Design Document

Students&Companies project Andrea Bellani Alessandro Capellino

Andrea Bellani, Alessandro Capellino

January 7, 2025

Politecnico di Milano

Contents

List of Figures

List of Tables

Introduction 1

1.1 Purpose

During their university studies, in order to start entering the workforce, a student might decide to apply for an internship related to their field of study. Similarly, companies offering internships may be interested in finding students that are adequate for them. To facilitate the matching between students and companies, a new platform called *Students and Companies* (S&C) is to be developed. S&C allows companies to look for suitable students by publish internship advice on the platform, while students can look for internships that interest them. Moreover, the platform implements recommendation mechanism to help student and companies to find each other. Once the contact is established, S&C can provide support to the students selection process.

1.2 Scope

As mentioned in the RASD document, there are two main users categories that interact with the system: *Companies* and *Students*. The companies publish announcements about the internships they want to offer where they specify *projects* that will be carried out and the *terms* of the offer. The system itself informs the companies about the availability of students who may be suitable for their internships (based on their profile).

Students, on the other hand, may use the platform to look for internships and S&D can also notify them if there are new internships that could meet their interests, but they can still independently search through all the available internships.

Once a *contact* is established and accepted by the two parties, the student selection process begins. At this point the company defines selection steps and schedules the interviews for each student. Once the selection is over, the system collects feedback and suggestions from both students and companies.

Finally, both students and companies can monitor the progress of the internships by providing information on its development and any issues that may arise.

1.3 Definitions, Acronyms and Abbreviations

1.3.1 Definitions

- **internship advice** : a call for application related to an internship that will be offered by a company;
- **recommendation** : the mechanism related to the fact that the system both informs students whether new internship advice that might interest them are published and notifies companies of the presence of students that might be suitable for their internships;
- **project** (of an internship advice) : the definition of the application domain, the set of tasks to be performed and the set of the most relevant adopted technologies (if any) for an internship;
- **terms** (of an internship advice) : the set of benefits offered by an internship (e.g. paid/not paid, training, lunch voucher...);
- **selection process** : each internship advice is followed by a sequence of selection steps.

1.3.2 Acronyms

- S&C: Students&Companies, the name of the platform;
- UML: Unified Modeling Language;
- CV: Curriculum Vitae.
- DB: Database
- RDBMS: Relational Database Management System
- API: Application Programming Interface

1.3.3 Abbreviations

- Gn: Goal number n;
- Rn: Requirement number n;
- Dn: Domain assumption number n;
- WPn: World Phenomena number n;
- SPn: Shared Phenomena number n;
- UC: Use Case.
- UI: User Interface

1.4 Revision history

First version
22/12/2024

1.5 Reference documents

The Documents used to deliver the RASD document are the following:

- the Specification of RASD and DD assignment of Software Engineering 2;
- the class slides on WeBeep, in particular slides on UML diagrams, software architectures styles and the part related to Integration and Testing

1.6 Document structure

1. **Introduction**: this section provides a brief introduction to the purpose of the platform to be developed, S&C in this case, focusing in particular on the most important goals which the system has to achieve and on the various phenomena identified, as we mentioned in the RASD document;
2. **Architectural Design** : this section provide a description on the general architecture chosen for the system, at different levels of granularity: it focuses on the component of the architecture, on how they are related (showing also how the most important functionalities of the platform are provided) and on the deployment view;
3. **User Interface Design** : in this section it is explained how the User Interface is designed to provide the functionalities of the platform
4. **Requirement Traceability** : in this section is explained how the goals described in the RASD document are satisfied showing the interaction between the requirements and the components of the architecture;

5. **Implementation, Integration and Test Plan**: this section provides a description of how the components of the system are integrated and then tested; this information are useful for the developers of the platform
6. **Effort Spent**:report of the time spent by any group member in any document section;
7. **References**: list of software and documents used to develop the document.

Architectural Design 2

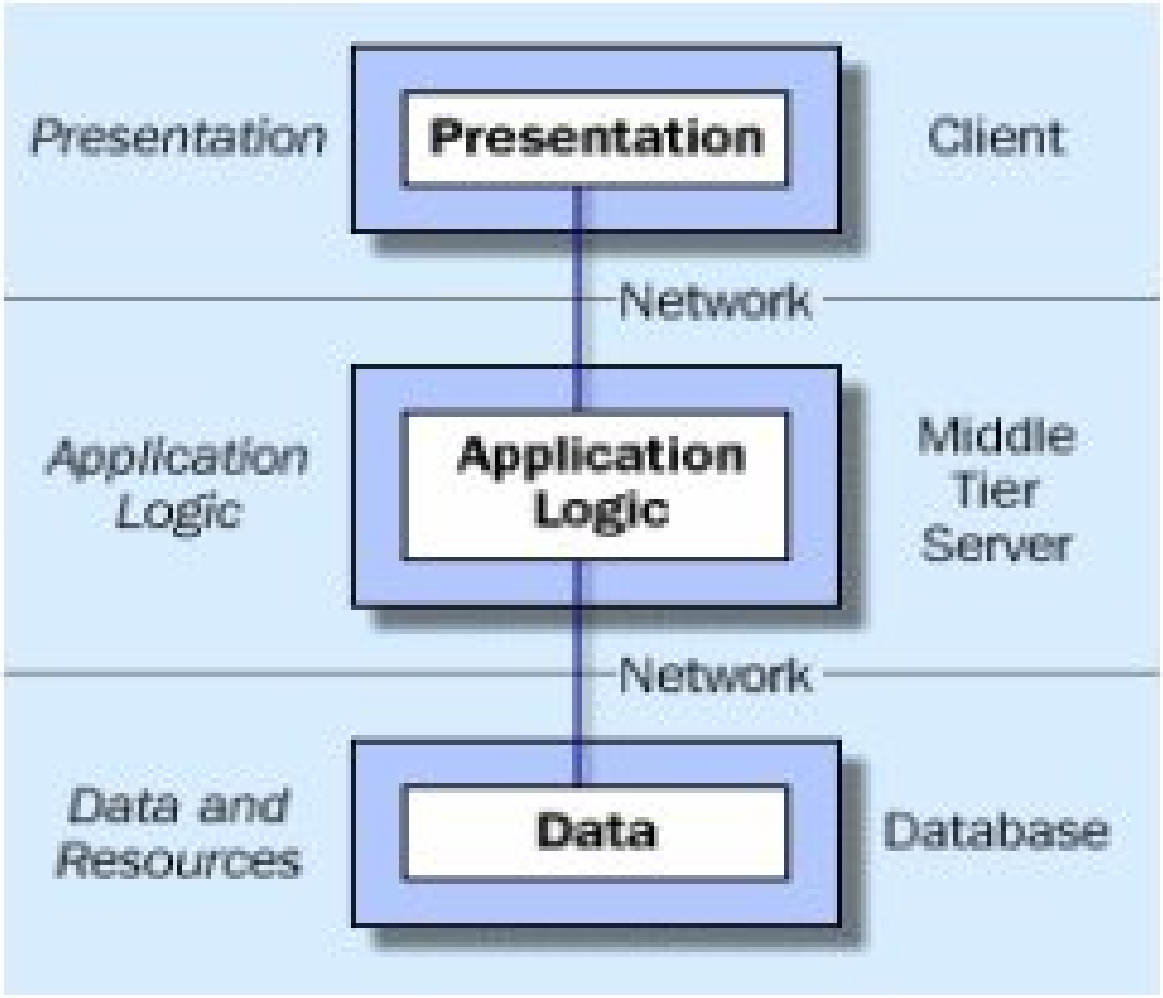
2.1 Overview

From an implementation perspective, it was decided to develop the S&C platform as a web application.For this reason, the architecture chosen for its development is a 3-tier architecture. It is a software design pattern that separates applications into three distinct layers:

1. **Presentation Layer**: The user interface (UI) where users interact with the application
2. **Application Layer**: the layer in which the data are processed
3. **Data Layer**: manages the storage, retrieval, and manipulation of data, involving databases

This architecture offers significant advantages in terms of scalability, maintainability, and especially modularity, allowing each layer to be developed and updated independently of the others.

Going into detail, the data layer employs two different types of databases: a relational database is used to manage the structured data of our system, which constitutes the majority of the data utilized by the platform (users, internship advice, internships, notifications, etc.). For the management of unstructured data, namely CVs and selection-related questionnaires, a non-relational database was chosen. Specifically, MongoDB was selected, as its JSON-based data storage format is perfectly suited for hierarchical and non-rigid structures, such as CVs and questionnaires.



