Tunisian republic
Ministry of Higher Education and Scientific Research

Carthage University

Higher Institute of Information and Communication Technologies

End of Studies Project Report

Fulfillment of the requirement of
Bachelor's degree in Software Engineering and Information System

Major : Computer Science
Speciality : Software Engineering and Information System

Design and Implementation of a Comex Workflow Management Application under the Microservices Architecture

Authors
Farah Hend
Bessrour Hichem

Achieved within Poulina Holding Group

Professional supervisor : Ikhlass Baccouche
Pedagogic advisor : Fahima Ben Guirat

Academic Year : 2021-2022

Appreciation

A particular acknowledgement is for ...

...

Dedication

All my thankfulness is for ...

...

Table des matières

Table des figures

Liste des tableaux

General Introduction

In the field of management, the organization of work is a complex problem, information's exchange, tasks's division, and deadlines are the main issues that burden the organization of the company.

Managing any project needs to be organized, fluid and progressive in order to advance the process. In addition, the field of innovation requires flawless coordination between the different actors and poles of a company in order to manage a project. So the success in your business process will depend on this coordination and collaboration both between internal team members and with external actors. All these participants must be able to follow the progress in real time and communicate with each other at the different phases of execution.

In general, workflow management systems can help streamline and coordinate business processes. These business processes are represented as workflows, this tool enhances the validation process, it shares with each intervener the information necessary to execute his tasks and indicates the the deadlines to complete the assignment and recalls the validation methods.Its various functions allow it to organise the different tasks that personnel must perform and to ensure that each task is performed correctly.This is why workflow is important.

The workflow makes it possible to memorize and preserve the way in which a project was carried out.This can be useful in order to return to the source in case of an error or unexpected incident, but also in order to establish a standard model of workflow for the organization of future projects, thus reducing the time of documentation each time, by making everything automated.

The ultimate goal is to transform paper workflows into digital processes, in order to obtain a dynamic visual aspect as well as optimal management approved by all.

Hence the interest of our end-of-study project, which aims to implement an application to automate workflow modeling, facilitated by the computerisation of processes. By creating a simplified work environment that makes employees have more time to focus on their immediate tasks, with more attention and concentration on product delivery and quality, not wait for administrator's approvals while senior management can save time by doing it all in this software environment.

This report consists of a presentation of the work done during this internship :

— **Chapter 1** entitled " Project Framework " which begins with the presentation of

the host organization, the presentation of the project, critic of the existing, proposed solution and the methodology of work.

— **Chapter 2** entitled " Needs Analysis " which presents the requirements part of the functional and non-functional needs.

— **Chapter 3** entitled " Sprint0 : MicroService : Manage administrative affairs " which describes the first sprint .

— **Chapter 4** entitled " Sprint1 : MicroService : Manage subsidiary "which describes the second sprint .

Chapitre 1

Project Framework

1.1 Introduction

In this first chapter, we will start with a brief introduction to the company. Next, we will make a presentation of the project, critic of the existing and the solution envisaged, before talking about the working methodology used to carry out this project. Finally, we will conclude this chapter .

1.2 Presentation of the host organization

1.2.1 Presentation

The first private group in Tunisia, showing a good financial situation characterized by average debt and cost control, Poulina Group Holding seems to be able to improve its positioning in the various sectors of activity thanks to an ambitious investment program financed, in part, by this Increase in capital.

Despite a very aggressive and competitive environment in which The group evolves, PGH has large means to achieve it growth goals.

Poulina Group Holding aims to manage and acquire holdings in industrial or service companies .

1.2.2 Historic

The Pauline group was founded in 1967 on the initiative of private Tunisian promoters in the poultry sector. The dynamism of the company articulated around an integration vertical and horizontal strategy allowed for the development of a multisectoral group. Working on various activities .

In 41 years, the Poulina group has become the first group with private capital in the country, it currently has over 70 companies, including fifteen established abroad. The group has establishments in Algeria, Morocco, Libya, France and recently in Saudi Arabia and China.

As part of the restructuring carried out as a prelude to its Initial public offering, In June 2008, a holding company known as Pauline Group Holding (PGH) was set up consist in six mini-holdings .

1.2.3 PGH Business sector

PGH owns today more than 108 companies and structured on the basis of nine activity sectors among them :

— **Poultry Sector** : Commerce and storage of animal nutrition, production and distribution of eggs and poultry meat.

— **Commerce and Services** : International commerce, commerce in the local market, trading (import/export), distribution, iT services (exclusivity of a few brands such as NEC), tourism (two hotels in the Solaria chain and the Médina tourist complex in Yasmine Hammamet), fast food and telemarketing.

— **Agrifood industry** : Manufacture and marketing of ice cream, Margarine, yoghurts, desserts, dairy products, crisps, juices and industrial pastries.

— **Packaging Sector** : Processing of cardboard, paper, honeycomb, stretched film and printing on plastic film (food packaging and cardboard boxes).

— **Estate Sector** : Estate Development and Purchase,importing and marketing of building components and products. Poulina Group Holding entered in the real estate sector through its subsidiary «ETTAAMIR» created in 1997, and became an essential actor in the estate sector in Tunisia .

1.3 Project presentation

Our project entitled "Design and Implementation of a Comex Workflow Management Application under the Microservices Architecture" is used to develop a platform that connects between internal team members and external actors and to which the employee of a subsidiary sends a request, then the director consults the list to approve or reject the request, finally the employee check their request with their status .

1.4 Critic of the existing

Paper-based processes are incredibly inefficient, they generate a lack of follow-up of requests and precision of information, impede productivity by wasting time , result in higher costs, and even pose security risks and several other failures .

There are many applications to draw a workflow. Most use an XML language to describe it. They make it possible to do much stuff, but they are also often extremely particular. Since this case is very specific, it was required to develop a product adapted to this environment. So "Poulina Group Holding" has proposed to us to implement the workflow solution that exactly satisfying its needs .

1.5 Proposed Solution

In order to avoid issues is in the previous section, the solution is Workflow Engine a process receives inputs and produces outputs and the best workflow engine solution will depend on their needs and existing software stack.

For instance, in our case we need to have specific requests reviewed by specific individuals. We propose a workflow solution that ensured each request made its way to everyone required to provide an answer. After getting the approval or refuse, the workflow should send the completed response back to relevant participant.

1.6 Methodology of work

1.6.1 Scrum

In order to have a suitable quality of a project and to meet the deadlines set in advance, we chose Scrum as a steering methodology for our project.

Why Scrum ?

