Software Engineering 2 - Computer Science and Engineering

Design Document

Students&Companies

Authors:
Alessandro Vitobello - 10710941
Jurij Diego Scandola - 10709931
Francesco Raimondi - 10767901

Academic Year:
2024 - 2025

Index

1 Introduction

1.1 Purpose

The purpose of this document is to present a complete and detailed design description of the S&C system, by illustrating each major design decision made for the system.
The document contains a complete overview of the system, from the general architecture layout to the design of each specific component and their interfaces. The document will also contain information concerning the physical deployment of the components and a presentation of some of the user graphical interfaces. A discussion on the Implementation, Integration and Testing plan will also be addressed at the end of the document. This Design Document will also provide a detailed description showing how the requirements and use cases discussed in the Requirement Analysis and Specification Document, are implemented by the system.
This document is aimed for the software developers and testers in order to provide them a road-map of the project.

1.2 Scope

S&C aims to address the specific needs of students seeking internship opportunities, companies looking to offer them, and universities that wish to remain actively involved in the process. Students can use the platform to apply more easily for internships, while companies are supported during the selection and interview phases. The platform also promotes a mutual exchange of feedback—potentially anonymous—focused on improving the experience for both parties. Universities maintain an important role: based on this feedback, they can step in to address issues that arise and, if necessary, terminate collaborations that are not working effectively.

1.3 Definitions Acronyms, Abbreviations

1.3.1 Definitions

Recommendation: used to refer to the mechanism used by the system to inform students that a possible interesting internship became available and to inform companies about the availability of a student's CV matching their needs.

User: we refer to user as any of the S&C actors between Student, University and Company.

Platform: refers to S&C's online information system that facilitates the connection between students and companies to create internship opportunities.

Match: the result of the recommendation process in which a potential connection between a student and an internship is identified.

Feedback: qualitative or quantitative information provided by students and companies regarding their experience with the platform, the selection processes, or the internship.

1.3.2 Acronyms

S&C: stands for Student&Companies, the platform's name.

CV: curriculum vitae; a written document where all the educational and professional experiences of an individual and his skills are listed (sometimes including also his personal interests).

UI: user interface.

DB: database.

1.4 Revision History

Version 1.0: 5/01/2025
Version 2.0: 7/01/2025

1.5 Reference Documents

- Requirement Analysis and Specification Document Students&Companies (Vitobello - Scandola - Raimondi)
- CreatingDD.pdf (WeBeep.polimi.it)

1.6 Document Structure

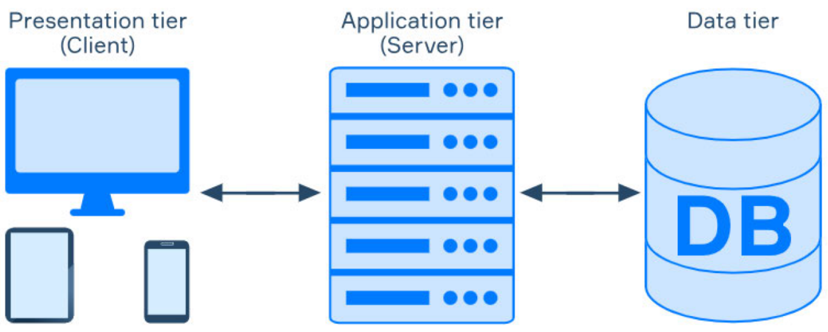
This document is composed of a total of six sections (apart of the Introduction section), detailed below. The Architectural Design section includes an overview on the System and some detailed views (Component, Deployment and Runtime views). Furthermore we also present the architectural styles used while design the application.
The following section includes all the User Interface mockups we produced, helping the reader to grasp an idea on how the user will interact with the platform. After this, the requirements mapping section in presented where all the requirements defined in the RASD are mapped to the design components we listed.

The Implementation, Integration and Test plan section include all the information about how the application will be developed and tested.
In section five a table with the team efforts is presented.
Section six contains the references used in the document.

2 Architectural Design

2.1 Overview: High-level components and their interaction

The architecture of S&C follows the three-tier pattern with a presentation layer, business logic layer and a data layer. The decision to use a three tier architecture was made in order to separate the business logic of the system from the data. The data in itself could reasonably be used for other applications in the future, and so decoupling them facilitates that. Moreover it provides an additional layer of safety as all data access will go through the middle layer. Below, there is first a short description of every tier, and then a graphical representation of the architecture.



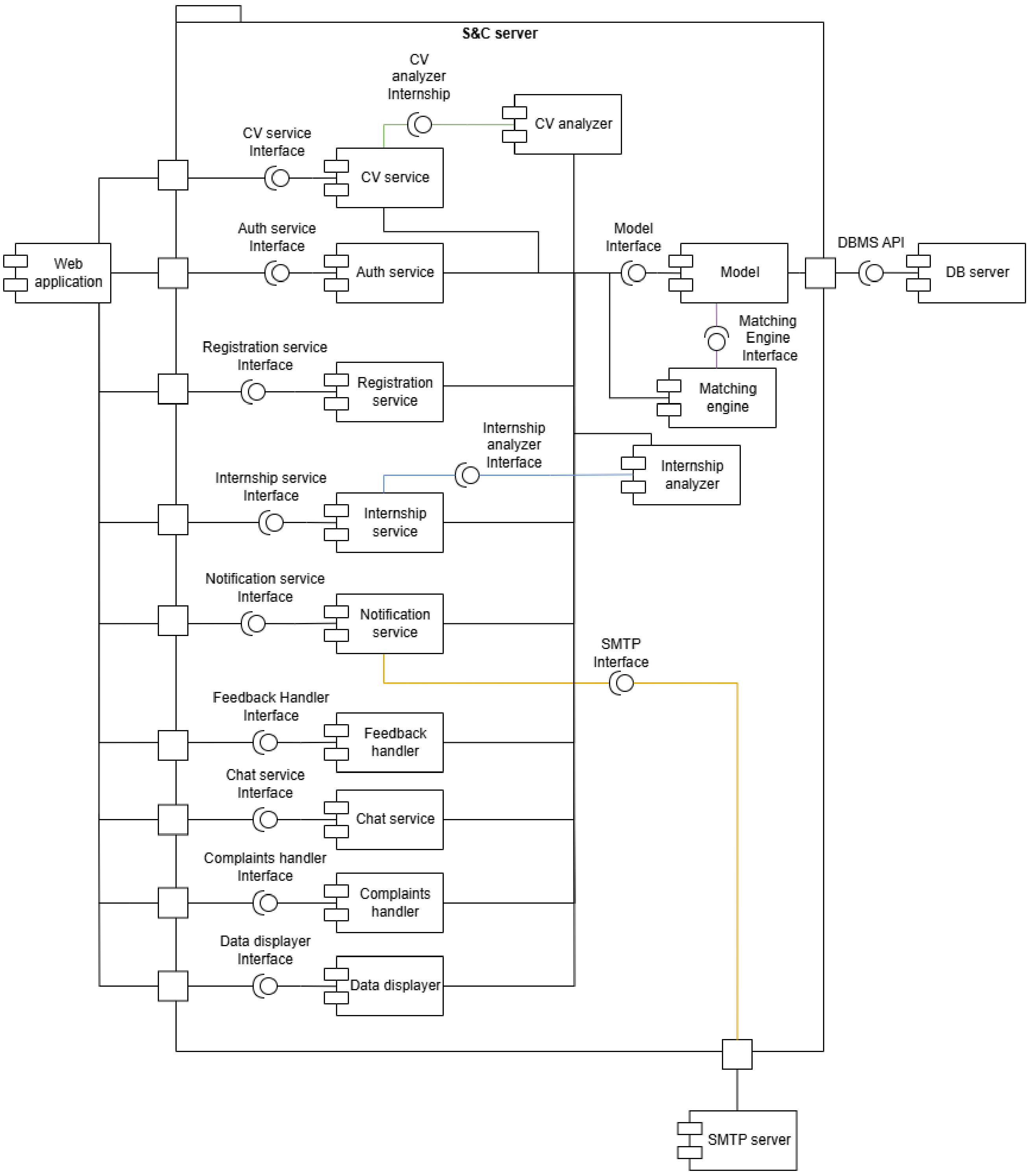
Presentation
The presentation tier handles the interaction with the user, by communicating with the business tier and presenting the information retrieved. The idea is that this layer will be a simple website and it will communicate only with the server containing the business logic.

Business logic
This tier is the core of the application's functionality. It includes all business logic, such as the matching engine. As the intermediary layer, it facilitates the transfer and translation of data between the data tier and the presentation tier.

Data
The data tier handles the storage, retrieval, and management of the system's persistent data. It acts as the foundation for the other tiers, ensuring that the information required for calculations, analytics, and user interaction is both accurate and accessible. Robust data management within this tier guarantees system reliability and supports complex queries and reporting needs.

2.2 Component view

Here is a visualization of all the server components.



