available: <https://www.asme.org/topics-resources/content/top-10-growing-smart-cities>. [Accessed: Jun - 2021].
- [92] "Home," Gov.sg. [Online]. Available: <https://www.smartnation.gov.sg/>. [Accessed: Jun -2021].
- [93] StephenOTT, "StephenOTT/Camunda-Monitoring: Monitor anything you can query in a Camunda BPM engine. Monitor KPIs, metrics, and any business performance," GitHub. [Online]. Available: <https://github.com/StephenOTT/Camunda-Monitoring>. [Accessed: Jun-2021].
- [94] Prometheus, "Prometheus - Monitoring system & time series database," Prometheus.io. [Online]. Available: <https://prometheus.io/>. [Accessed: Sep-2021]
- [95] Grafana.com. [Online]. Available: <https://grafana.com/>. [Accessed: 04-Sep-2021].
- [96] K. Desrosiers, "How Docker can improve your development environments," Phytochemia Tech Blog, 04-Oct-2017. [Online]. Available: <https://medium.com/phytochemia-tech-blog/how-docker-can-improve-your-development-environments-731cdfda0b9a>. [Accessed: 04-Sep-2021].
- [97] R. Bauer, "Docker containers vs. VMs: Pros and cons of containers and virtual machines," Backblaze.com, 28-Jun-2018. [Online]. Available: <https://www.backblaze.com/blog/vm-vs-containers/>. [Accessed: 04-Sep-2021].
- [98] T. Olzak, J. Boomer, R. M. Keefer, and J. Sabovik, "What is virtualization?," in Microsoft Virtualization, Elsevier, 2010, pp. 1–12.
- [99] "Virtualization via Containers," Cmu.edu. [Online]. Available: <https://insights.sei.cmu.edu/blog/virtualization-via-containers/>. [Accessed: 04-Sep-2021].
- [100] "Dockerfile reference," Docker.com, 02-Sep-2021. [Online]. Available: <https://docs.docker.com/engine/reference/builder/>. [Accessed: 04-Sep-2021].
- [101] "Overview of Docker Compose," Docker.com, 02-Sep-2021. [Online]. Available: <https://docs.docker.com/compose>. [Accessed: 04-Sep-2021].
- [102] "What is a workflow engine?," Processmaker.com, 13-Apr-2021. [Online]. Available: <https://www.processmaker.com/blog/what-is-a-workflow-engine/>. [Accessed: 04-Sep-2021].
- [103] Flyaps, "What are 'Front-end' & 'back-end,'" Flyaps.com, 30-Jul-2020. [Online]. Available: <https://flyaps.com/blog/difference-front-end-back-end-development-in-different-applications/>. [Accessed: 04-Sep-2021].
- [104] N. Pandit, "What and why React.Js," C-sharpcorner.com. [Online]. Available: <https://www.c-sharpcorner.com/article/what-and-why-reactjs/>. [Accessed: 04-Sep-2021].
- [105] T. Hamilton, "What is software testing? Definition, basics & types in Software Engineering," Guru99.com, 02-Jan-2020. [Online]. Available: <https://www.guru99.com/software-testing-introduction-importance.html>. [Accessed: 04-Sep-2021].

- 
- [106] Softwaretestinghelp.com. [Online]. Available: <https://www.softwaretestinghelp.com/guide-to-functional-testing/>. [Accessed: 04-Sep-2021].
- [107] L. Smith, "Amsterdam smart city: A world leader in smart city development," Beesmart.city. [Online]. Available: <https://hub.beesmart.city/city-portraits/smart-city-portrait-amsterdam>. [Accessed: 04-Sep-2021].
- [108] "Reporting issues in public space," Amsterdam.nl. [Online]. Available: <https://algoritmeregister.amsterdam.nl/en/reporting-issues-in-public-space/>. [Accessed: 04-Sep-2021].
- [109] "city of Bloomington data portal," Bloomington.in.gov. [Online]. Available: <https://data.bloomington.in.gov/>. [Accessed: 04-Sep-2021].
- [110] J. Kanjilal, "Pros and cons of monolithic vs. microservices architecture," Techtargget.com, 24-Jan-2020. [Online]. Available: <https://searchapparchitecture.techtargget.com/tip/Pros-and-cons-of-monolithic-vs-microservices-architecture>. [Accessed: 04-Sep-2021].
- [111] J. Reis, "Angular vs React: which framework to pick?," Imaginarycloud.com, 06-Feb-2020. [Online]. Available: <https://www.imaginarycloud.com/blog/angular-vs-react/>. [Accessed: 04-Sep-2021].
- [112] "Use volumes," Docker.com, 02-Sep-2021. [Online]. Available: <https://docs.docker.com/storage/volumes/>. [Accessed: 04-Sep-2021].
- [113] "Open Plans," Openplans.org. [Online]. Available: <https://www.openplans.org/>. [Accessed: 04-Sep-2021].
- [114] T. Hamilton, "What is functional testing? Types & examples (complete tutorial)," Guru99.com, 04-Apr-2020. [Online]. Available: <https://www.guru99.com/functional-testing.html>. [Accessed: 06-Sep-2021].
- [115] "API Documentation & Design Tools for Teams," Swagger.io. [Online]. Available: <https://swagger.io/>. [Accessed: 01-Sep-2021].
- [116] "ApparkB, un any facilitant l'aparcament," Barcelona.cat. [Online]. Available: [https://ajuntament.barcelona.cat/guardiaurbana/ca/noticia/apparkb-un-any-facilitant-laparcament\\_94837](https://ajuntament.barcelona.cat/guardiaurbana/ca/noticia/apparkb-un-any-facilitant-laparcament_94837). [Accessed: 07-Sep-2021].