Scrum is the most used Agile method nowadays. In short, projects that follow the “SCRUM” Agile method are divided into several cycles of relatively short work called “sprints”. These allow team members to regularly evaluate project progress and plan next steps for development. But above all, it makes it possible to adjust or reorient the direction taken by the project if necessary, from a base of work already completed and validated (sprint).

1.6.2 Work tools

To implement this project, we used development tools and software configured and installed on a Windows 10 operating system. These development tools are :

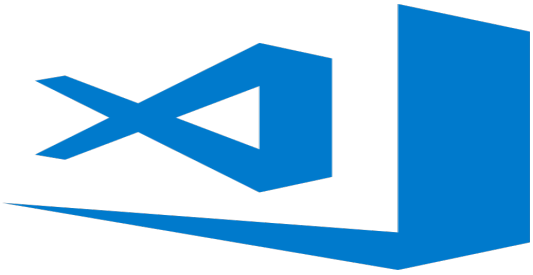
An integrated development environment (IDE) is a feature-rich program that supports many aspects of software development. The Visual Studio IDE is a creative launching pad that you can use to edit, debug, and build code, and then publish an app. Over and above the standard editor and debugger that most IDEs provide, Visual Studio includes compilers, code completion tools, graphical designers, and many more features to enhance the software development process.

QL Server Management Studio (SSMS) is an integrated environment for managing any SQL infrastructure. SSMS provides tools to configure, monitor, and administer instances of SQL Server and databases. Use SSMS to deploy, monitor, and upgrade the data-tier components used by your applications, and build queries and scripts.
Use SSMS to query, design, and manage your databases and data warehouses, wherever they are on your local computer or in the cloud.

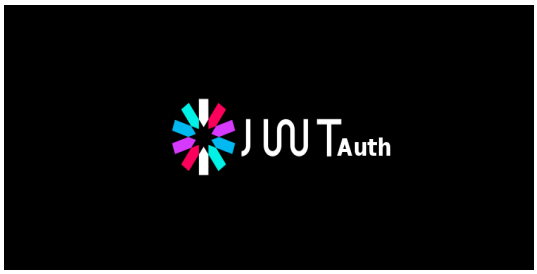
NET is a Free and Open Source Framework for the Windows, macOS and Linux3 operating systems. It includes CoreCLR, a complete runtime environment for CLR, the virtual machine that manages the execution of .NET programs.

.NET Core fully supports C and F and partially supports Visual Basic.Net. Currently, VB.NET compiles and runs on .Net Core, but the separate Visual Basic Runtime is not implemented. Microsoft has announced4 that .NET Core 3 will include Visual Basic Runtime. C++/CLI is not supported.

Angular is an application design framework and development platform for creating efficient and sophisticated single-page apps. These Angular docs help you learn and use the Angular framework and development platform, from your first application to optimizing complex single-page apps for enterprises. Tutorials and guides include downloadable examples to accelerate your projects..



Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, macOS and Linux. It comes with built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages (such as C++, C, Java, Python, PHP, Go) and runtimes (such as .NET and Unity).



JSON Web Token (JWT) is an open standard (RFC 7519) that defines a compact and self-contained way for securely transmitting information between parties as a JSON object. This information can be verified and trusted because it is digitally signed. JWTs can be signed using a secret (with the HMAC algorithm) or a public/private key pair using RSA or ECDSA.

Although JWTs can be encrypted to also provide secrecy between parties, we will focus on signed tokens. Signed tokens can verify the integrity of the claims contained within it, while encrypted tokens hide those claims from other parties. When tokens are signed using public/private key pairs, the signature also certifies that only the party holding the private key is the one that signed it.

Swagger allows you to describe the structure of your APIs so that machines can read them. The ability of APIs to describe their own structure is the root of all awesomeness in Swagger. Why is it so great ? Well, by reading your API's structure, we can automatically build beautiful and interactive API documentation. We can also automatically generate client libraries for your API in many languages and explore other possibilities like automated testing. Swagger does this by asking your API to return a YAML or JSON that contains a detailed description of your entire API. This file is essentially a resource listing of your API which adheres to OpenAPI Specification..

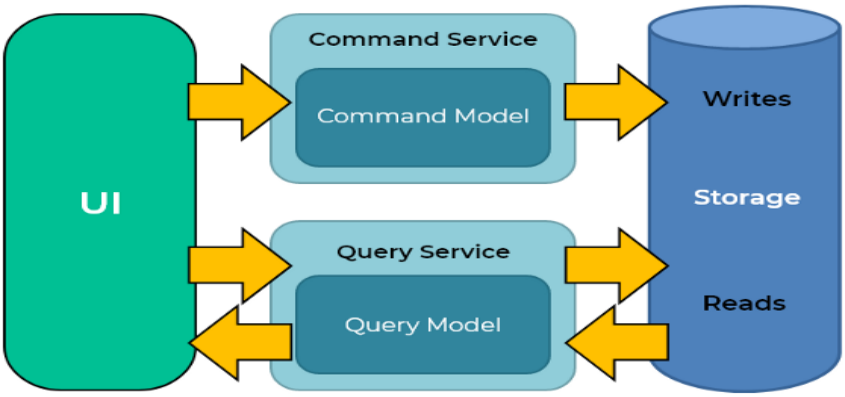
CQRS Architecture :

The Command and Query Responsibility Segregation "CQRS" is an architectural pattern that separates the read and writes operations of a data source. Here Query refers to querying data from a source and Command refers to database command that can be either Insert/Update or Delete operations.

So the main idea with CQRS is to allow an application to work with different models. In traditional architectures, the same data model is used to query and update data sources. But in practical applications, there is a difference between reading and writing forms of data, like extra validation or properties, which are needed for an update or insert operation whereas the read model doesn't need any of these. But with CQRS you can have different data models for update/insert and query which provides flexibility for complex scenarios.

Even in more complex applications like microservices or any kind of application that has high demand of data consumption(like StackOverflow) in that case the traditional architecture doesn't fit well because you will stick with the same data source and having much writing and reading in the same data source can affect the performance of the system.

In that scenario, we can think about CQRS because it gives us the flexibility to use a separate data source that ensures the scalability of our application.