Let's briefly describe them:

Web Application: the application used by the user on his device to connect to S&C.

CV service: component that enables all the functions connected to the CV: uploading and creation.

CV analyzer: it analyzes the uploaded CV checking if the descriptions are well formed, it also extrapolates useful information such as skills, soft skills, student location and so on to advice the student if he should be including other information.

Internship service: component that let the company representative compose the internship form and then sends it to the model.

Internship analyzer: when a company is uploading a new internship post, this component analyze what has been inserted by the user and eventually suggest improvements on: how to express some phrases, what to emphasize and what other information should be included.

Registration Service: allows each user to create his/her own profile on the platform.

Auth Service: allows each user to login his/her personal area.

Notification service: it is responsible for all the notifications the platform sends to its users whenever something relevant occurs (such as a change of the internship state, an interruption event, new recommendations, etc.).

Matching engine: software that allows S&C platform to find relevant match between students and companies based on skills, location and so on.

Complaints handler: it's the module responsible to end a student internship.

Data displayer: component responsible to request data from the model whenever an user requires them.

Feedback handler: it receives feedback from students and companies about the internship.

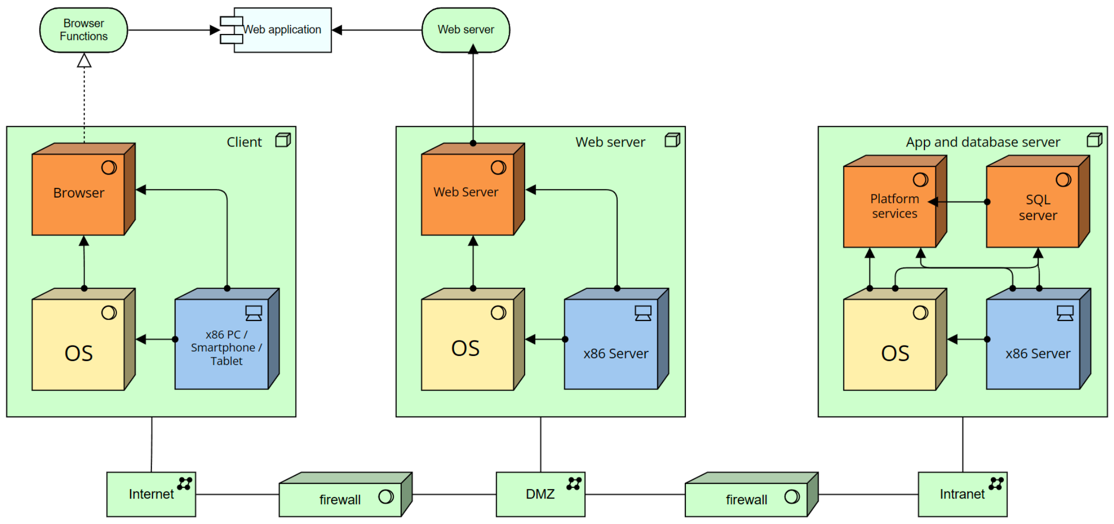
Chat service: component that allows user to directly interact on the platform. It is connected to the model to store chat history.

Model: it is the bridge between the DBMS and all the other components. It verifies that data are corrected before adding/removing to/from the database. Before passing the data back to the module it verifies if the result is different from null, if not it signal an error.

DBMS: memory storage where all S&C data are stored.

SMTP server: server that is connected to the notification service that sends users important notification via email.

2.3 Deployment view



The deployment diagram above illustrates the main components of our architecture. Client nodes communicate with the web application using the HTTPS protocol for secure communication, while internal communications within the intranet are based on the TCP/IP protocol.

2.3.1 Client Node

The client node represents the end-user device, such as a computer, smartphone, or tablet, that interacts with the application via a web browser. This node sends requests to the system and receives responses, allowing the user to interact with the web application.

2.3.2 Web server node

The web server node hosts the web application, which is responsible for processing incoming requests from clients. This node manages the application logic, interacts with the database, and returns responses to the client. It serves as the intermediary between the client and the back-end services. The web application node is connected to the DBMS/application server node through an internal network (intranet), ensuring reliable and fast communication. Placing the web application on a separate node helps maintain a clear separation of concerns between the client interface and the back-end services. This setup enhances scalability and security while optimizing resource allocation for handling incoming user requests. In particular, a certain standard of security is granted by the utilization of multiple firewalls and a DMZ.

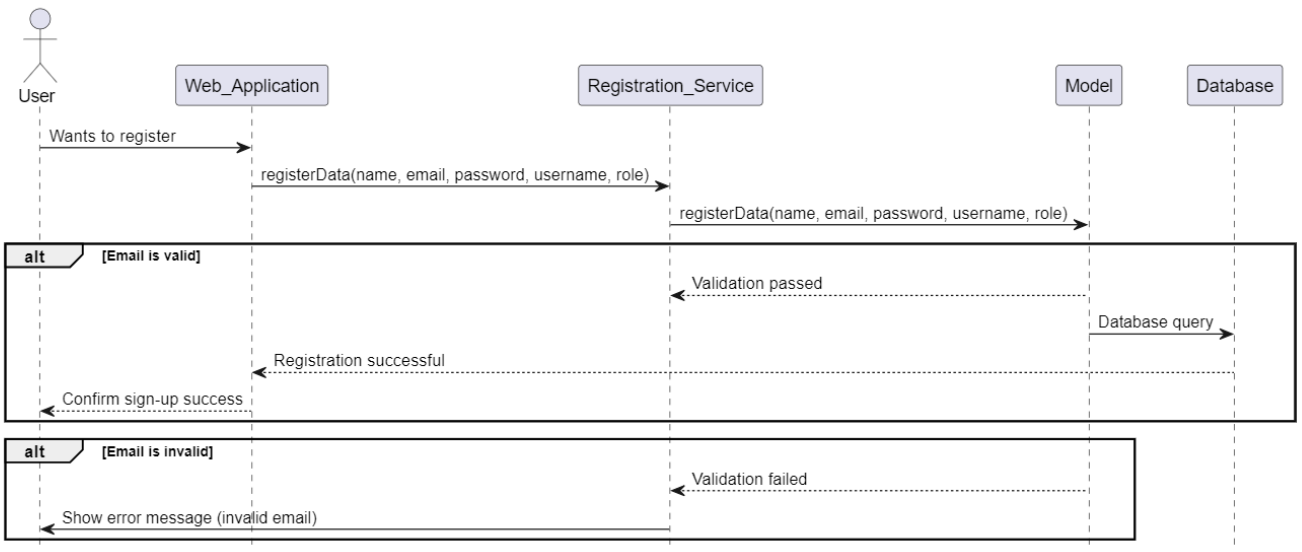
2.3.3 Application and database node

The DBMS/application server node hosts both the Database Management System (DBMS) and the application server. This architectural choice means that the database and application logic reside on the same server, which can improve performance by reducing the need for complex network communication between separate servers. In this node, the DBMS manages data persistence, while the application server handles business logic and database queries. This integration reduces the overhead of communication between separate systems, providing more efficient data access. Combining the DBMS and application server on a single node simplifies the architecture and improves performance by reducing the latency between the application logic and the database.

2.4 Runtime view

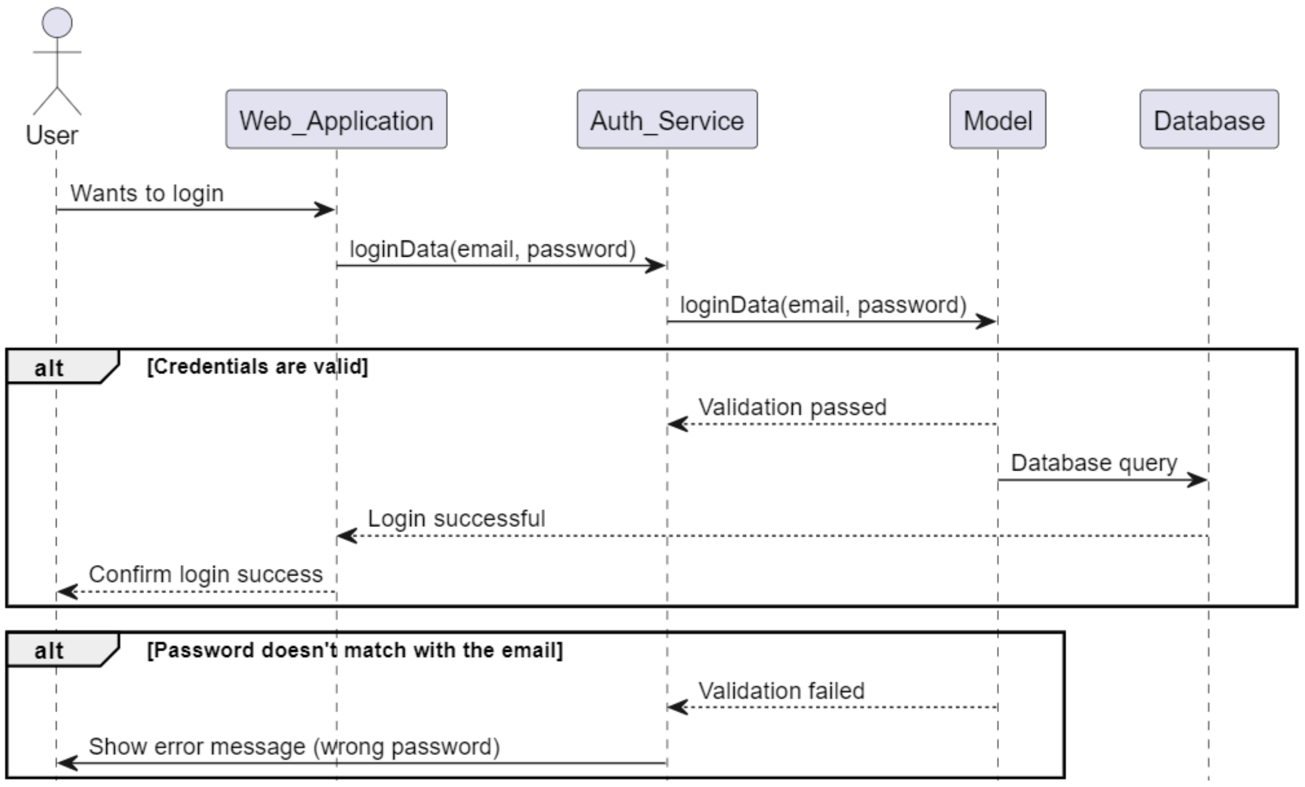
In this section we use sequence diagrams to describe the way components interact between them and with the Database. For all sequence diagrams, except for Registration and Login, it is assumed that the user is already logged in.