2.2 Component View

As mentioned, the platform is developed used a 3-tier architecture approach with 2 DB(a RDBMS and MongoDB); As for the back-end components, four main macro-components can be identified, which are:

1. **Social Application System**: It provides the "social" functionalities of the platform, such as user registration/login, searching for internship advice, viewing profiles, applying for/sending an application proposal, sending feedback/complaints, and so on. It is divided in several components:
   a) **Authentication Manager**: It handles user authentication (initial registration, subsequent logins);
   b) **Application Manager**: It is responsible for managing application proposals, acceptances, and application requests;
   c) **Advice Presenter**: It provides functionalities for viewing individual internship advice and the list of all internship advice published on the platform;
   d) **Advice Publisher**: It provides functionalities for publishing an internship advice;
   e) **Feedback Manager**: It provides functionalities for viewing feedback questionnaires and receiving the corresponding responses;
   f) **Internship Manager**: It displays the ongoing internships (both the list and individual ones) and manages user complaints;
   g) **Profile Manager**: it provides functionalities for managing a profile, from creation to modification and deletion
   h) **Profile Presenter**: It allows viewing profiles
2. **Recommendation System**: It is the component responsible for the recommendation functionality offered by the platform, which involves analyzing the necessary information (feedback, CVs for students, etc.), determining which internships/students may be of interest to a specific user, and finally sending this information to the interested user. It is divided into 3 sub-components:
   a) **Recommendation Interface**: It is the component that retrieves the necessary information from the databases to develop the recommendation;
   b) **Recommendation Analyzer**: t is the component that performs the analysis on the information obtained from the recommendation interface (feedback, CVs, etc.);
   c) **Recommendation Presenter**: It inserts the recommendation results into the database and triggers the sending of notifications to inform that the recommendation has been made
3. **Selection Management System**: It provides functionalities related to the entire selection process, from its configuration to the sending of results. It is divided in 3 sub-components:
   a) **SP Initializer**: It provides functionalities for configuring the selection process (dates, metrics, questionnaires, etc.);
   b) **SP Manager**: It manages the insertion of answers in the questionnaires and the subsequent finalization of the selection process
   c) **SP presenter**: It displays interview dates and the results of the selection process
4. **Notification System**: It is the component responsible for managing all notifications generated and sent by the system; it handles the creation, sending, and triggering of notifications with the mail service. It is divided in 2 sub-components:
   a) **Notification presenter**: It allows viewing notifications (both individual ones and the list);
   b) **Notification Generator**: It is the component that generates a notification and triggers the sending of an email;

there are other important components of the system:

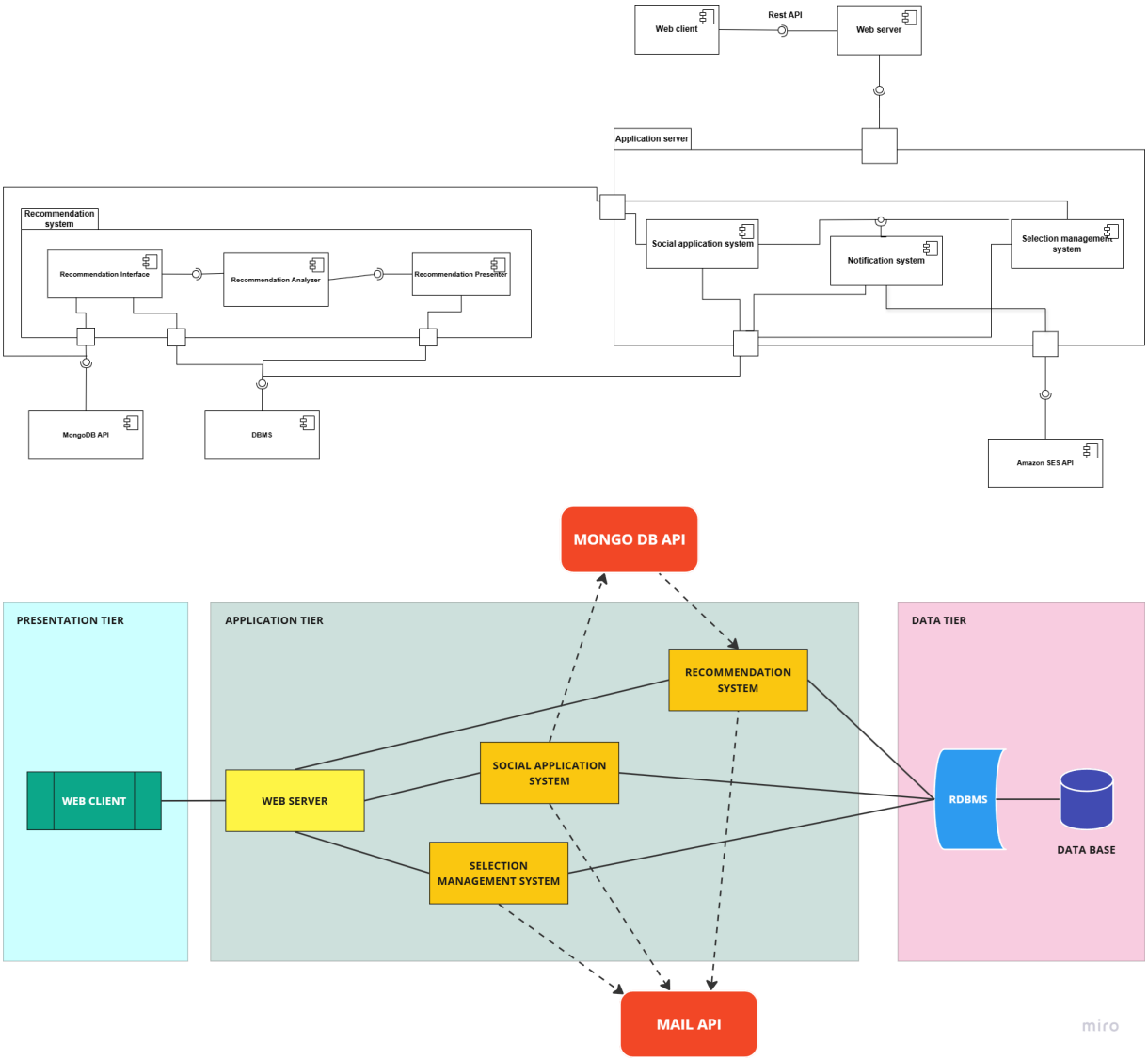
1. **DBMS**: stores all the structured data of the platform
2. **WebServer**: it is the component related to routing the incoming request to the appropriate internal component.

3. **MongoDB API**: A programming interface that allows developers to interact with a MongoDB database through HTTP requests, using CRUD operations (Create, Read, Update, Delete) on data stored in a MongoDB database;
4. **SES API**: Amazon Simple Email Service (SES) is an email sending and receiving service offered by Amazon Web Services (AWS). The SES API allows to easily integrate email sending functionality into applications/websites.

for the front-end, there are 2 clients which interact with the back-end:

1. **WebClient**: the UI of the application; it interacts with the back-end through REST API, which is based on a set of principles and constraints that allow systems to communicate over the web using standard HTTP methods such as GET, POST, PUT, DELETE, etc.
2. **Client Mail**: the client used to access and manage the mail; it communicates with the back-end (in particular with the Notification system) through SES API

the component diagram is shown in the following figure:



2.3 Deployment View

2.4 Runtime View

[UC1] - Student Registration

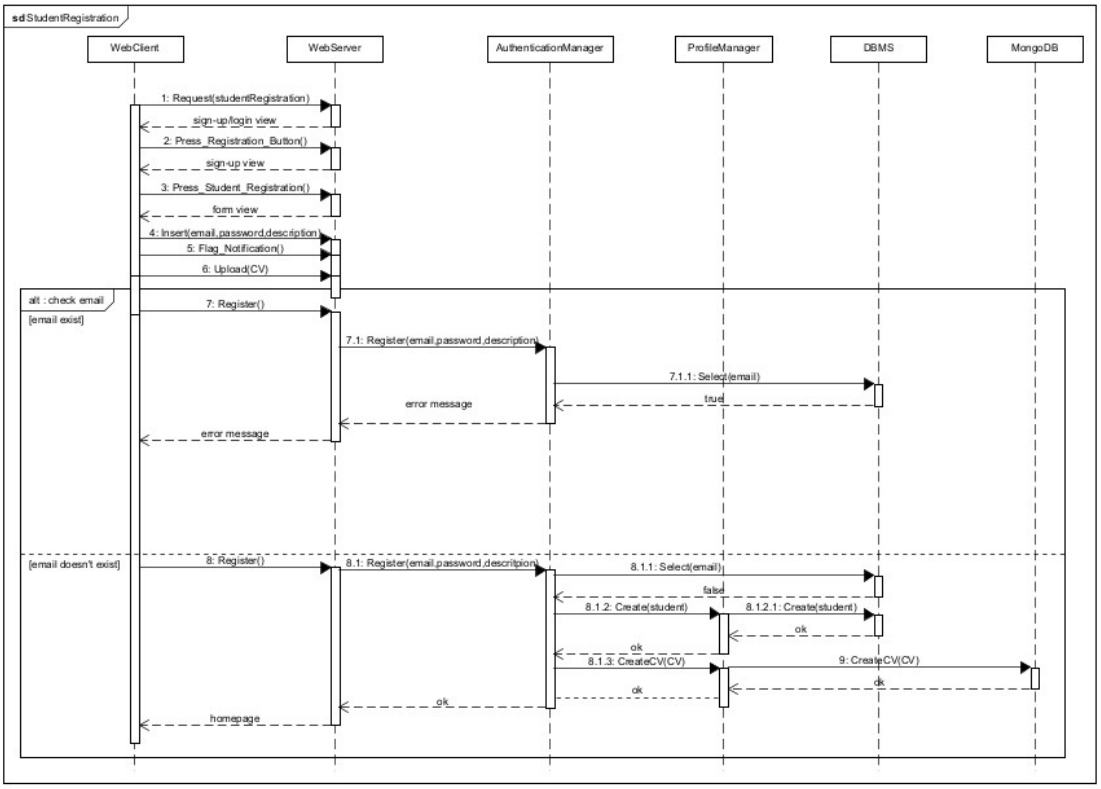


Figure 2.1: [UC1] - Student Registration

The figure shows the student registration process; the request is sent to Authentication Manager, which is also responsible for receiving the registration form completed by the student. Once received, Authentication Manager queries the DBMS to check if the student's email is already present in the database. If it is not present, the student will be saved in the database, while the CV will be stored in MongoDB as an unstructured data. If the email is already in the database, an error message will be displayed to the student.

[UC2] - Company Registration

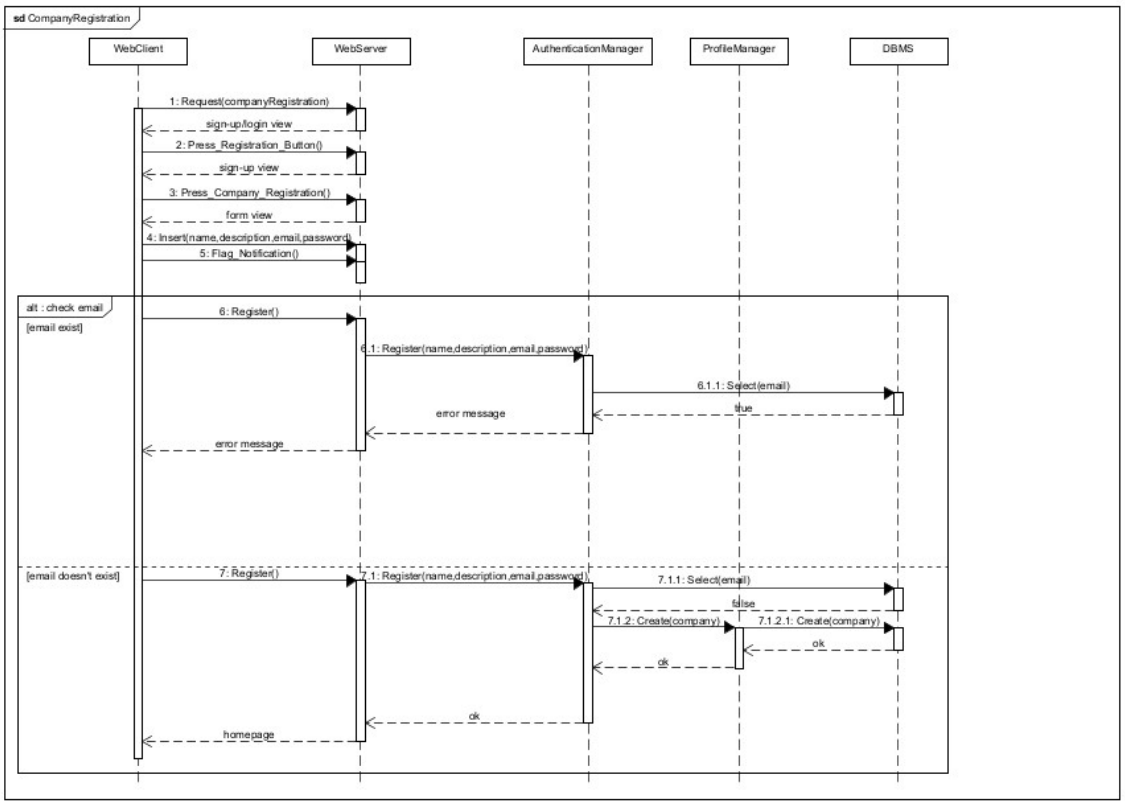


Figure 2.2: [UC2] - Company Registration

The figure shows the company registration process; the flow is similar to the student registration case, with the difference that in this case, different data is saved (the company has a name and a description), and all the data is stored in the relational database since they are all structured data.

[UC3] - User Login

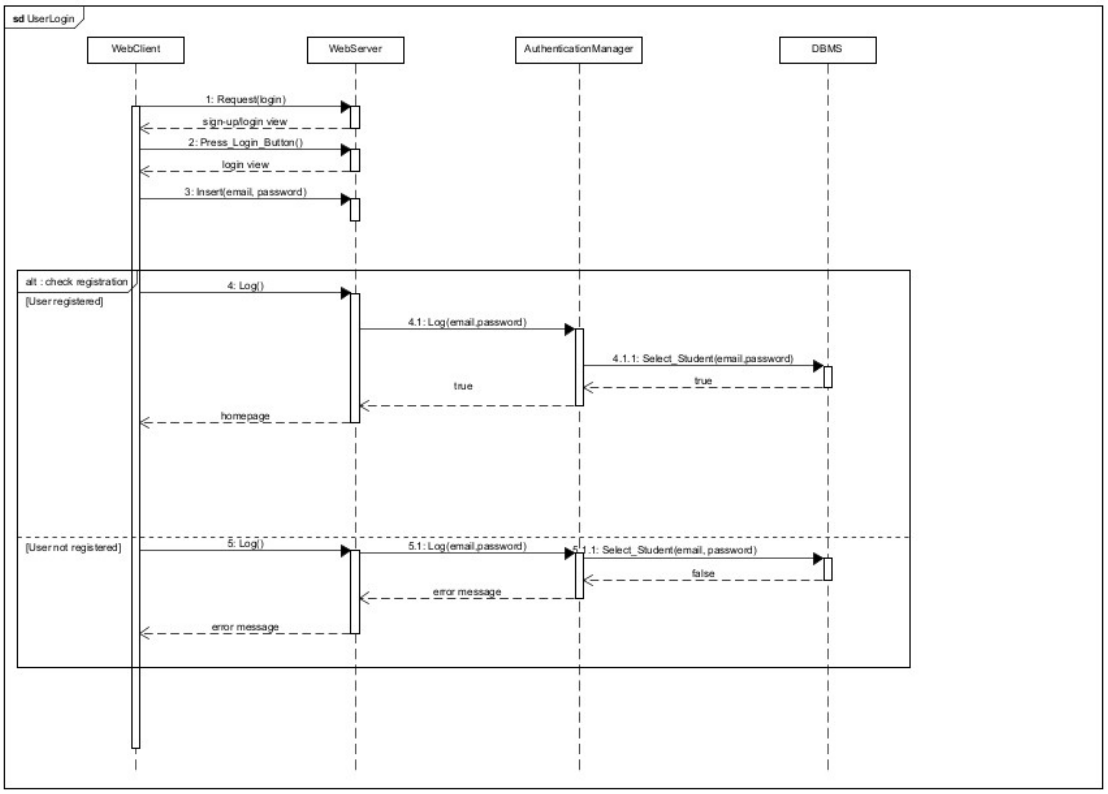


Figure 2.3: [UC3] - User Login

The figure shows the user login process, which is similar for both students and companies; in this case, the request is handled by the Authentication Manager, which selects the student from the database to allow login. If the student is not found, an error message is displayed.