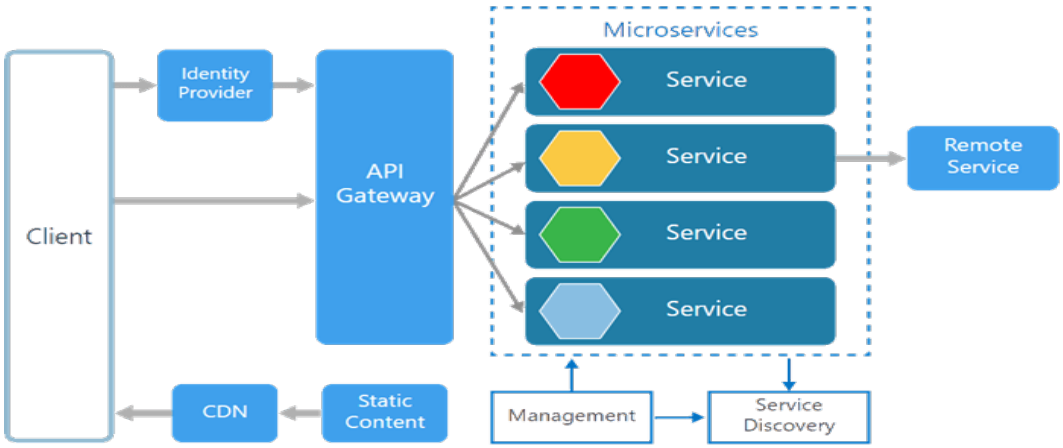


Microservices Architecture :

Microservices also known as the microservice architecture , is an architectural style that structures an application as a collection of services that are Highly maintainable and testable , Loosely coupled , Independently deployable and Organized around business capabilities.

The microservice architecture enables the rapid, frequent and reliable delivery of large, complex applications. It also enables an organization to evolve its technology stack.

Microservices architecture (often shortened to microservices) refers to an architectural style for developing applications. Microservices allow a large application to be separated into smaller independent parts, with each part having its own realm of responsibility. To serve a single user request, a microservices-based application can call on many internal microservices to compose its response.



1.7 Conclusion

This chapter presents an overview of the framework of this project, we have presented the critique of the existing in order to propose a solution.
The next chapter is devoted to the presentation of functional and non-functional requirements.

Chapitre 2

Specification of requirements

2.1 Introduction

The purpose of a project is to satisfy a need, so it must be clearly expressed .
In this context we begin with the capture of needs which generally formulated in the form of functional and non-functional requirements, identification of actors, product backlog, and at the end the description of global use case diagram.

2.2 Needs Analysis

2.2.1 Functional requirements

A functional requirement is a description of what a system must do and how it must behave from a user perspective.

the system aims to :

√ Authenticate.
√ register.
√ Manage administrative affairs.
√ Manage subsidiary.
√ Manage intervention's requests.
√ Treat requests.

2.2.2 Non functional requirements

These requirements may relate to implementation constraints and they are important because they act indirectly on the result which means that they must not be negligeous, for that the following requirements must be met :

Reliability :
The system must function without errors and satisfied.

Ergonomics and good interface :
The system must be adapted to the user without any effort so the use must be clear and easy and the interfaces simple and ergonomic.
Methodology :
The system must comply with the SCRUM methodology for a quality website.
Security :
The system must, above all, respect the confidentiality of employees' personal data, which remains one of the most important constraints.

2.2.3 Identification of actors

An actor interacts directly with the system by transmitting and/or receiving possibly data-carrier messages.

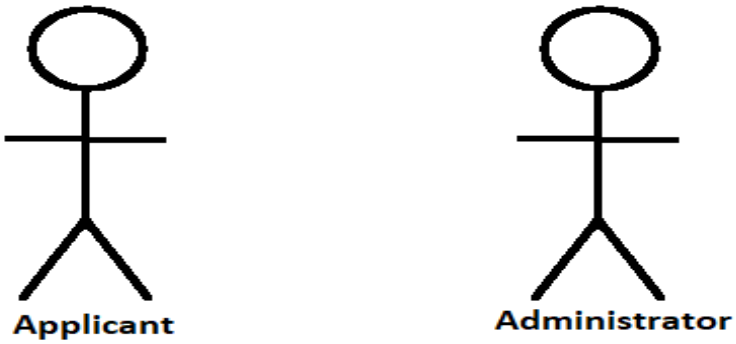


FIGURE 2.1 – Actors

The table below shows the roles of each actor :

Actors	Needs
Applicant	△ Register. △ Authenticate. △ Manage intervention's requests.
Administrator	△ Authenticate. △ Manage administrative affairs. △ Manage subsidiary. △ Treat requests.

TABLE 2.1 – The roles of each actor

2.2.4 Use Case Diagram

The following diagram represents the use cases of the architecture. It describes the system behaviour from the user's point of view.

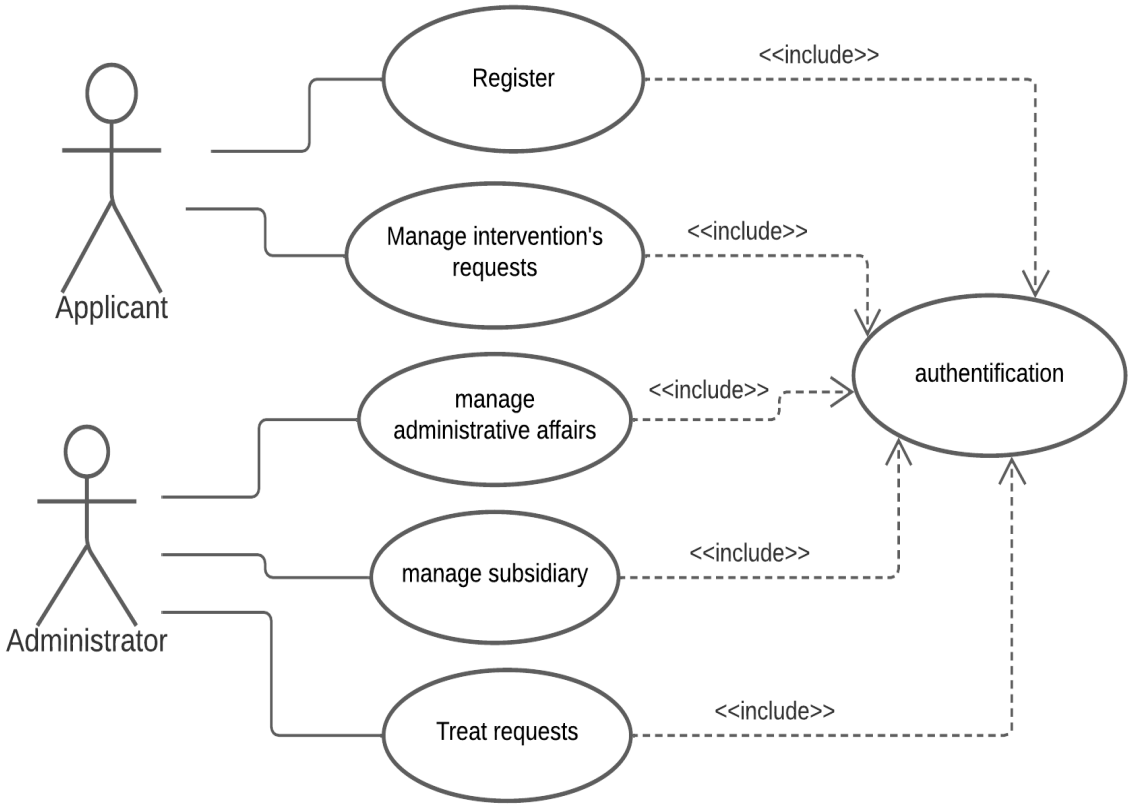


FIGURE 2.2 – Diagramme de cas d'utilisation globale

2.2.5 Product Backlog

The backlog is a list of prioritized tasks defining the characteristics of a product. It is one of the fundamental elements of the Scrum methodology. This is the Product Owner's main job aid that collects needs from stakeholders and transforms them into a list of features.