2.4.1 User registration



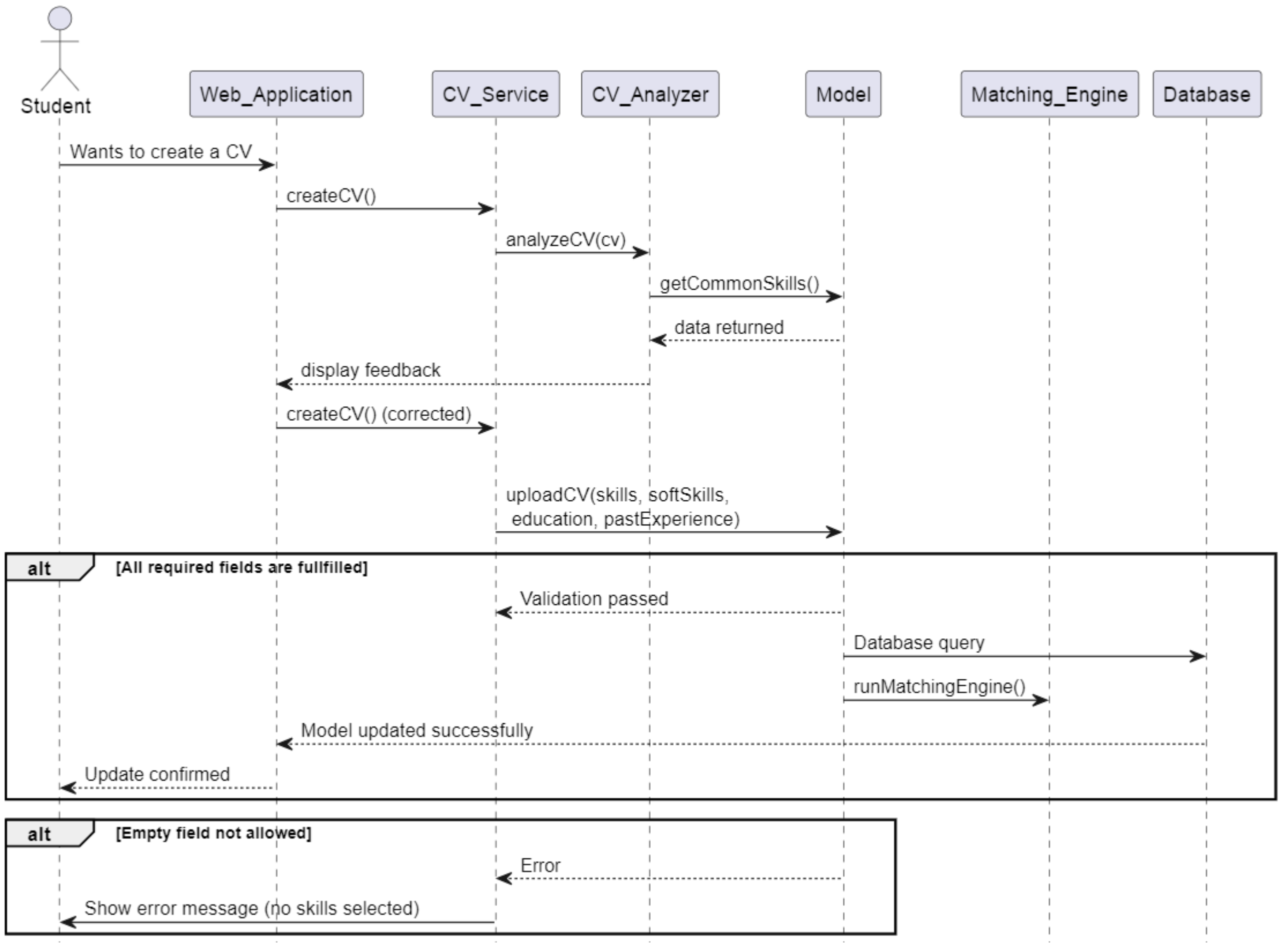
When a user wants to register, he inserts his data through the web application and, if everything is validated, his information are saved in the Database.

2.4.2 User login



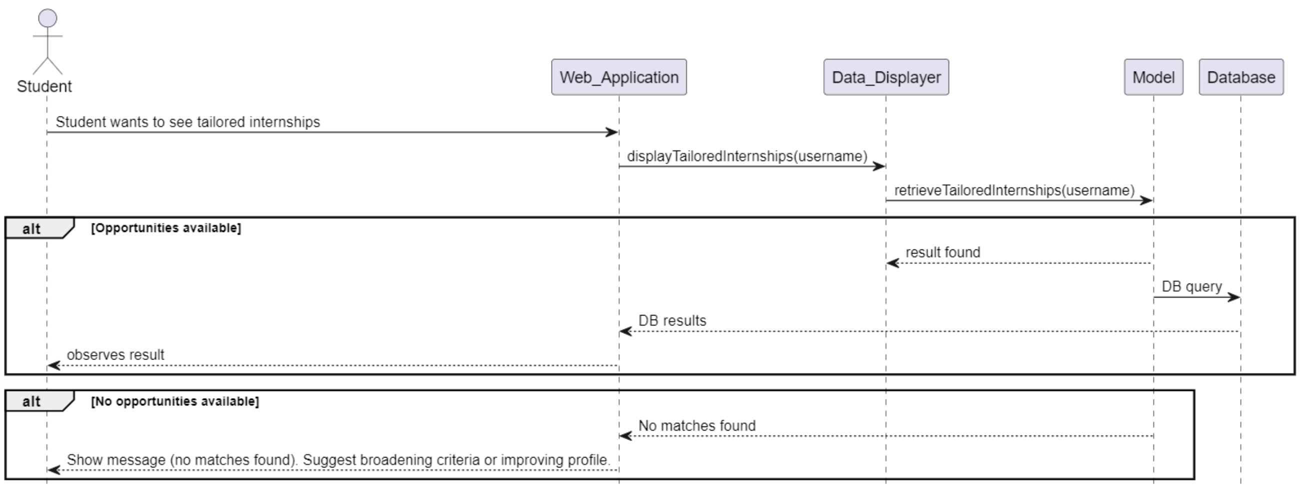
When an already registered user wants to login, he has to insert an email and the password associated with his account. If correct, verified through a db query, the login procedure is successful.