2.5 Component Interfaces

2.6 Selected Architectural Styles and Patterns

3 User Interface Design

While in the RASD we defined platform's proper UI structure, in this section we formalize navigation maps and present mock-up for each page:

- public pages:
  - registration pages:
    - **WelcomePage**
    - **RegistrationMenuPage**
    - **StudentRegistrationPage**
    - **CompanyRegistrationPage**
  - logic pages:
    - **LoginPage**
- private pages:
  - profile pages:
    - **StudentProfilePage**
    - **CompanyProfilePage**
  - notifications pages:
    - **NotificationsPage**
  - companies pages:
    - **CompanySearchPage**
  - advice-related pages:
    - **AdviceSearchPage**
    - **PersonalAdvicePage**
    - **PublishAdvicePage**
    - **AdviceDetailsPage**
    - **InterestingAdvicePage**
  - invites pages:
    - **InvitesPage**
  - applications pages:
    - **ApplicationsPage**
  - process-related pages:
    - **ConfigProcessPage**
    - **ProcessManagementPage**
    - **StepsManagementPage**
    - **ViewStatsPage**
    - **ProcessFinalizationPage**
  - interviews pages:
    - **InterviewsPage** : questa ci si è dimenticati di metterla nel menu delle compagnie, quando si fa la revisione metterla;
  - internships pages:
    - **InternshipsPage**
    - **InternshipDetailsPage**

Moreover, in this chapter we also specified how the sections defined in the User Interfaces section of the RASD are mapped onto the sections effectively designed.

3.1 Public pages

3.1.1 Public pages navigation map

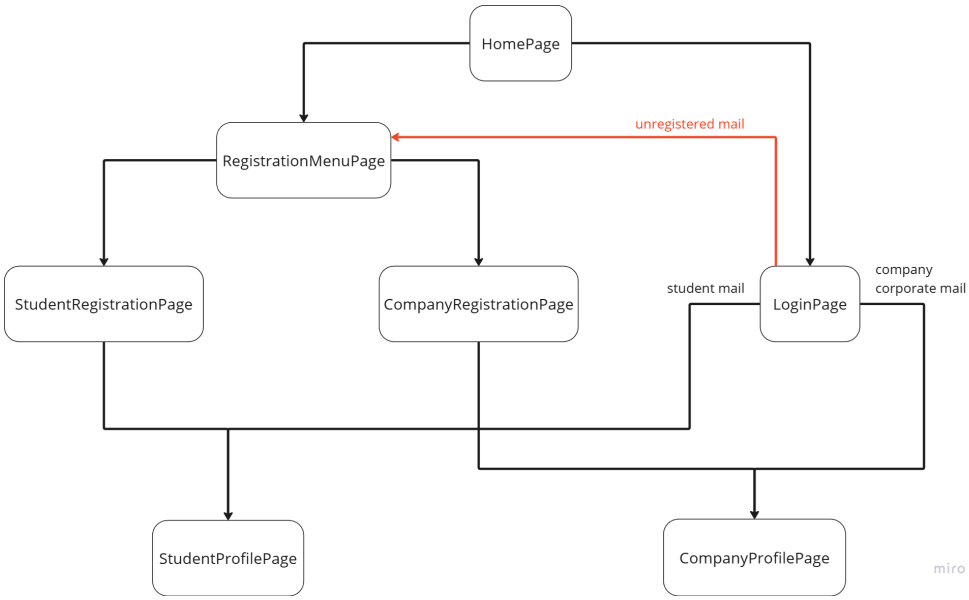


Figure 3.1: Public pages navigation map

3.1.2 Registration pages mock-up

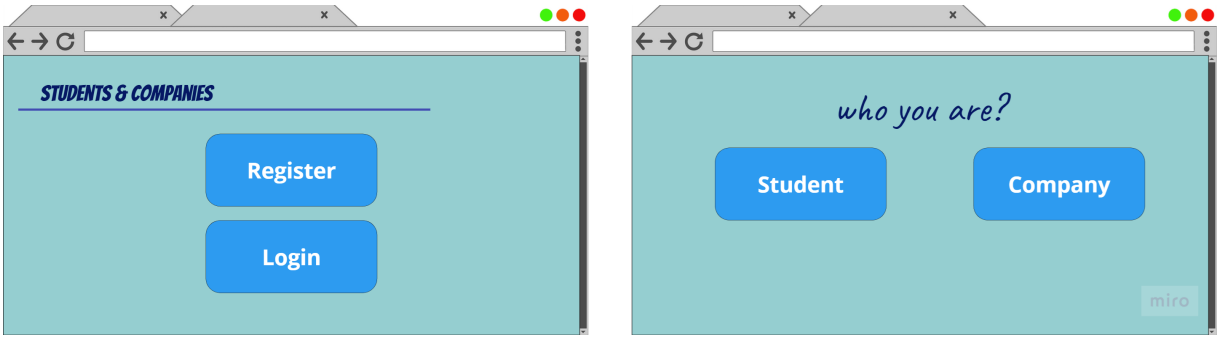


Figure 3.2: WelcomePage and RegistrationMenuPage mock-up

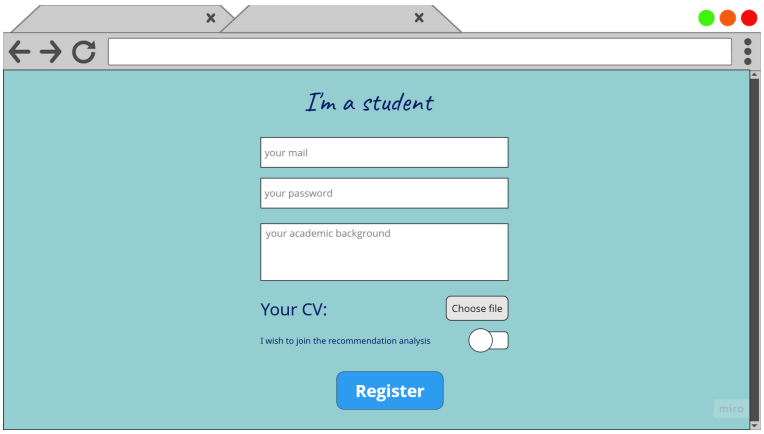


Figure 3.3: StudentRegistrationPage mock-up

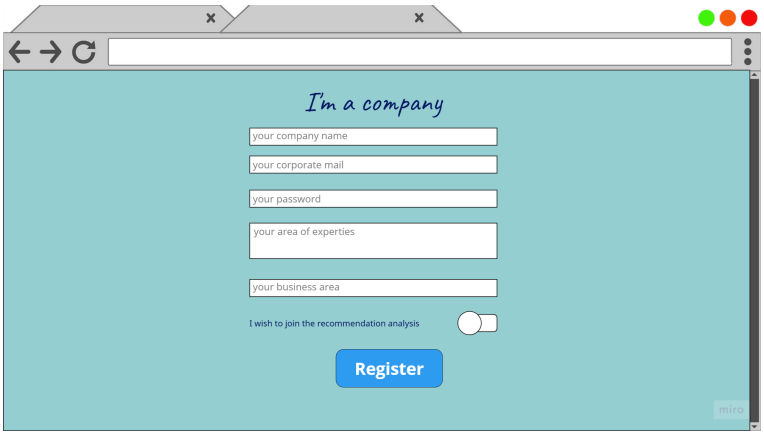


Figure 3.4: CompanyRegistrationPage mock-up

3.1.3 Login pages mock-up

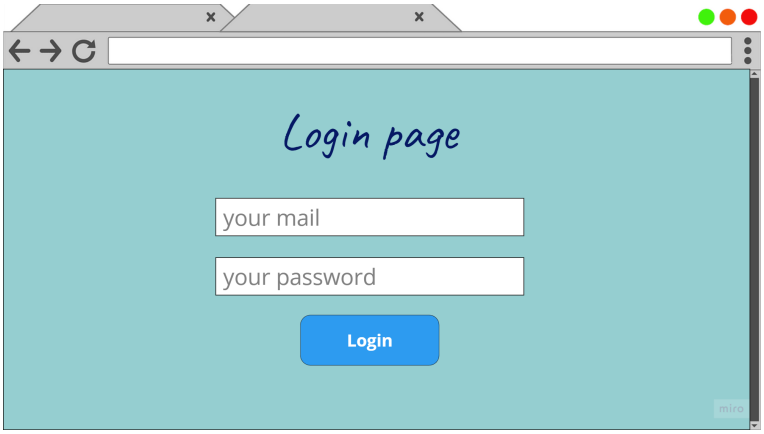


Figure 3.5: LoginPage mock-up

3.2 Private pages

Note that for what concerns user navigation in private pages:

- as mock-up show, private sections can be accessed by means of a side-bar (this explains why navigation maps are not connected graphs);
- navigation maps present the designated possibilities to navigate throw sections, while mock-ups often omit navigation buttons/links (e.g. InternshipDetailsPage → CompanyProfilePage). Therefore, to implement navigation hyperlinks always refer to maps.