Product Backlog	Priority	Estimation	Planning	
As an applicant, I can register.	1	Medium	Sprint 0	
As an applicant, I can authenticate.	1	Medium	Sprint 0	
As an administrator, I can authenticate.	1	Medium	Sprint 0	Release 1
As an administrator, I can manage administrative affaires.	1	Strong	Sprint 0	
As an administrator, I can manage subsidiary.	2	Medium	Sprint 1	Release 2
an administrator, I can treat requests.	2	Strong	Sprint 1	

Table 2.2 – Product Backlog

2.2.6 Activity Diagram

Above the activity diagram allows to graphically represent the behavior of a method or the course of a use case. The transition from one activity

to another is materialized by a transition. Transitions are triggered by the end of one activity and cause the immediate beginning of another (they are automatic).

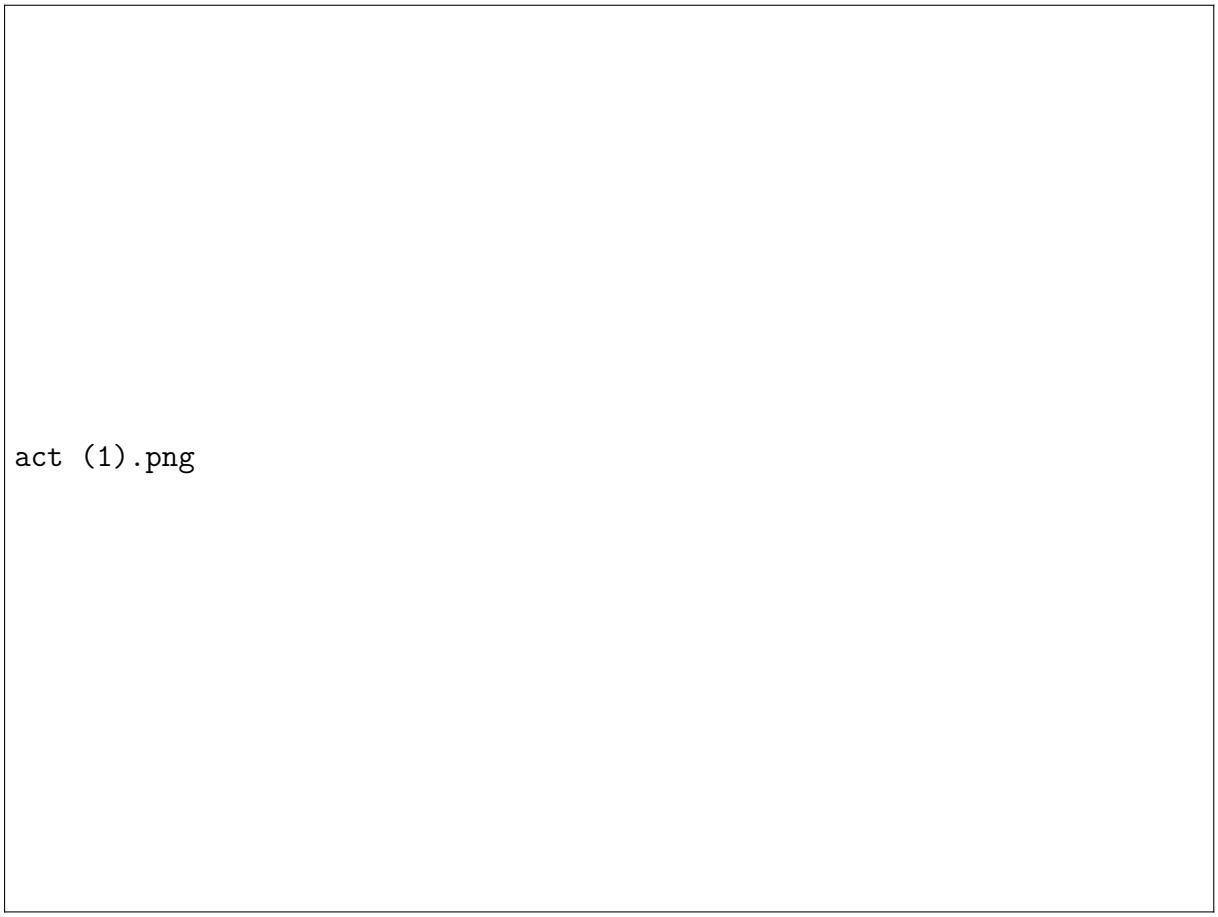


FIGURE 2.3 – Activity Diagram

2.3 Conclusion

In this chapter, we have tried to present the actors of this project, the functional and non-functional needs and the working methodology in order to clarify my objectives to be achieved.

Indeed, the specification of requirements allows us to prepare a good design and provides a deeper understanding of the tasks to be carried out that will make the content of the next chapter.

Chapitre 3

Sprint0 : MicroService : Manage administrative affairs

3.1 Introduction

During this chapter, we will trigger a sprint0 named "Manage administrative affairs". This first sprint covers the refinements of use cases, design, and realization.

3.2 Sprint 0 backlog identification

In this section, I will present sprint 0 backlog

Product Backlog	Priority	Estimation
As an applicant, I can register.	1	Medium
As an applicant, I can authenticate.	1	Medium
As an administrator, I can authenticate.	1	Medium
As an administrator, I can manage administrative affaires..	1	Medium

TABLE 3.1 – Sprint 0 backlog identification

3.3 Refinement of priority use cases

Refinement of use cases provides more detail on the features provided by the system and associated constraints.

3.3.1 Refinement of the 'Register' use case

this is the refinement of the 'Register' use case

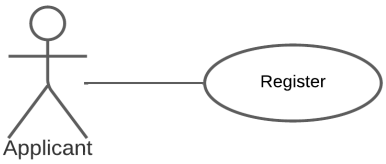


FIGURE 3.1 – Use Case Diagram "Register"

Relative text description for requirement "Register"

use case	Register .
Actor	Applicant.
Pré-condition	View the platform.
Post condition	create an account.
Description of the main scenario	1. The system displays a registration form to the employee. 2. The employee enters their information. 3. the system checks the validity of the information entered. 4. the system records this information in the database.
Exception	the system displays the registration failure by an error message.