2.4.3 Student creates his/her CV on the platform



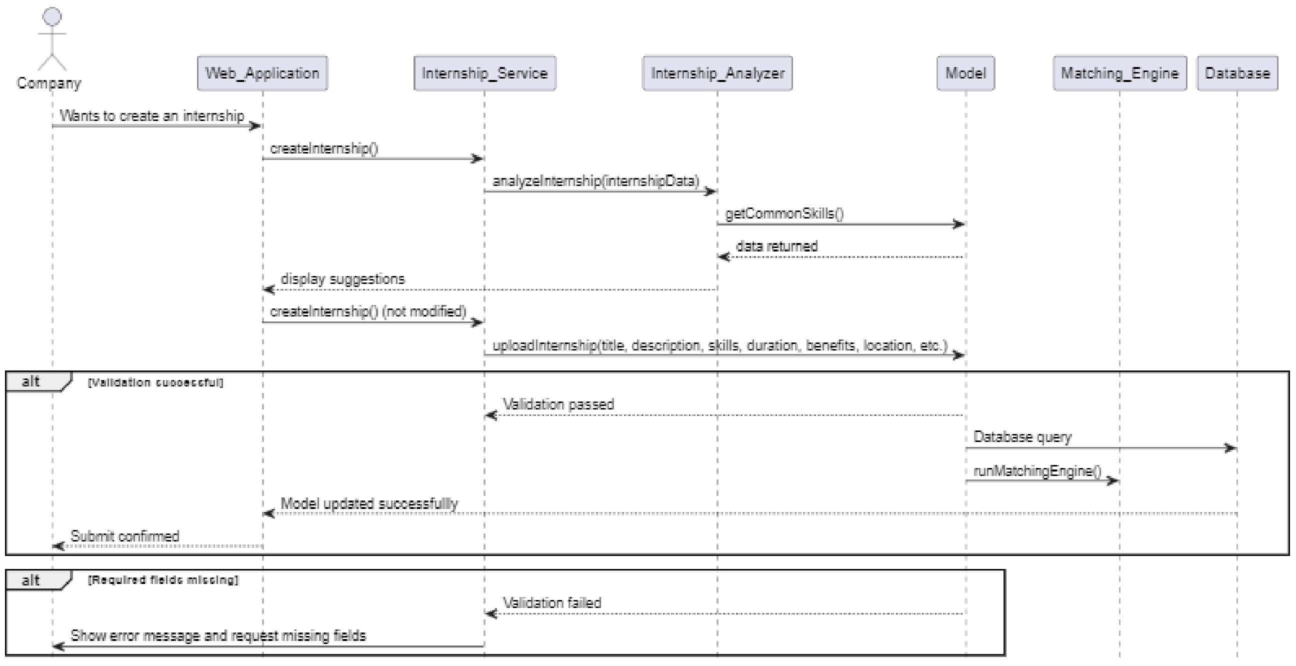
In this runtime view, we show a student that accepts a suggestion made by the CV Analyzer component, for example, verifying if he inserted the most common skills, that are required in many internships post, in his CV. It is also worth mentioning how the matching engine component works. It is called by the model after it was verified that the data was valid: the engine analyzes CVs skills and internships required skills to find possible matches; then, it calls back the model in order to update the database. This last step is not displayed here because it is not the purpose of this specific runtime view.

2.4.4 Student observes tailored internships


Student
Web_Application
Data_Displayer
Model
Database
Student wants to see tailored internships
displayTailoredInternships(username)
retrieveTailoredInternships(username)
alt [Opportunities available]
result found
DB query
DB results
observes result
alt [No opportunities available]
No matches found
Show message (no matches found). Suggest broadening criteria or improving profile.

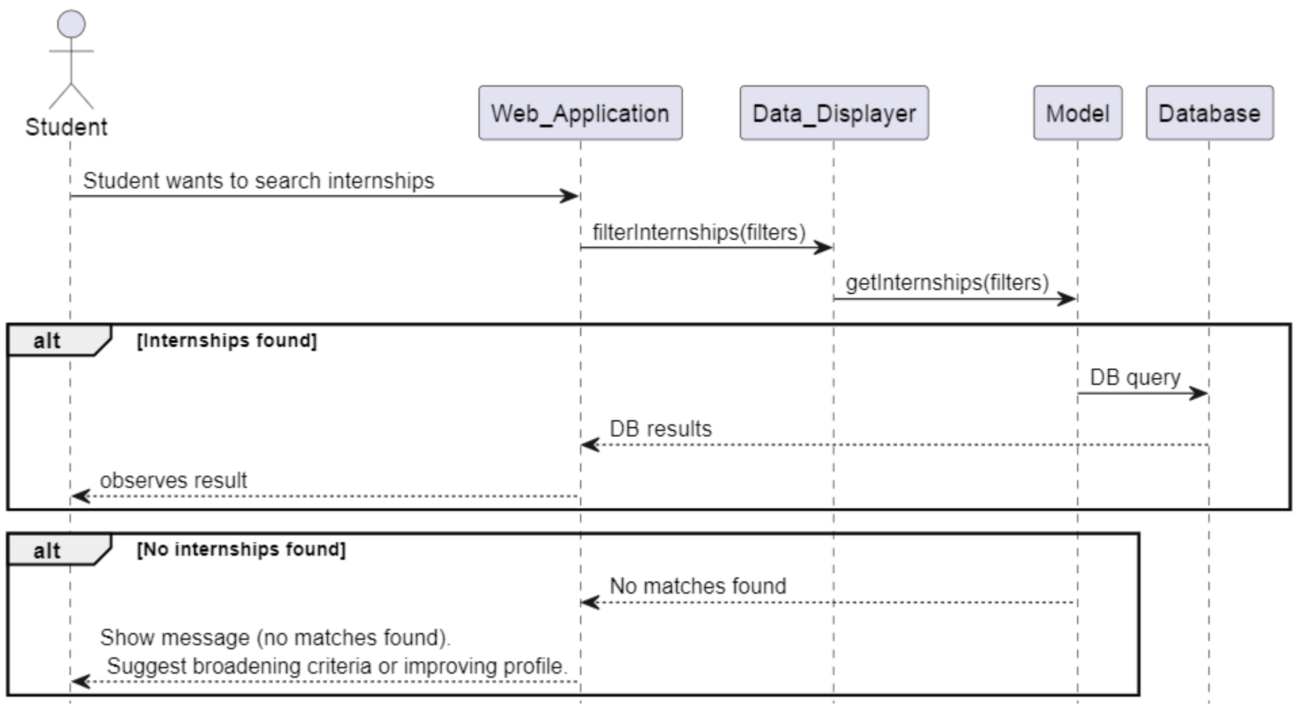
The data displayer component absolve its function, showing the user only the internship that were found to suit the most to him.

2.4.5 Company uploads an internship announcement


Company
Web_Application
Internship_Service
Internship_Analyzer
Model
Matching_Engine
Database
Wants to create an internship
createInternship()
analyzeInternship(internshipData)
getCommonSkills()
data returned
display suggestions
createInternship() (not modified)
uploadInternship(title, description, skills, duration, benefits, location, etc.)
alt [Validation successful]
Validation passed
Database query
runMatchingEngine()
Model updated successfully
Submit confirmed
alt [Required fields missing]
Validation failed
Show error message and request missing fields

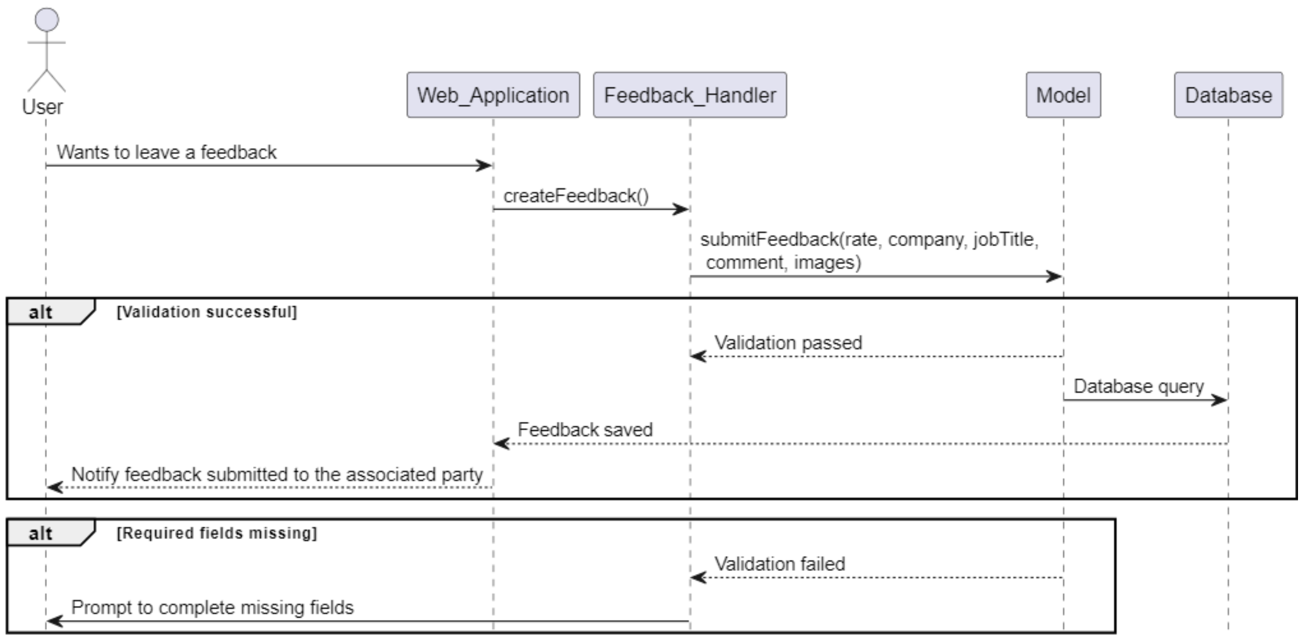
As in the previous runtime view dedicated to a student uploading his CV, here it is possible to see how the Internship Analyzer works, inspecting the data inserted by the company representative and eventually suggesting improvements. When the post is then finally uploaded, the matching engine works to find new suitable matched between students and companies.