This page is intentionally left blank.





# Chapter 10

## Annexes

### 10.1 Development

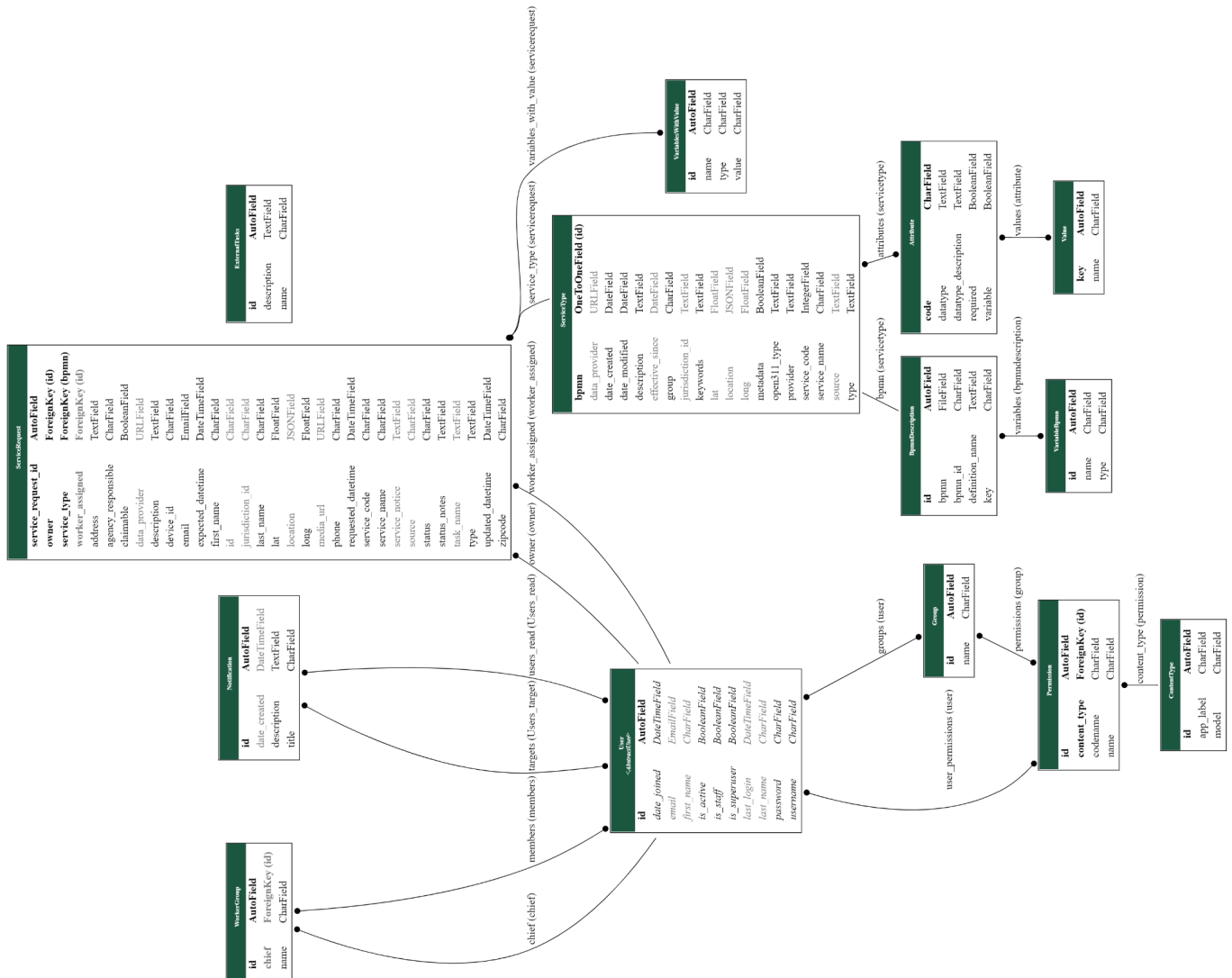


Figure 10.1: Detailed relationships between the models of the back end application

## 10.2 Tests

### 10.2.1 Unit Testing

ID	UN01
Title	request to create Service type object without credentials
Expected Output	An error response is shown
Pass/fail	Pass

ID	UN02
Title	request to create Service type object with the wrong credentials
Expected Output	An error response is shown
Pass/fail	Pass

ID	UN03
Title	request to create Service type object with the right credentials
Expected Output	a new service type is created
Pass/fail	Pass

ID	UN05
Title	request to list Service type object with wrong credentials
Expected Output	An error response is shown
Pass/fail	Pass

ID	UN06
Title	request to list Service type object with right credentials
Expected Output	A list with service types is shown
Pass/fail	Pass

ID	UN07
Title	request to create a variable BPMN object variable without credentials
Expected Output	An error response is shown
Pass/fail	Pass

ID	UN08
Title	request to create a variable BPMN variable object with wrong credentials
Expected Output	An error response is shown
Pass/fail	Pass

ID	UN04
Title	request to list Service type object without credentials
Expected Output	An error response is shown
Pass/fail	Pass

ID	UN09
Title	request to create a variable BPMN object with the right credentials
Expected Output	A variable BPMN object is created
Pass/fail	Pass

ID	UN10
Title	request to create a new BPMN description object without credentials
Expected Output	An error response is shown
Pass/fail	Pass

ID	UN11
Title	request to create a new BPMN description object with the wrong credentials
Expected Output	An error response is shown
Pass/fail	Pass

ID	UN12
Title	request to create a new BPMN description object with the wrong credentials
Expected Output	An error response is shown
Pass/fail	Pass

ID	UN13
Title	request to create a new BPMN description object with the right credentials
Expected Output	A new BPMN description object is created