TABLE 3.2 – Refinement of the 'Register' use case

3.3.2 Refinement of the 'Authenticate' use case

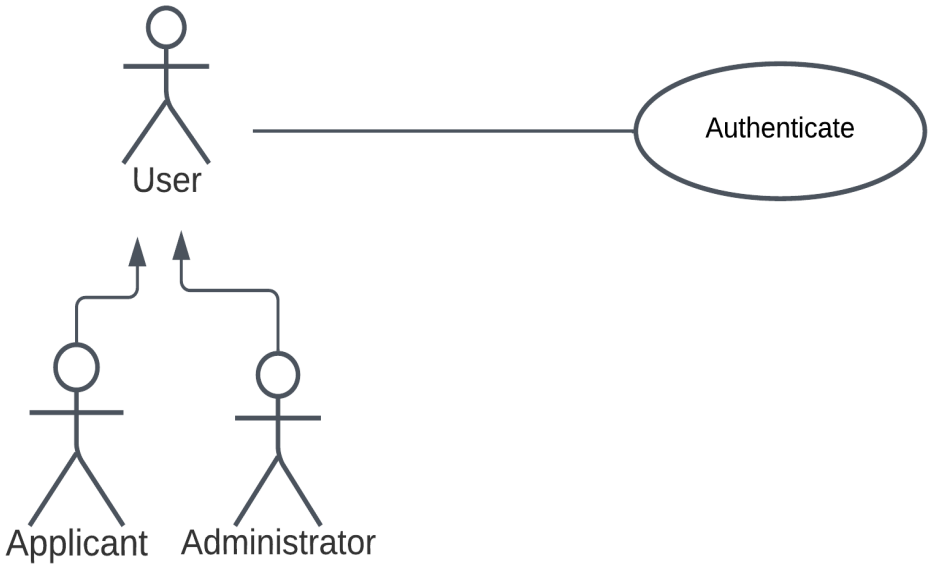


FIGURE 3.2 – Use Case Diagram 'Authenticate'

text description for requirement "Authenticate" :

use case	Authenticate.
Actor	User.
Pré-condition	user has an account.
Post condition	the user authenticates.
Description of the main scenario	1. The system prompts the user to enter their username and password. 2. Actor enters username and password. 3. the user clicks on the login button. 4. The system checks the parameters. 5. The system opens the space corresponding to the profile.
Exception	The email and/or password is incorrect : The system informs the user that the data entered is incorrect.

TABLE 3.3 – Refinement of the 'Authenticate' use case

3.3.3 Refinement of the 'Manage administrative affaires' use case

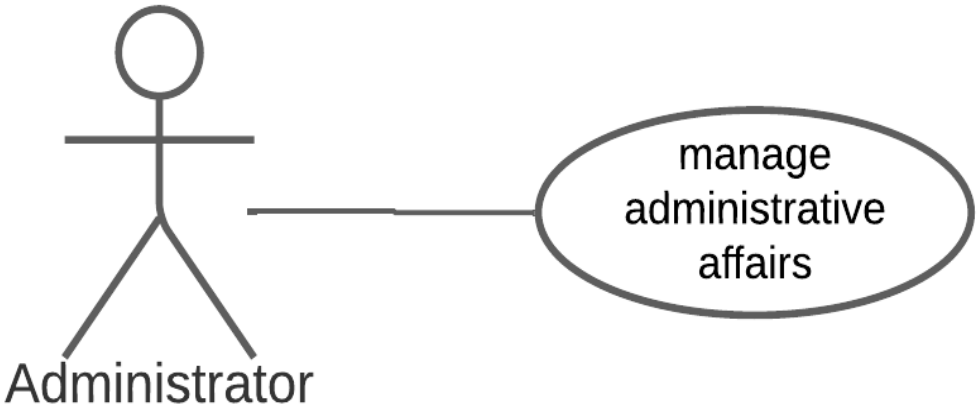


FIGURE 3.3 – Use Case Diagram "Manage administrative affaires"

text description for requirement "Manage administrative affaires" :

use case	Manage administrative affaires .
Actor	Administrator.
Pré-condition	authenticated administrator.
Post condition	administrative affairs will be managed.
Description of the main scenario	1. The system displays the interface. 2. The user chooses the operation to perform. 3. The system displays the appropriate interface.
Scénaries	— Add user. — Consult user. — Update user. — Delete user. — Add role. — Consult role. — Update role. — Delete role . — assign access rights.

TABLE 3.4 – Refinement of the 'Manage administrative affaires' use case

3.4 Design

Design is an important phase before any project is completed, This phase requires methods to enable a model to be built upon.

That is to say, to create a representation similar to reality in such a way as to highlight the points we are interested in.For this work I opted for the UML modeling language.

3.4.1 Use Case Design "Register"

class diagram

The class diagram is considered to be the most important of the modelling object oriented, it is the only mandatory during such modelling. While the use case diagram shows a system from the point of view of of the actors, the class diagram shows its internal structure. It abstract representation

of system objects that will interact to realize cases of use.

The diagram below is the participating class diagram for the "Register" function.

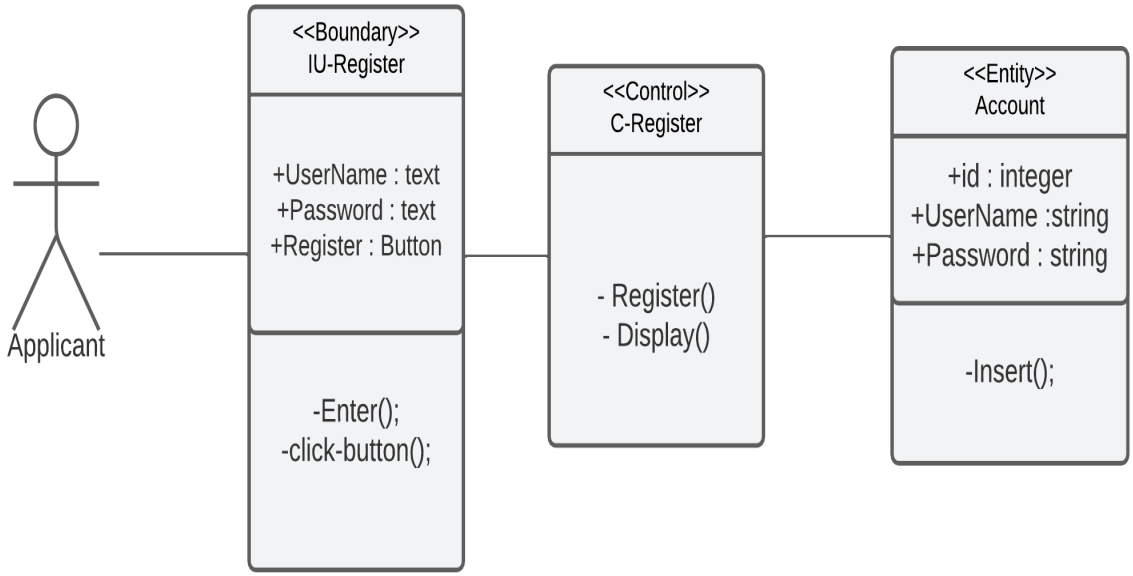


FIGURE 3.4 – Use Case Class Diagram ”Register”