2.4.6 Student searches for an internship through the platform



This is the runtime view associated to the possibility that has a student to pro-actively search through the platform for internships he may like. This is done for example inserting parameters such as location, duration and so on. The data displayer interact with the model (and then with the DB) to extract them.

2.4.7 Student or Company gives feedback

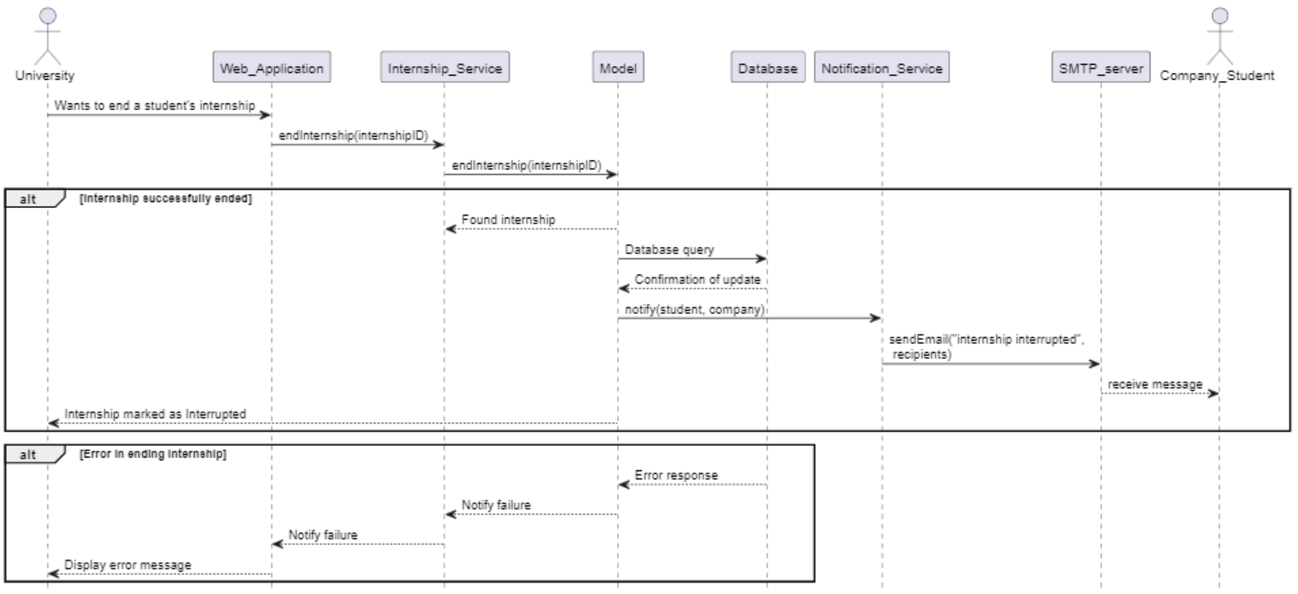


User
Web_Application
Feedback_Handler
Model
Database
Wants to leave a feedback
createFeedback()
submitFeedback(rate, company, jobTitle, comment, images)
alt
[Validation successful]
Validation passed
Database query
Feedback saved
Notify feedback submitted to the associated party
alt
[Required fields missing]
Validation failed
Prompt to complete missing fields

When an internship experience ended (gracefully or not) it is required to both the student and the company a feedback to improve the matching algorithm.

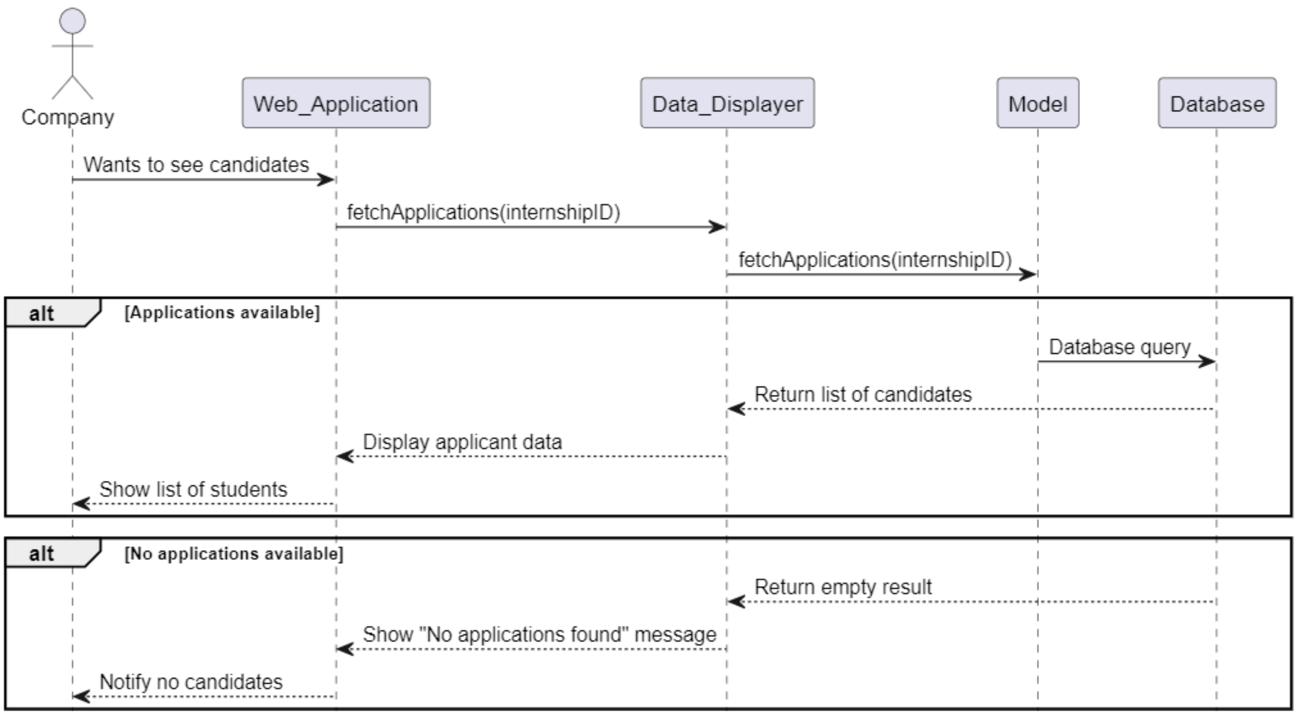
The form is completed through the web application and passed to the Feedback handler component.

2.4.8 University manages one of its student internship



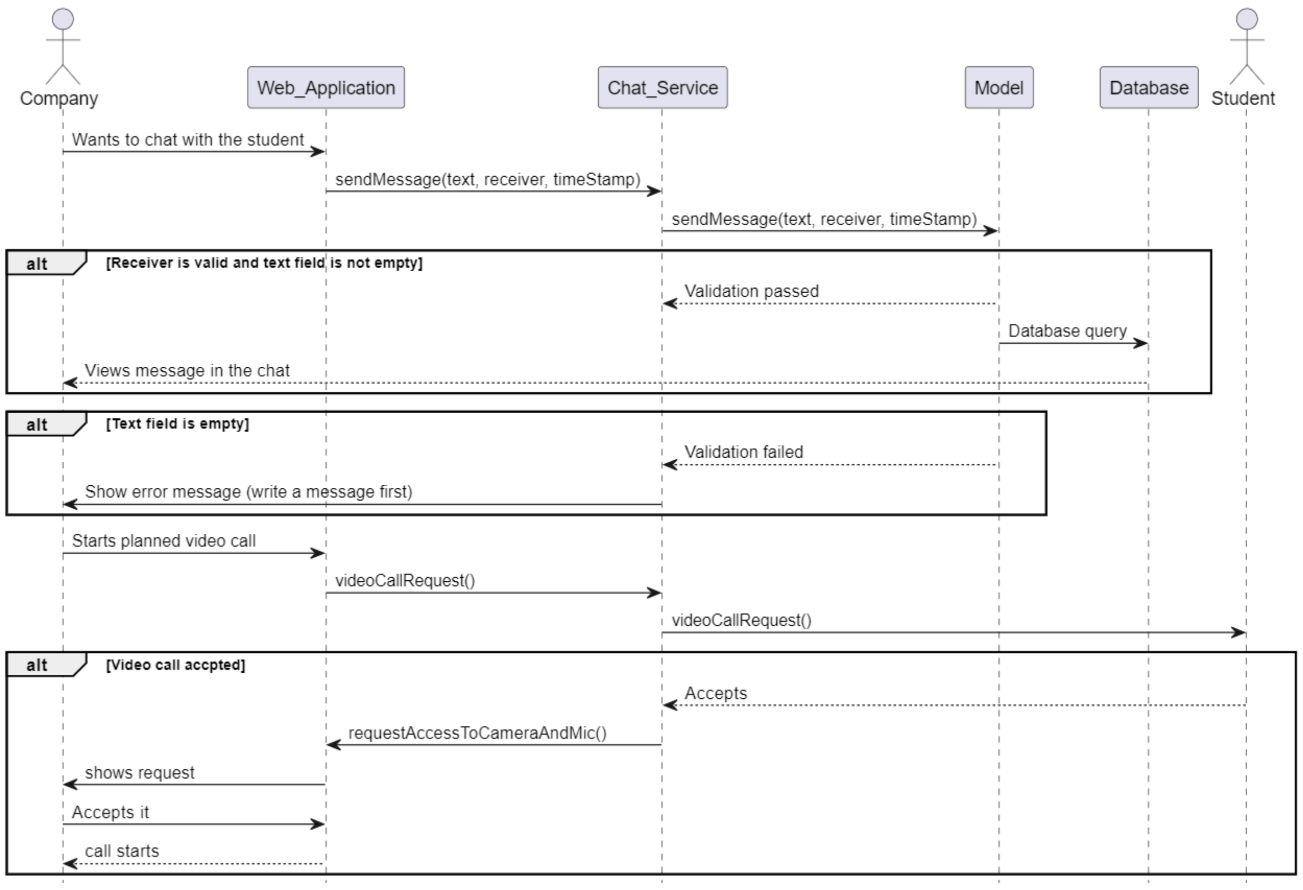
Here it is shown what happens when the university decides to interrupt one of its student's internship. Both the company and the student are notified and then the internship is marked as "Interrupted".