3.2.1 Private pages students navigation map

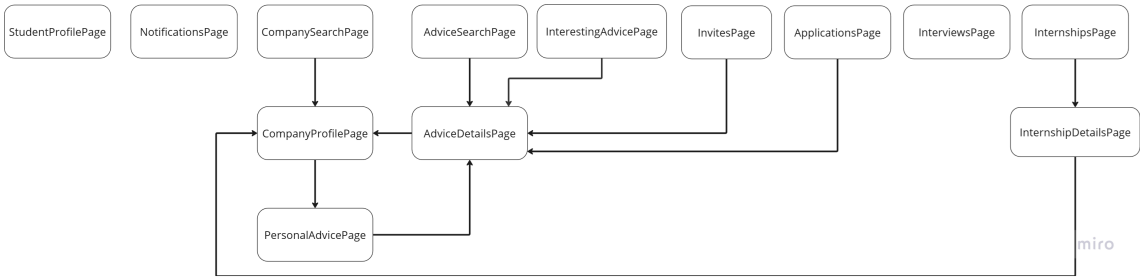


Figure 3.6: Private pages students navigation map

3.2.2 Private pages companies navigation map

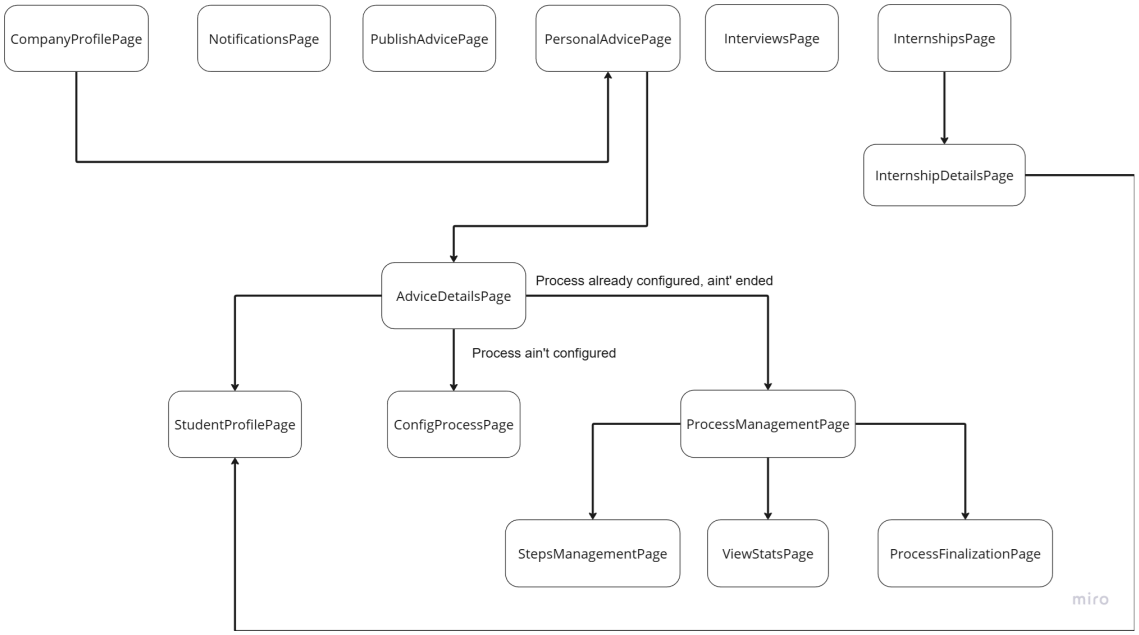


Figure 3.7: Private pages companies navigation map

3.2.3 Profile pages

3.2.3.1 Profile pages mock-up

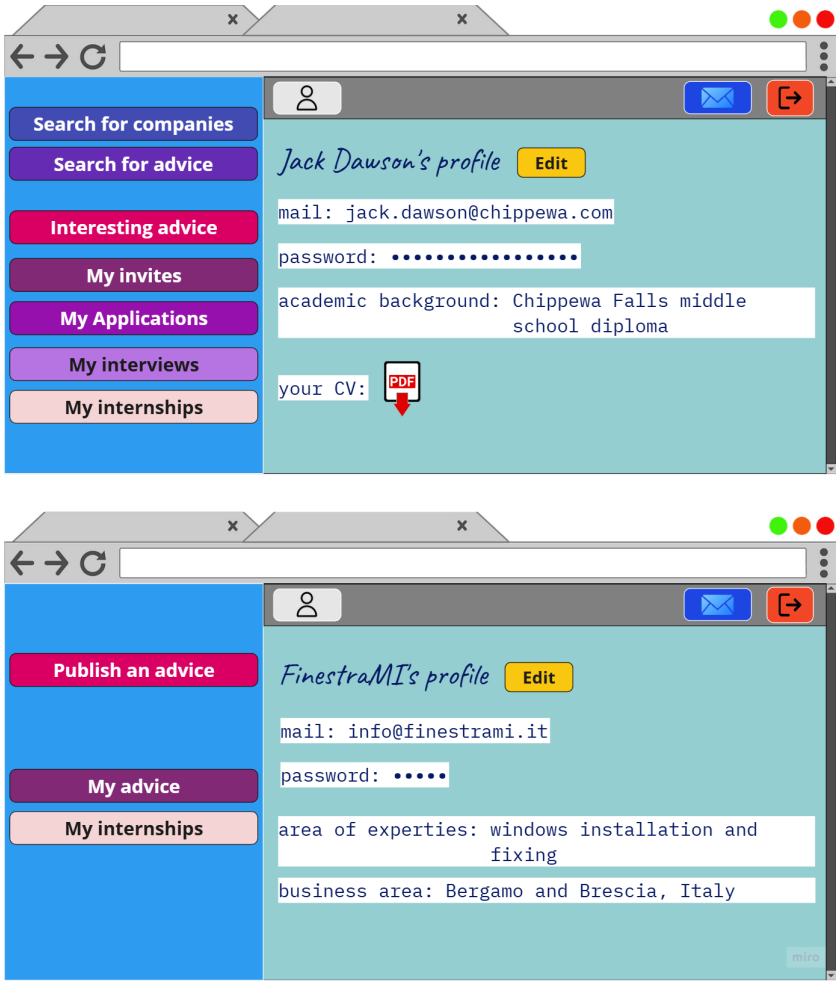


Figure 3.8: Profiles pages (students above, companies below) mock-up from the profile's owner view point

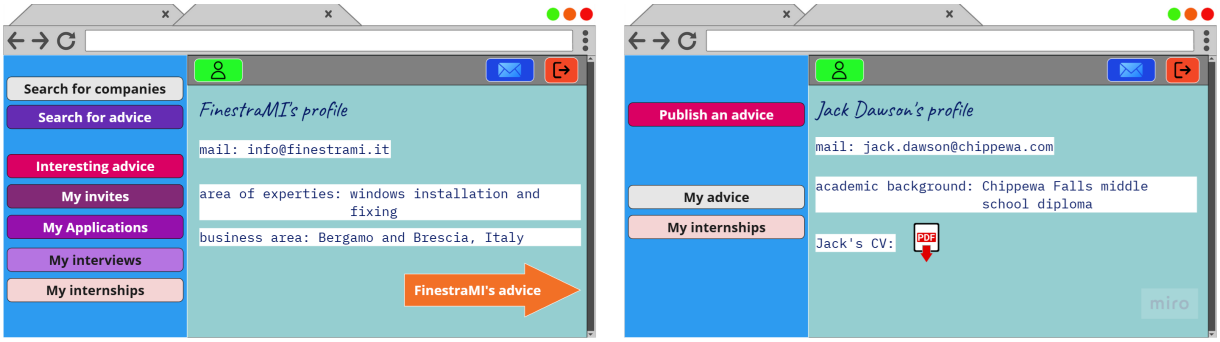


Figure 3.9: Profiles pages (students on the left, companies on the right) mock-up from other users view point