Sequence diagram

The diagram below is the participating sequence diagram for the "Register" function.

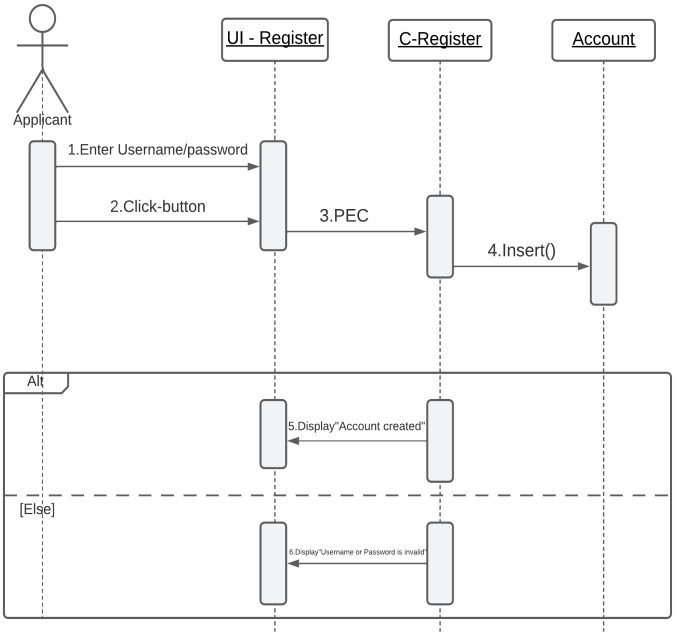


FIGURE 3.5 – Use Case Sequence Diagram ”Register”

3.4.2 Use Case Design ”Authenticate”

class diagram

The diagram below is the participating class diagram for the "Authenticate" function.

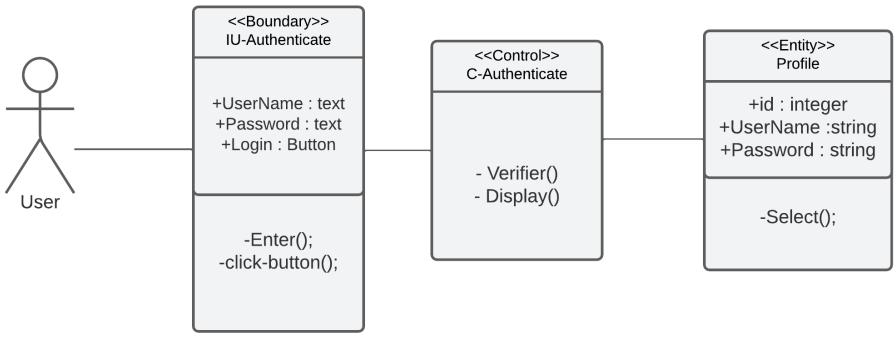


FIGURE 3.6 – Use Case Class Diagram ”Authenticate”

Sequence diagram

The diagram below is the participating sequence diagram for the "Authenticate" function.

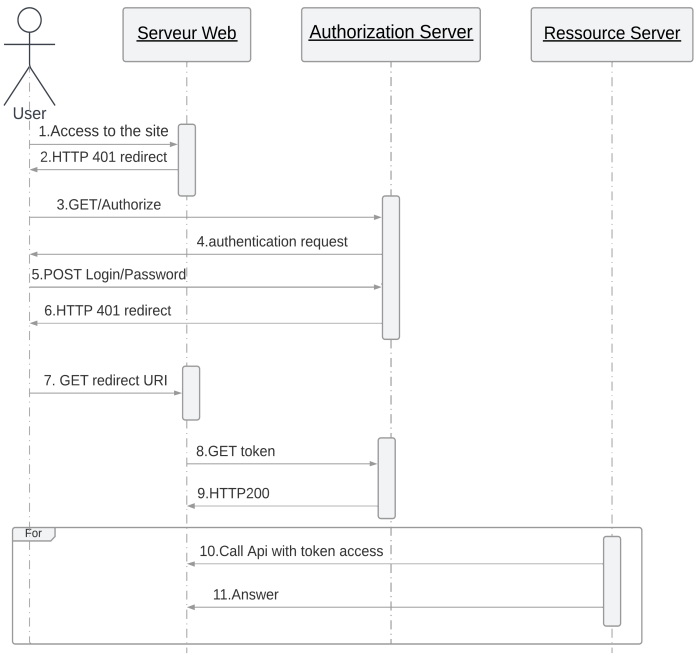


FIGURE 3.7 – Use Case Sequence Diagram ”Authenticate”

3.4.3 “Manage administrative affaires” use case design

class diagram

The diagram below is the participating class diagram for the "Manage administrative affaires" function.

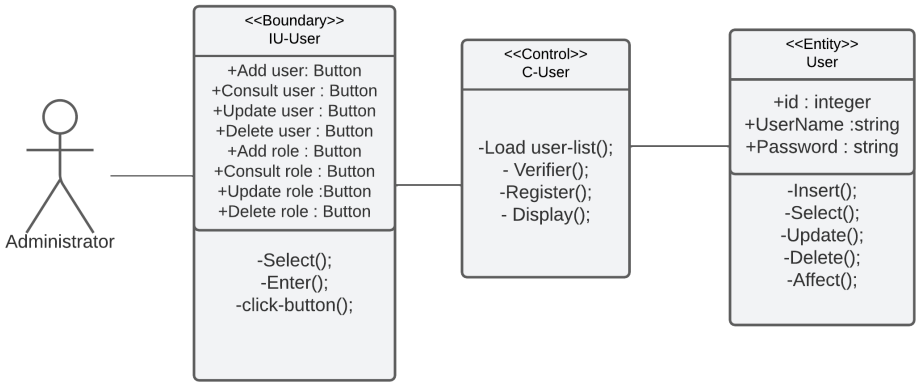


FIGURE 3.8 – Use Case Class Diagram ”Manage administrative affaires”

Sequence diagram

The diagram below constitutes the participating sequence diagram for the function "Manage administrative affairs".