2.4.9 Company observes candidates for its internship



Here it is possible to observe the runtime view describing the process of a company observing all the students who applied to its internship. The data displayer fetch the applications, the one obtained through the recommendation system are highlighted.

2.4.10 Student participates to an interview


Company
Web_Application
Chat_Service
Model
Database
Student
Wants to chat with the student
sendMessage(text, receiver, timeStamp)
sendMessage(text, receiver, timeStamp)
alt [Receiver is valid and text field is not empty]
Validation passed
Database query
Views message in the chat
alt [Text field is empty]
Validation failed
Show error message (write a message first)
Starts planned video call
videoCallRequest()
videoCallRequest()
alt [Video call accpted]
Accepts
requestAccessToCameraAndMic()
shows request
Accepts it
call starts

In this runtime it is described how an interview is handled, between a Student and a Company. We decided not to replicate the web application and the chat service modules on the student side to make the diagram readable. When the video call is started by the company, a request in the chat is sent to the student that can see it through the web application. He can join it and be interviewed.

2.5 Component Interfaces

This section defines the interfaces for the various components within the system. Each interface lists the method signatures, describing the functionality provided by the respective component.

- **CV Service**
  - `createCV()`
  - `uploadCV(CVFile)`
  - `updateCV(field, data)`
  - `deleteCV()`
  - `getCVFile()`
  - `searchCVs(criteria)`
  - `validateCV(data)`
  - `analyzeCV(cv)`
- **Internship Service**
  - `createInternship()`
  - `updateInternship(field, data)`
  - `deleteInternship()`
  - `endInternship(internshipID)`
  - `checkStatus(internshipID)`
  - `checkAvailability(internshipID)`
  - `addStudent(studentID, internshipID)`
  - `removeStudent(studentID, internshipID)`
  - `listInternships(filters)`
  - `extendInternship(internshipID, duration)`
- **Registration Service**
  - `registerData(name, email, password, username, role)`
  - `updateData(data)`
  - `deactivateAccount(username)`
  - `resendVerification(email)`
- **Auth Service**

  - `loginData(email, password)`
  - `logout(username)`
  - `resetPassword(email)`

- **Notification Service**
  - `notify(message, receiverUsernames[])`
  - `scheduleNotification(username, time, message)`
  - `unsubscribe(userID)`

- **Matching Engine**
  - `findMatch(CV[], internshipID)`
  - `recommendMatches(username)`
  - `sendResult()`
  - `analyzeMatchQuality(matchID)`
  - `uploadStatistics()`
  - `runMatchingEngine()`

- **CV Analyzer**
  - `getPersonalData(cv)`
  - `getSkills(cv)`
  - `getSoftSkills(cv)`
  - `getEducation(cv)`
  - `getPastExperiences(cv)`
  - `getInterests(cv)`
  - `summarizeCV(cv)`
  - `analyzeCV(cv)`

- **Internship Analyzer**
  - `getTitle()`
  - `getDescription()`
  - `getSkills()`
  - `getDuration()`
  - `getBenefits()`
  - `getLocation()`

  - `rankInternships(criteria)`
  - `analyzeInternship(internshipData)`

- **Complaints Handler**
  - `getComplaint(complaintID)`
  - `getComplaintsByUser(username)`
  - `filterComplaint()`
  - `updateRating(rating, companyUsername)`
  - `escalateComplaint(complaintID)`
  - `resolveComplaint(complaintID, resolutionDetails)`

- **Data Displayer**
  - `fetchApplications(internshipID)`
  - `displayInternships()`
  - `displayTailoredInternships(username)`
  - `filterApplications(filters)`
  - `filterInternships(filters)`
  - `displayStudents()`

- **Feedback Handler**
  - `createFeedback()`
  - `fetchFeedback(username)`
  - `deleteFeedback(feedbackID)`

- **Chat Service**
  - `sendMessage(text, receiver, timestamp)`
  - `videocallRequest(sender, receiver)`
  - `endVideocall(videoCallID)`
  - `blockUser(usernameBlocking, usernameBlocked)`

- **Model**
  - `checkUsage(username)`
  - `loginData(email, password)`
  - `cancelUser(username)`

  - `checkCredentials(email, password)`
  - `checkRole(username)`
  - `selectProfile(username)`
  - `updateProfile(username, data)`
  - `retrievTailoredInternships(username)`
  - `getInternships(filters)`
  - `getCVs()`
  - `getCommonSkills()`
  - `applyToInternship(username, internshipID)`
  - `checkApplication(username, internshipID)`
  - `endInternship(InternshipID)`
  - `registerData(name, email, password, username, role)`
  - `uploadCV(skills, softSkills, education, pastExperience)`
  - `uploadInternship(title, description, skills, duration, benefits, location, softSkills, hours)`
  - `submitFeedback(rate, company, jobTitle, comment, images)`

- **DBMS**
  - `add(data: Query)`
  - `remove(data: Query)`
  - `update(data: Query)`
  - `fetch(query)`

- **SMTP**
  - `notify(message, receiverEmails[])`

2.6 Selected architectural styles and patterns

The *Students & Companies (S&C)* platform utilizes a **three-tier architectural style**, composed of a **presentation layer**, a **business logic layer**, and a **data layer**. This choice supports separation of concerns, scalability, and maintainability, aligning with the system's requirements.

2.6.1 Architectural Styles

- **Three-Tier Architecture**:
  - **Presentation Layer**: Handles interactions with users via a web application. This layer communicates with the business logic layer to present tailored internship suggestions, CV creation forms, and other user-specific content.
  - **Business Logic Layer**: Acts as the intermediary, managing core functionalities such as the matching engine for internships, feedback collection, and authentication. It enforces business rules and processes data exchanges between the other layers.
  - **Data Layer**: Manages the storage, retrieval, and consistency of system data, including user profiles, CVs, internships, and feedback records.

  This structure ensures system modularity and provides a foundation for potential extensions, such as integrating new algorithms for student-internship matching or analytics.

2.6.2 Design Patterns

- **Model-View-Controller (MVC)**: The S&C platform uses the MVC pattern to maintain a clear separation between:
  - **Model**: Represents the data and business rules, ensuring data integrity during user operations.
  - **View**: Displays information to the user through responsive web pages, adapting dynamically based on user interactions.
  - **Controller**: Manages communication between the model and view, processing user inputs and executing business logic.
- **Observer Pattern**: Notifications to students and companies (e.g., new internship postings, application status updates) are handled using the observer pattern. The **Notification Service** acts as the observer, ensuring timely updates across all registered stakeholders.

- **Dependency Injection**: Components such as the **Matching Engine**, **CV Analyzer**, and **Internship Analyzer** utilize dependency injection to maintain flexibility and simplify testing by decoupling dependencies.
- **RESTful APIs**: Internal communication between the presentation and business logic layers is facilitated by RESTful APIs, ensuring standardized and stateless interactions.

2.6.3 Advantages

- **Scalability**: The architecture allows independent scaling of components. For instance, the **Matching Engine** and **Notification Service** can be scaled based on usage patterns without affecting other modules.
- **Maintainability**: The separation of concerns between layers and the adoption of modular components enable easier updates and the introduction of new features with minimal disruptions.
- **Flexibility**: The use of patterns like MVC and dependency injection ensures that changes in one layer or component have limited ripple effects across the system.