3.2.4 Notifications pages

3.2.4.1 Notifications pages mock-up

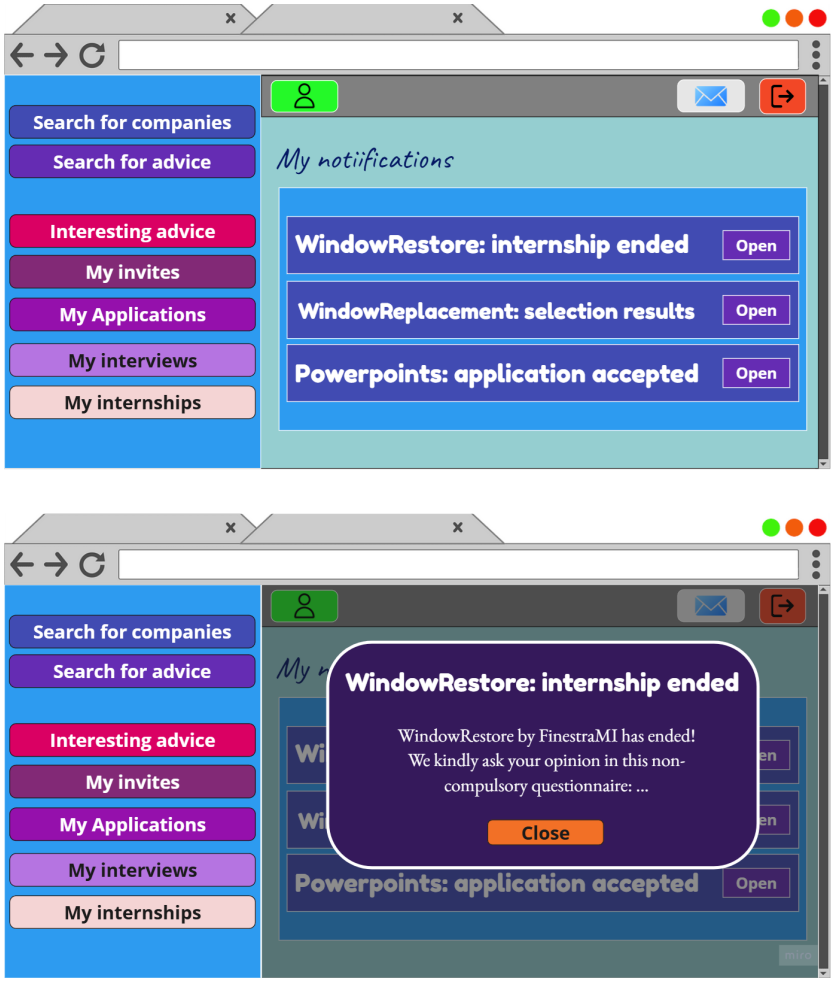


Figure 3.10: NotificationsPage and Notification modal box mock-up

Non-compulsory questionnaires are shown on a single window accessible from a link in the notification related to them:

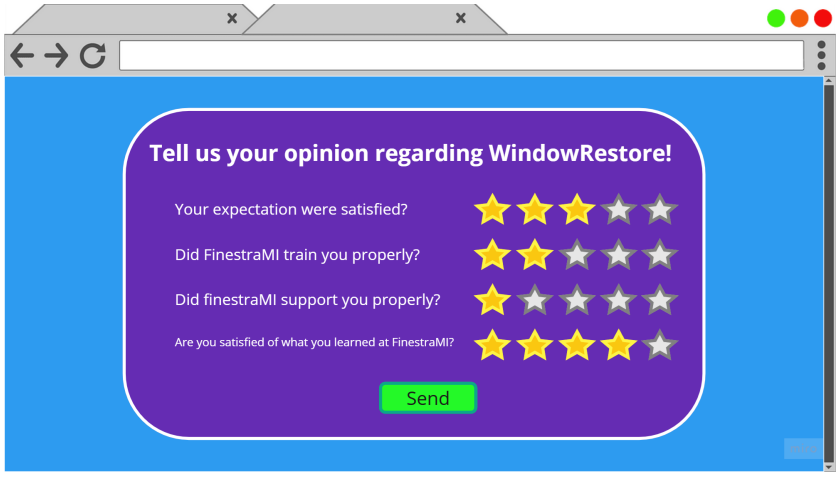


Figure 3.11: Feedback questionnaire window mock-up

3.2.5 Companies pages

3.2.5.1 Companies pages mock-up

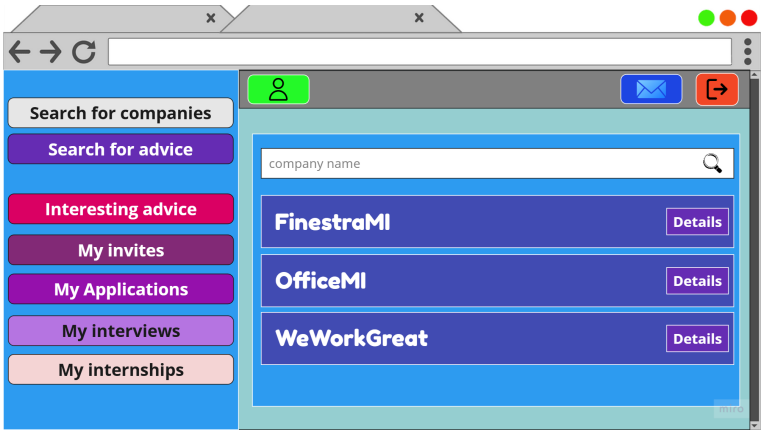


Figure 3.12: CompanySearchPage mock-up

3.2.6 Advice-related pages

3.2.6.1 Advice-related pages mock-up

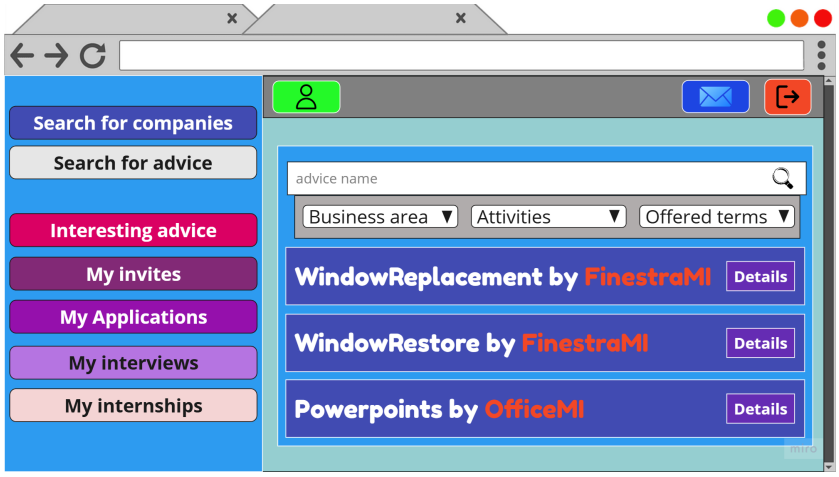


Figure 3.13: AdviceSearchPage mock-up

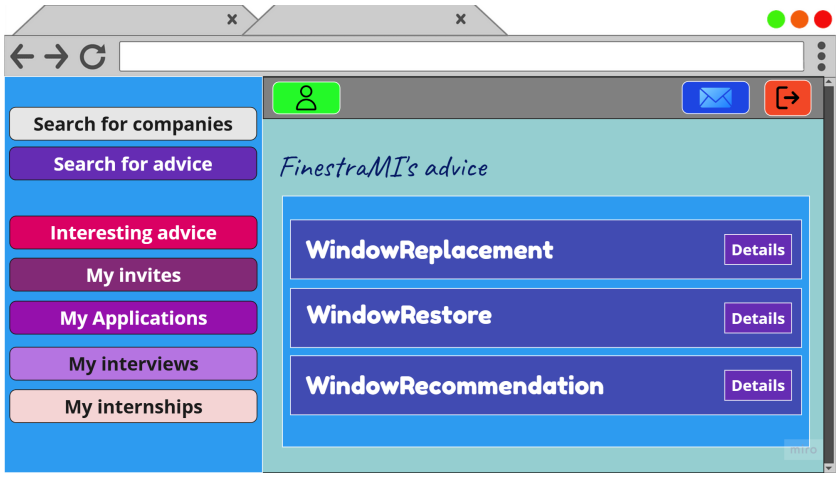


Figure 3.14: PersonalAdvicePage mock-up

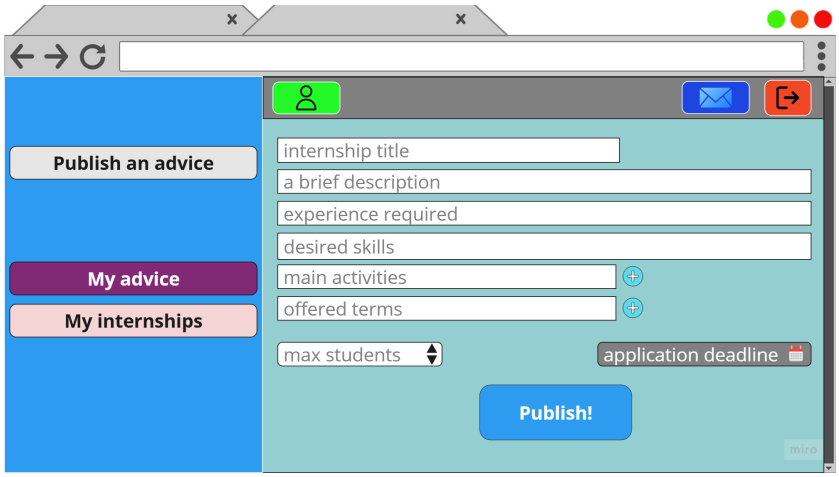


Figure 3.15: PublishAdvicePage mock-up

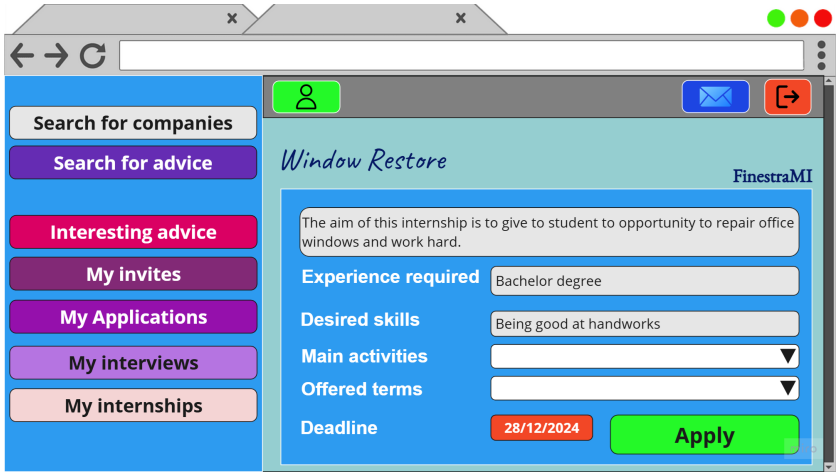


Figure 3.16: AdviceDetailsPage (students view-point) mock-up

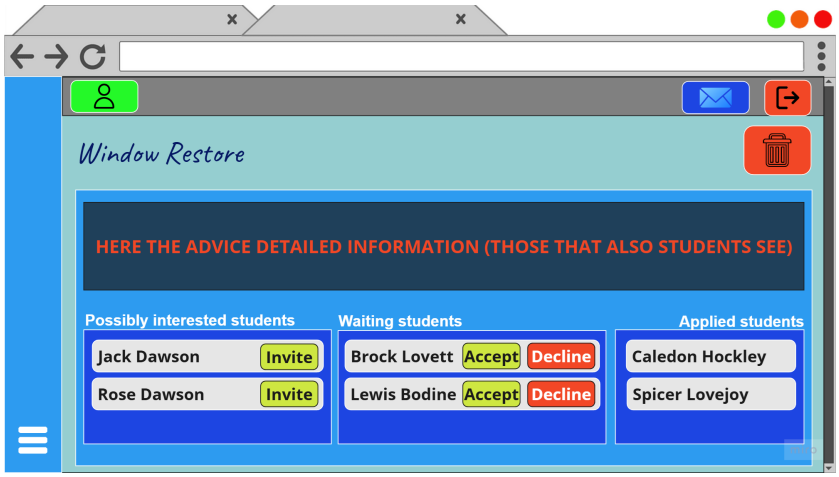


Figure 3.17: AdviceDetailsPage (company view-point) mock-up

The trash button will turn into a *Configure selection process* button once the advice deadline has last.