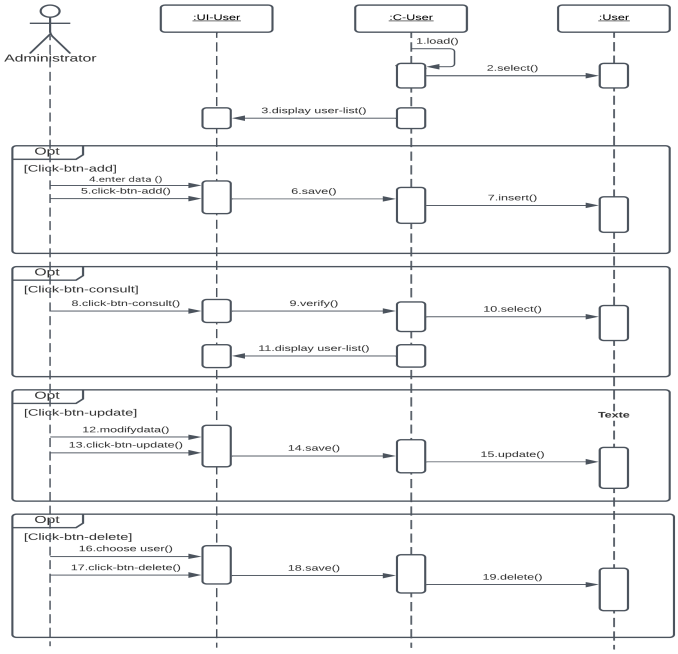


FIGURE 3.9 – Use Case Sequence Diagram "Manage administrative affairs"

3.5 Realisation

3.5.1 Register

The registration interface : the user enters his username and password presented by the figure .

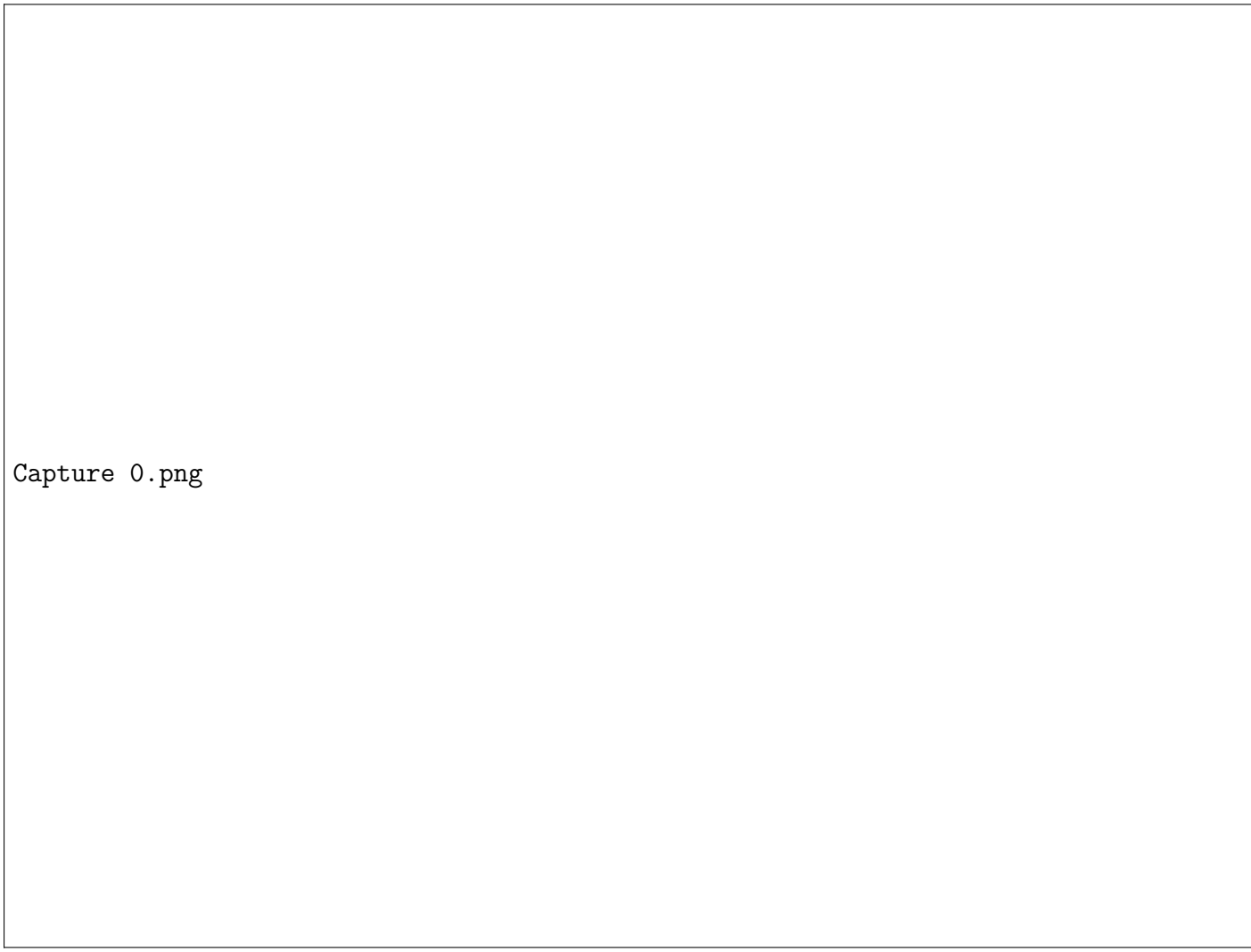


FIGURE 3.10 – Registration interface

3.5.2 Authenticate

The authentication interface : this interface allows the user to enter his username, his email and his password in order to access his own account as shown in the figure .