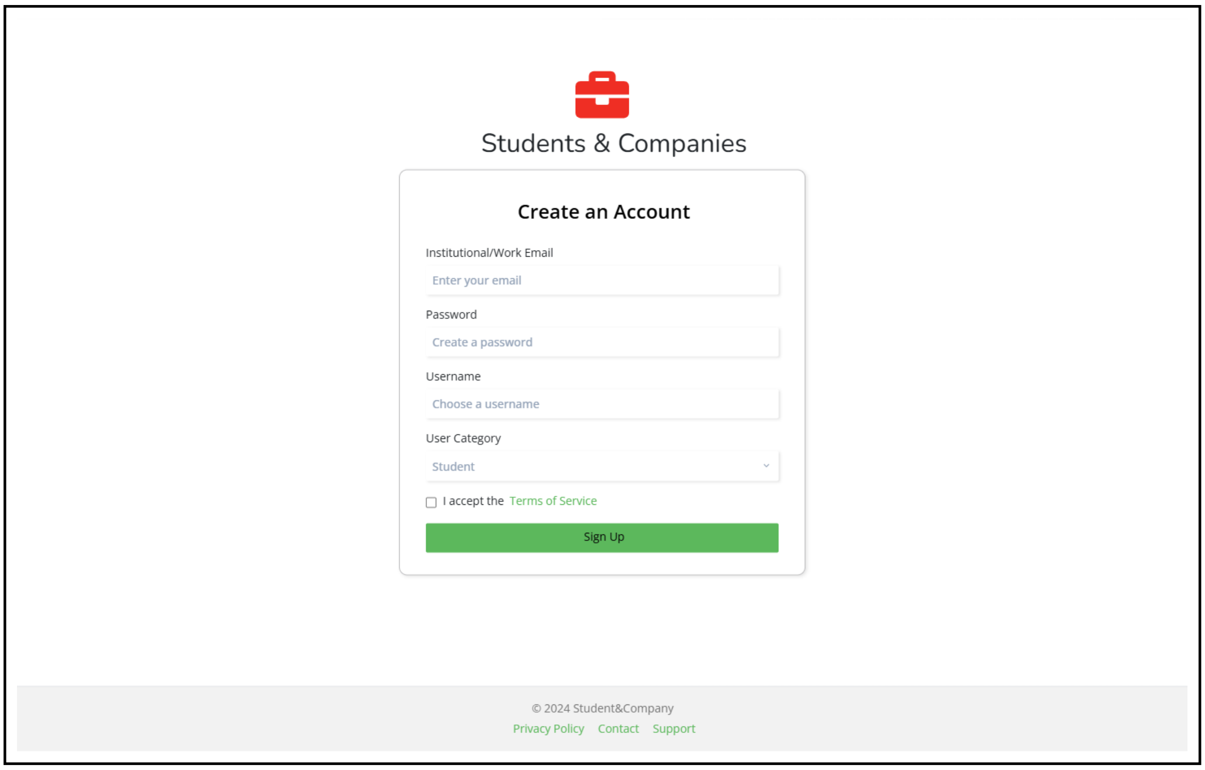
By employing these styles and patterns, the S&C platform achieves a robust and adaptable architecture tailored to its goal of streamlining interactions between students, companies, and universities.

3 User Interface Design

3.1 UI mockups

In this section we present the main tabs of the Student&Company website.

3.1.1 Registration page


Students & Companies

Create an Account

Institutional/Work Email

Enter your email

Password

Create a password

Username

Choose a username

User Category

Student

I accept the Terms of Service

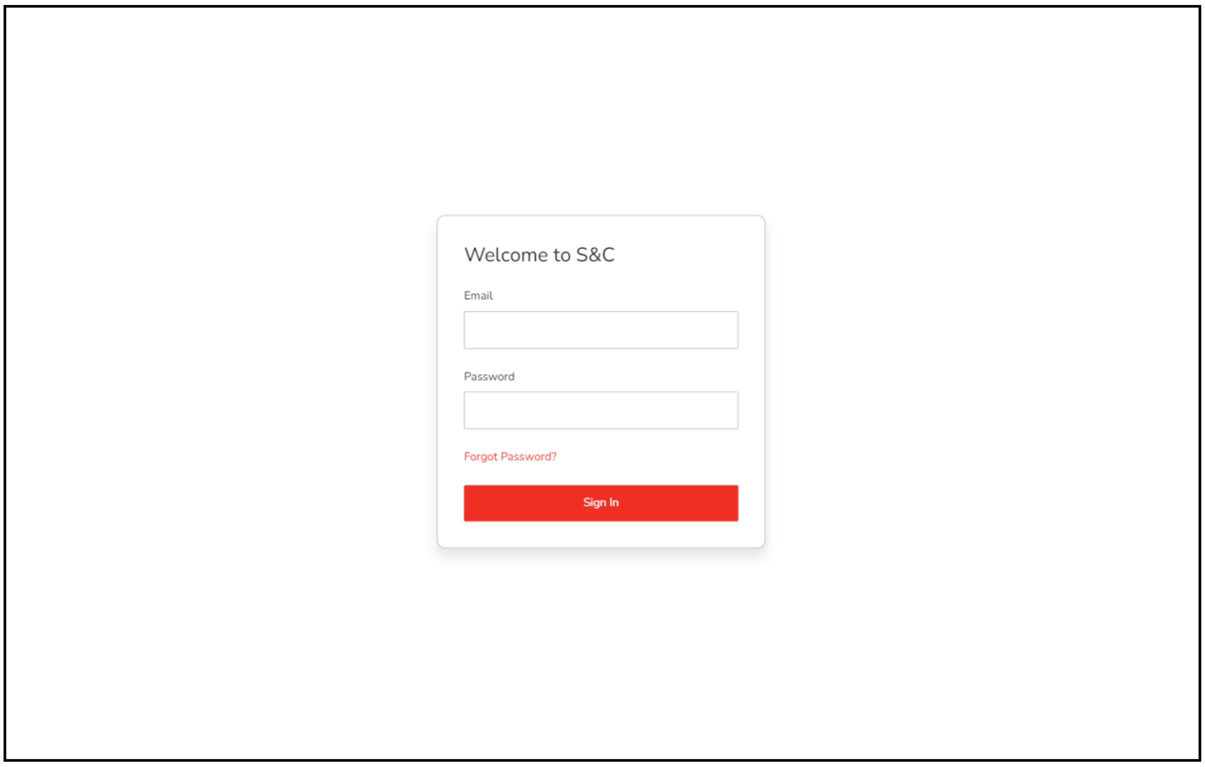
Sign Up

© 2024 Student&Company

Privacy Policy Contact Support

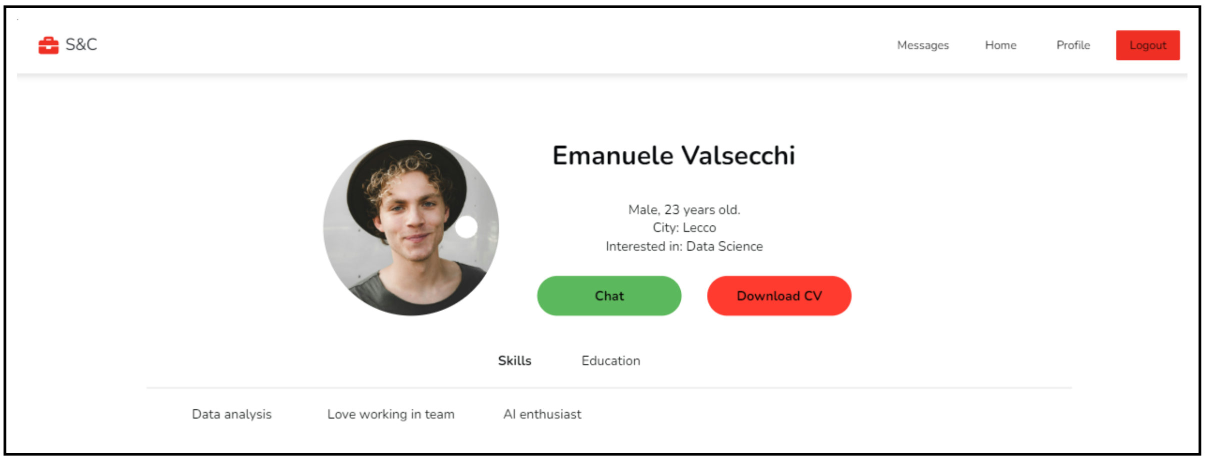
Here we see the registration page, where each unregistered user has to enter his/her name, the work/school email, a username, the role and a password to register. He/She also has to accept the terms of service before continuing.

3.1.2 Login page



The login page consists of two boxes where a registered user can enter his/her credentials gathered from the registration process.