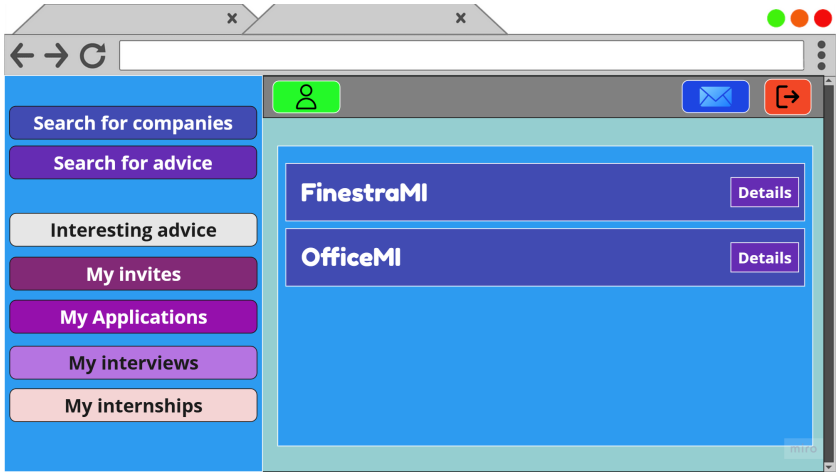


Figure 3.18: InterestingAdvicePage mock-up

3.2.7 Invites and Applications pages

3.2.7.1 Invites and Applications pages mock-up

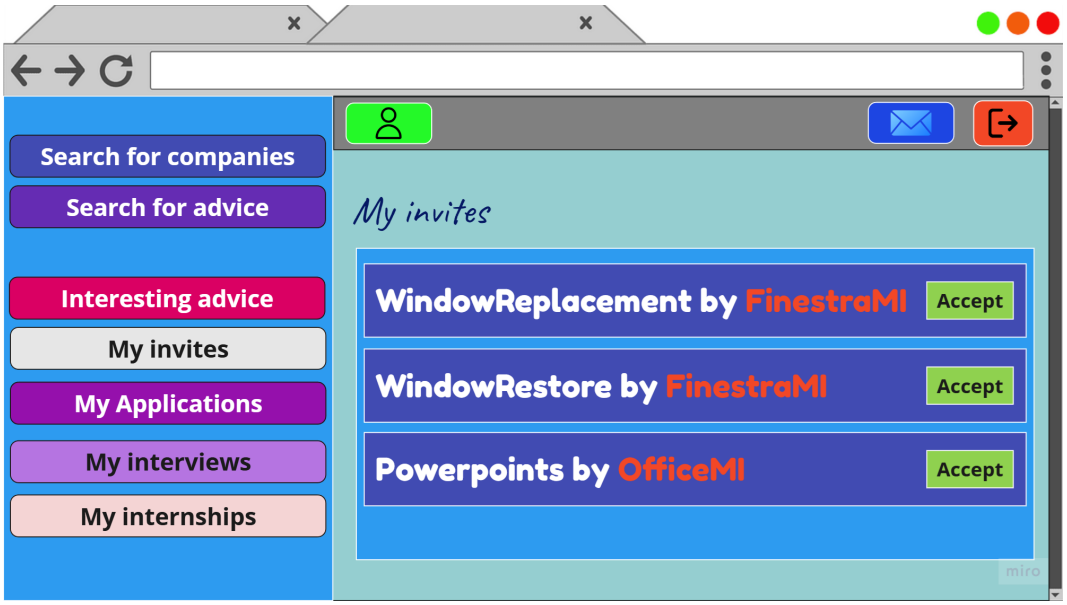


Figure 3.19: InvitesPage and ApplicationsPage mock-up

3.2.8 Process-related pages

3.2.8.1 Process-related pages mock-up

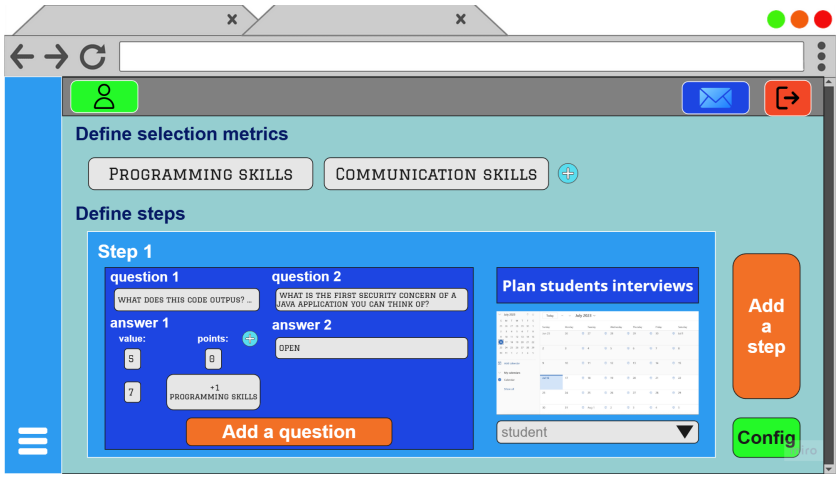


Figure 3.20: ConfigProcessPage mock-up

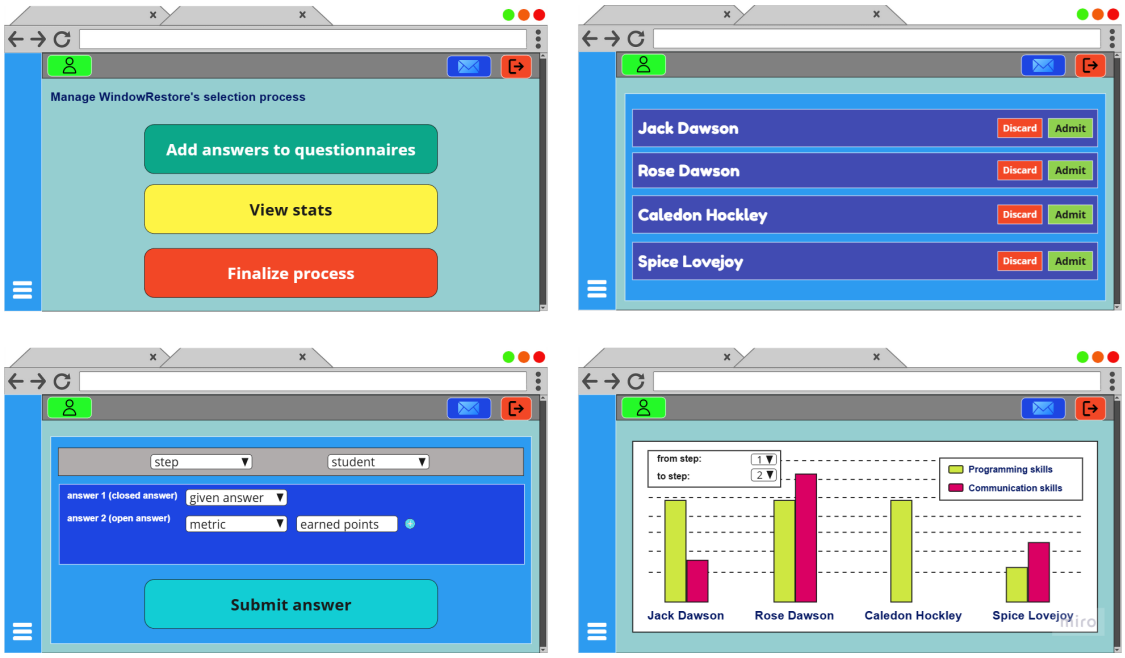


Figure 3.21: ProcessManagementPage, StepsManagementPage, ViewStatsPage and ProcessFinalizationPage mock-up

3.2.9 Interviews pages

3.2.9.1 Interviews pages mock-up

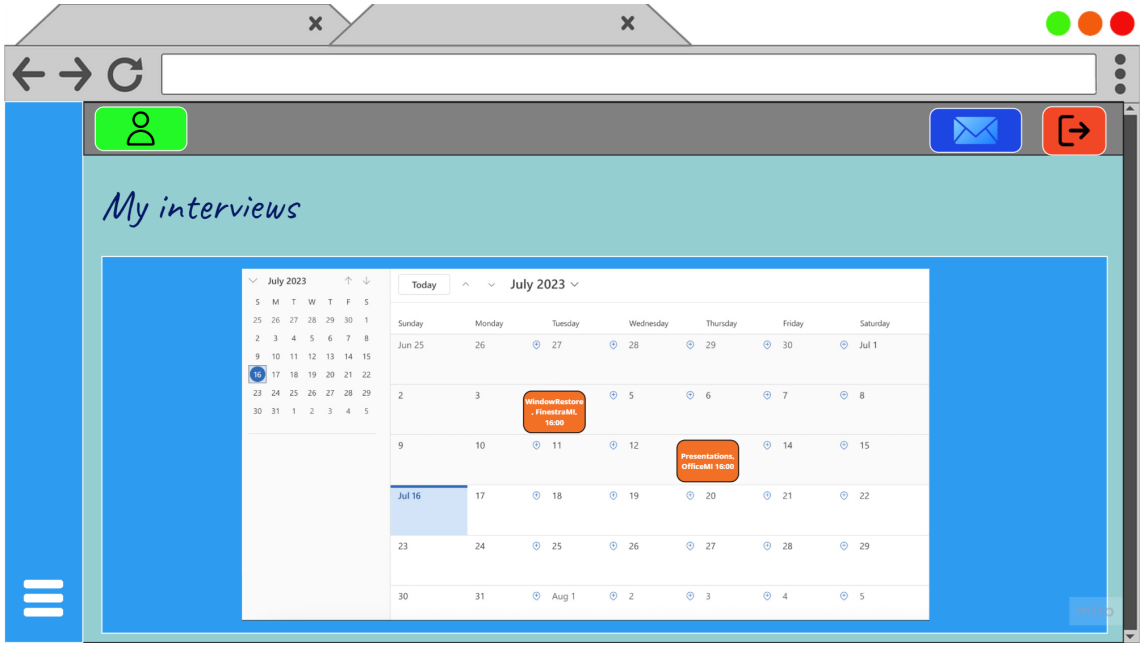


Figure 3.22: InterviewsPage mock-up

3.2.10 Internships pages

3.2.10.1 Internships pages mock-up

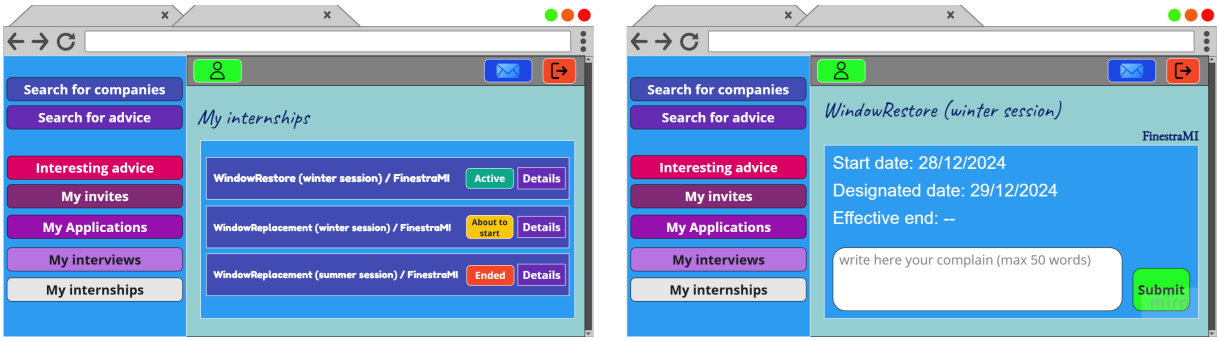


Figure 3.23: InternshipsPage and InternshipDetailsPage mock-up (from students view point)

In fact there is not much difference in what internships owners see. Companies can end instantly any ongoing internship that belongs to them.

3.3 Requirements UI sections mapping

4 Requirements traceability

5 Implemention, Integration and Test Plan

6 Effort Spent

Table 6.1: Effort spent overview

	Andrea		Alessandro	
Week	**Hours**	**Category**	**Hours**	**Category**
23/12/2024 - 29/12/2024	16	User interface and high-level structure	4	high-level structure
30/12/2024 - 05/01/2025	2	Data logic model	12	Introduction, data logic model, sequence diagrams
06/01/2025 - 07/01/2025	8	Final revision, requirements mapping, component view, deployment view, effort spent and software used compilation	8	Final revision, requirements mapping, integration and test plan, component view, integration and test plan
TOTAL	**26**		**24**	

Software Used 7

Table 7.1: Software used overview

Software	**Version**	**Usage**
TeXstudio	4.6.3	Writing Latex
PdfLaTeX	-	Latex compilation
XeLaTex	-	Latex compilation
Astah UML	9.2.0/0248cd	UML diagrams
Alloy Analyzer	6.1.0	Alloy modeling
miro.com	-	User interface maps and mock-up, high-level view, logic scheme
draw.io	-	Component view, deployment view

This document was written over the template kaobook designed by Federico Marotta (https://github.com/fmarotta/kaobook), with few adjustments by us.