Pass/fail	Pass

ID	UN14
Title	request to list all of the occurrences
Expected Output	An error response is shown
Pass/fail	Pass

ID	UN15
Title	request to get all of the assigned tasks to one user without credentials
Expected Output	An error response is shown
Pass/fail	Pass

ID	UN16
Title	request to get all of the assigned tasks to one user with wrong credentials
Expected Output	An error response is shown
Pass/fail	Pass

ID	UN17
Title	request to get all of the assigned tasks to one user with the right credentials
Expected Output	A list of occurrences that is assigned to the user is sended
Pass/fail	Pass

ID	UN18
Title	request to claim an occurrence without credentials
Expected Output	Is shown an error response
Pass/fail	Pass

ID	UN19
Title	request to get all of the assigned tasks to one user without credentials
Expected Output	Is shown an error response
Pass/fail	Pass

ID	UN20
Title	request to get all of the assigned tasks to one user with the wrong credentials
Expected Output	Is shown an error response
Pass/fail	Pass

ID	UN21
Title	request to get all of the assigned tasks to one user with the right credentials
Expected Output	Is returned a list of occurrences that are assigned to a user
Pass/fail	Pass

ID	UN22
Title	request to complete a task without credentials
Expected Output	Is shown an error response
Pass/fail	Pass

ID	UN23
Title	request to complete a task with the wrong credentials
Expected Output	Is shown an error response
Pass/fail	Pass

ID	UN24
Title	request to complete a task with the right credentials
Expected Output	An occurrence is complete
Pass/fail	Pass

ID	UN25
Title	request to get all notifications to a certain user without credentials
Expected Output	Is shown an error response
Pass/fail	Pass

ID	UN26
Title	request to get all notifications to a certain user with the wrong credentials
Expected Output	Is shown an error response
Pass/fail	Pass

ID	UN27
Title	request to get all notifications to a certain user with the right credentials
Expected Output	Get notifications associated with one user
Pass/fail	Pass