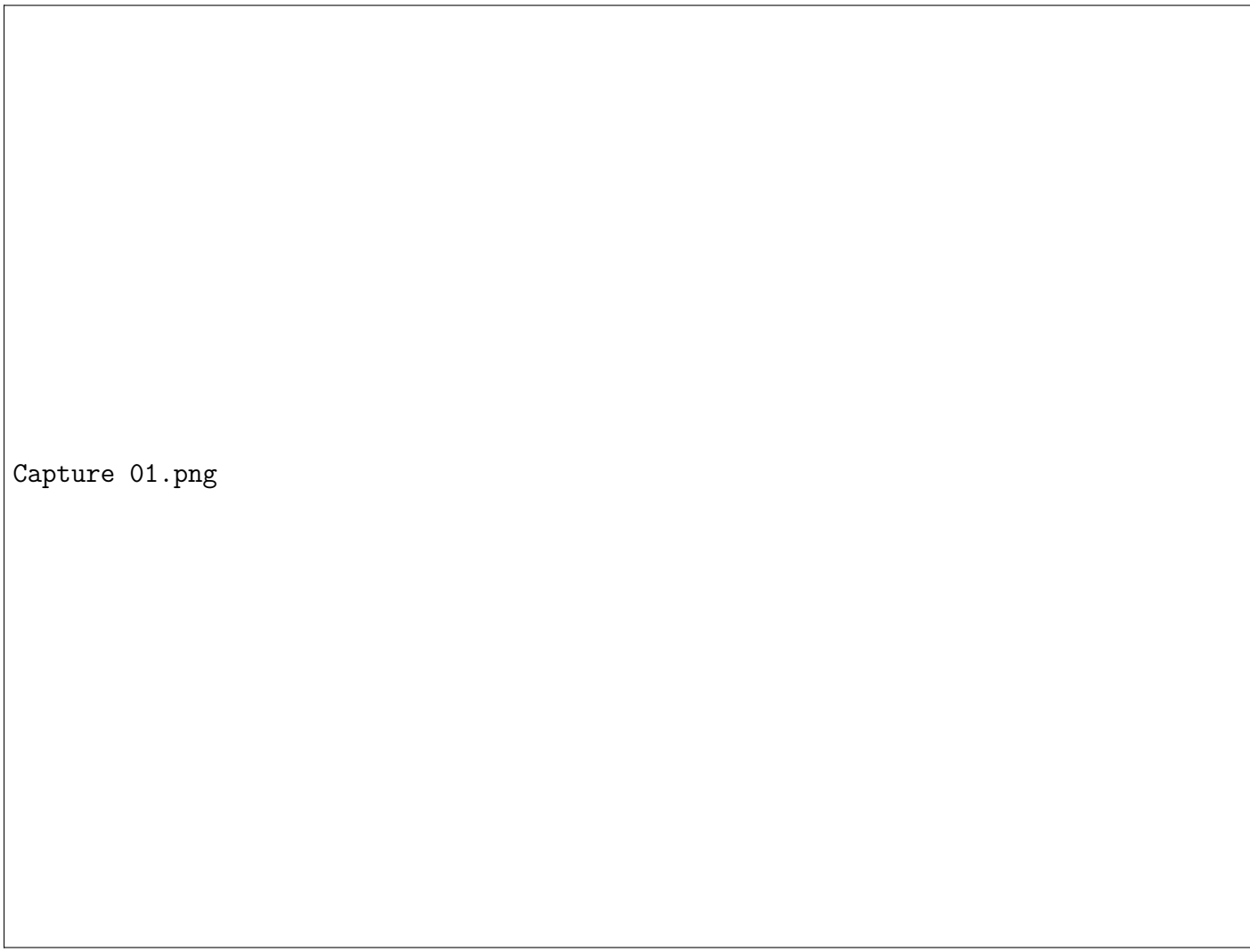


FIGURE 3.11 – Authentication interface

3.5.3 Space « Employee »

The interface below is the first page that Staff gets once authenticated.

Capture 10.png

FIGURE 3.12 – Home interface for a staff

3.5.4 Manage administrative affaires

Capture 00.png

FIGURE 3.13 – main interface

The following screen is the interface through which the administrator can add a new employee .

FIGURE 3.14 – Interface contains all users

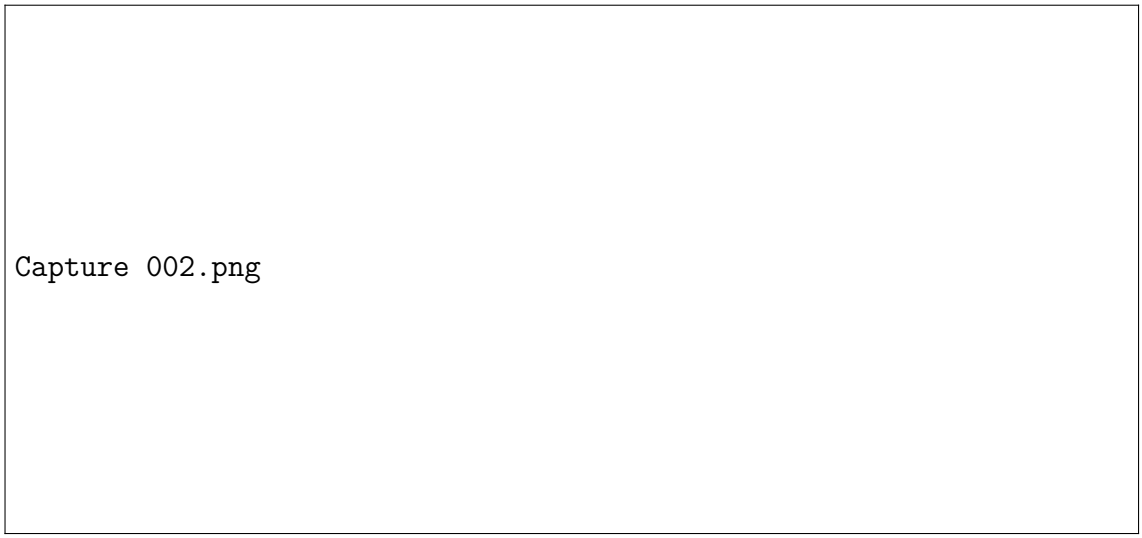


FIGURE 3.15 – Interface contains all roles

3.6 Conclusion

Throughout this sprint, we are interested in the administrative part. We also presented the various diagrams made. Its various points led us to give a final version of this project.

Chapitre 4

Sprint1 : MicroService : Manage subsidiary

4.1 Introduction

This chapter aims to presents the sprint1. First, we present the refinement of the use cases, the sequence diagrams and class diagrams. Next, we describe the work done in detailing some screenshots of the functionalities performed.

4.2 Identification of Sprint 1 backlog

In this section, I will present sprint1 backlog

Backlog de produit	Priorité	Estimation
As an administrator, I can manage subsidiary.	1	Medium

Table 4.1 – Identification de Backlog de Sprint 1

4.3 Conclusion

We presented the backlog of the sprint1 product by specifying the various features that make it up. We have also detailed the different design elements, namely sequence diagrams and class diagrams with some interfaces. The next chapter is dedicated to the sprint1 based on the adapted AGILE methodology which is SCRUM.

General Conclusion

Bibliographie