### 10.2.2 Functional Testing

ID	FNT01
User Story ID	US#2
User Story	As an unregistered user, I want to authenticate in the application so that I can access features that unauthenticated users cannot.
User Story Priority	Nice to Have
Scenario	The user provides a valid set of credentials
Pass/Fail	PASS

ID	FNT02
User Story ID	US#8
User Story	As an admin, I want to add a new workflow so that a new occurrence type can be automated and described in the application.
User Story Priority	Nice to Have
Scenario	The user provides an invalid set of credentials.
Pass/Fail	PASS

ID	FNT03
User Story ID	US#8
User Story	As an admin, I want to add a new workflow so that a new occurrence type can be automated and described in the application.
User Story Priority	Must Have
Scenario	The user wants to add a workflow to the system
Pass/Fail	PASS

ID	FNT04
User Story ID	US#9
User Story	As an admin, I want to edit an existing workflow so that will improve the efficiency of the system
User Story Priority	Must Have
Scenario	The admin wants to substitute the original diagram from another
Pass/Fail	Pass

ID	FNT05
User Story ID	US#9
User Story	As an admin, I want to edit an existing workflow so that will improve the efficiency of the system
User Story Priority	Must Have
Scenario	The admin wants to modify variables to the already deployed diagram
Pass/Fail	Pass

ID	FNT06
User Story ID	US#10
User Story	As an admin, I want to add observations to the submitted report so that it can be more easily understood.
User Story Priority	Should Have
Scenario	The user wants to provide additional information to the request
Pass/Fail	Pass

ID	FNT07
User Story ID	US#12
User Story	As an admin, I want to create groups of workers so that they would be represented in the application.
User Story Priority	Must Have
Scenario	The user wants to create a group of workers in the application
Pass/Fail	Pass

ID	FNT08
User Story ID	US#13
User Story	As an admin, I want to add new employees to the platform so that they will be represented.
User Story Priority	Must Have
Scenario	The user wants to add a worker to the system
Pass/Fail	Pass

ID	FNT09
User Story ID	US#14
User Story	As an admin, I want to see the statistics about the individual tasks of the workflows so that it is possible to make conclusions from them and see how they can be improved.
User Story Priority	Must Have
Scenario	The admin wants to see statistics about the different steps of a workflow.
Pass/Fail	Fail



ID	FNT10
User Story ID	US#15
User Story	As an admin, I want to see statistics about the different teams so that it is possible to evaluate their work.
User Story Priority	Should Have
Scenario	The admin wants to see statistics associated with current and past work.
Pass/Fail	Fail

ID	FNT11
User Story ID	US#16
User Story	As an admin, I want to see the Users and the tasks associated with them so that I can keep track of what they are doing.
User Story Priority	should have
Scenario	The admin wants to see users and the tasks associated with them.
Pass/Fail	Pass

ID	FNT12
User Story ID	US#17
User Story	As an admin, I Want to edit the existing teams so that I can modify the teams.
User Story Priority	should have
Scenario	The admin wants to change the existing teams
Pass/Fail	Pass

ID	FNT13
User Story ID	US#18
User Story	As an admin, I Want to eliminate existing teams so that it is possible to erase them from the platform.
User Story Priority	Should have
Scenario	The admin wants to delete an existing team
Pass/Fail	pass

ID	FNT14
User Story ID	US#19
User Story	As a worker, I want to see the occurrences that are attributed to me or my team.
User Story Priority	Must have
Scenario	The worker wants to see the occurrences left to be assigned
Pass/Fail	Pass

ID	FNT15
User Story ID	US#20
User Story	As a worker, I want to select the occurrence that I am going to resolve and remove it from the list when it is resolved.
User Story Priority	Must have
Scenario	The worker wants to select a task to execute
Pass/Fail	Pass

ID	FNT16
User Story ID	US#21
User Story	As a worker, I want to change the current status of the occurrence to inform the citizens and other entities.
User Story Priority	Must have
Scenario	The worker wants to change the state of the occurrence.
Pass/Fail	Pass

ID	FNT17
User Story ID	US#22
User Story	As a worker, I want to ask for Human resources to resolve the occurrence to solve it better or faster.
User Story Priority	Must have
Scenario	The worker needs the support of other teams to complete a task
Pass/Fail	Pass

ID	FNT18
User Story ID	US#23
User Story	As a worker, I want to receive notifications about the new occurrence that I was appointed to have that information.
User Story Priority	Must have
Scenario	The worker needs to know what occurrence was appointed to him.
Pass/Fail	Pass