3.1.3 Student profile



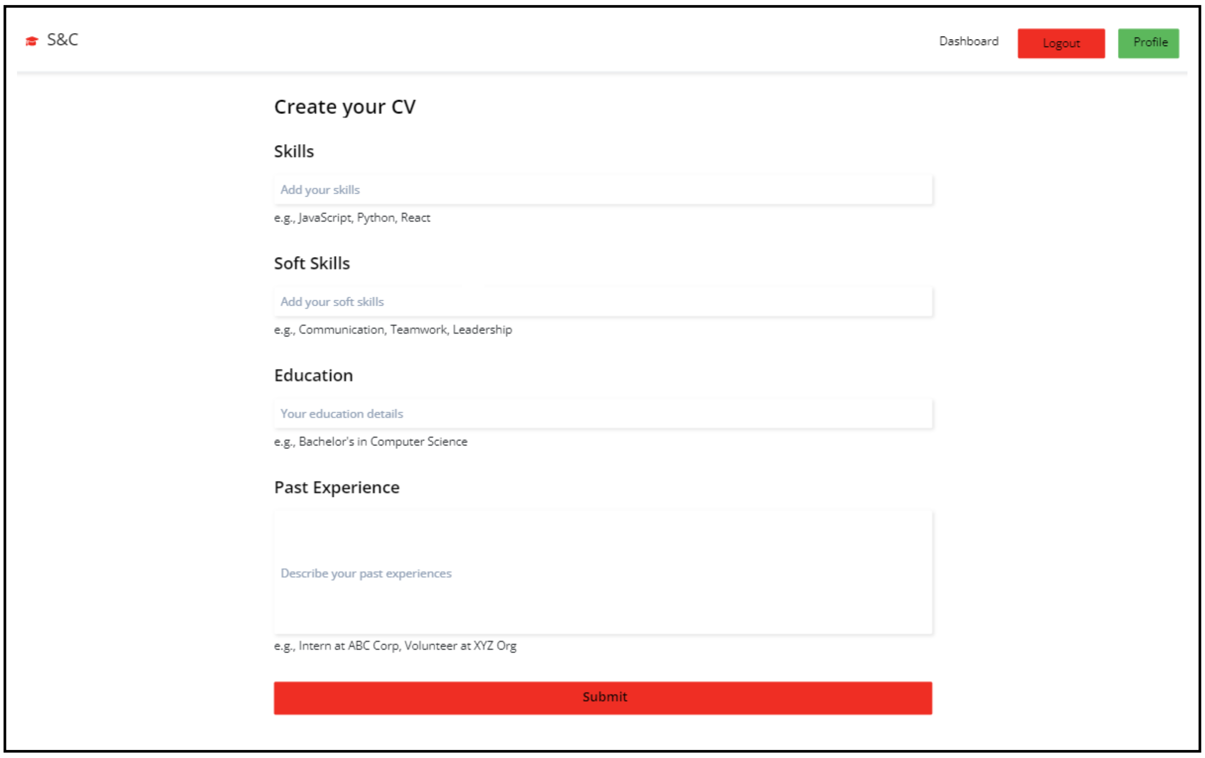
Emanuele Valsecchi

Male, 23 years old.
City: Lecco
Interested in: Data Science

Data analysis
Love working in team
AI enthusiast

This is a preview of a possible student profile where it is possible to see his skills and possibly his education, in order to have a complete overview of a possible candidate. It can be seen by University, Company and the student himself.

3.1.4 Form to create CV


S&C

Dashboard | Logout | Profile

Create your CV

Skills

Add your skills

e.g., JavaScript, Python, React

Soft Skills

Add your soft skills

e.g., Communication, Teamwork, Leadership

Education

Your education details

e.g., Bachelor's in Computer Science

Past Experience

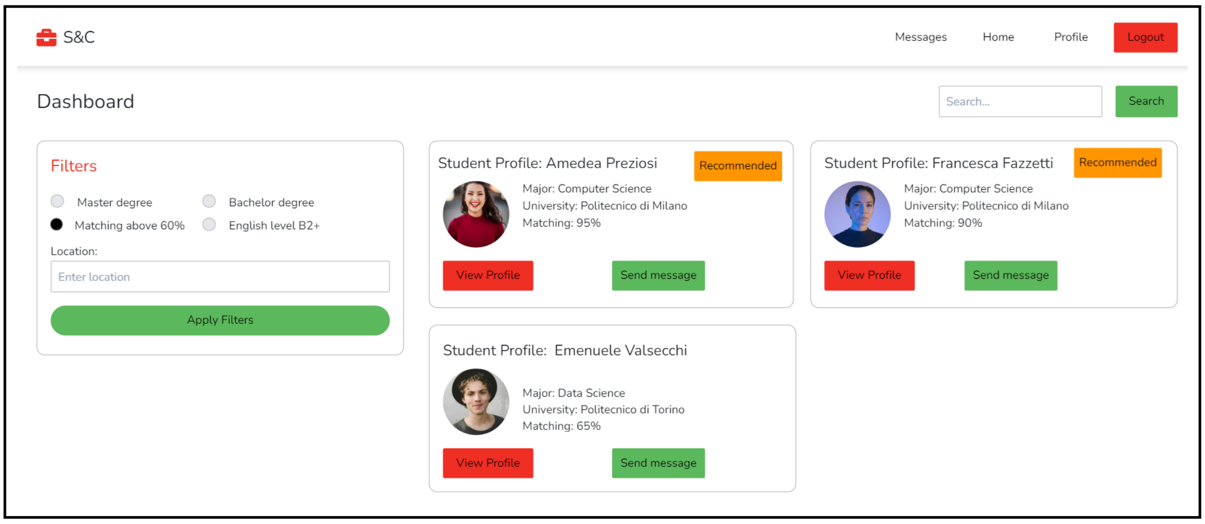
Describe your past experiences

e.g., Intern at ABC Corp, Volunteer at XYZ Org

Submit

This is the form to create the CV directly on the web application, accessible only to students.

3.1.5 Company visualizes internship's applications


S&C

Messages | Home | Profile | Logout

Dashboard

Search... | Search

Filters

Master degree

Bachelor degree

● Matching above 60%

English level B2+

Location:

Enter location

Apply Filters

Student Profile: Amedea Preziosi — Recommended

Major: Computer Science
University: Politecnico di Milano
Matching: 95%

View Profile | Send message

Student Profile: Francesca Fazzetti — Recommended

Major: Computer Science
University: Politecnico di Milano
Matching: 90%

View Profile | Send message

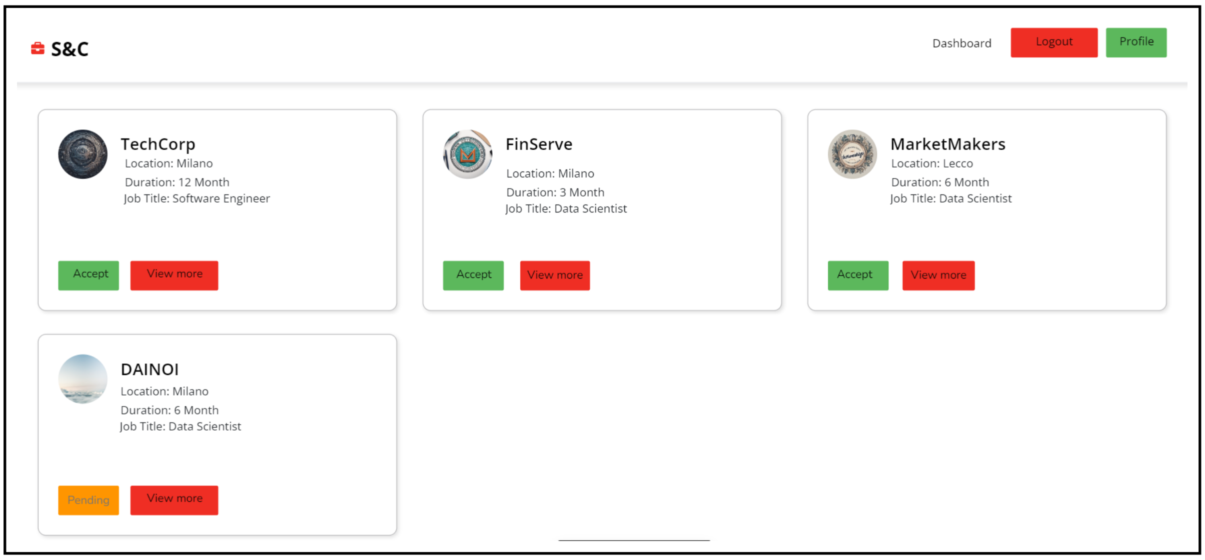
Student Profile: Emenuele Valsecchi

Major: Data Science
University: Politecnico di Torino
Matching: 65%

View Profile | Send message

In this page the company can see applicants to its internship. Students who found the internship post thanks to the recommendation system are highlighted. Non-recommended students are shown as well with their relative matching percentage.

3.1.6 Student manage its future internships



TechCorp
Location: Milano
Duration: 12 Month
Job Title: Software Engineer

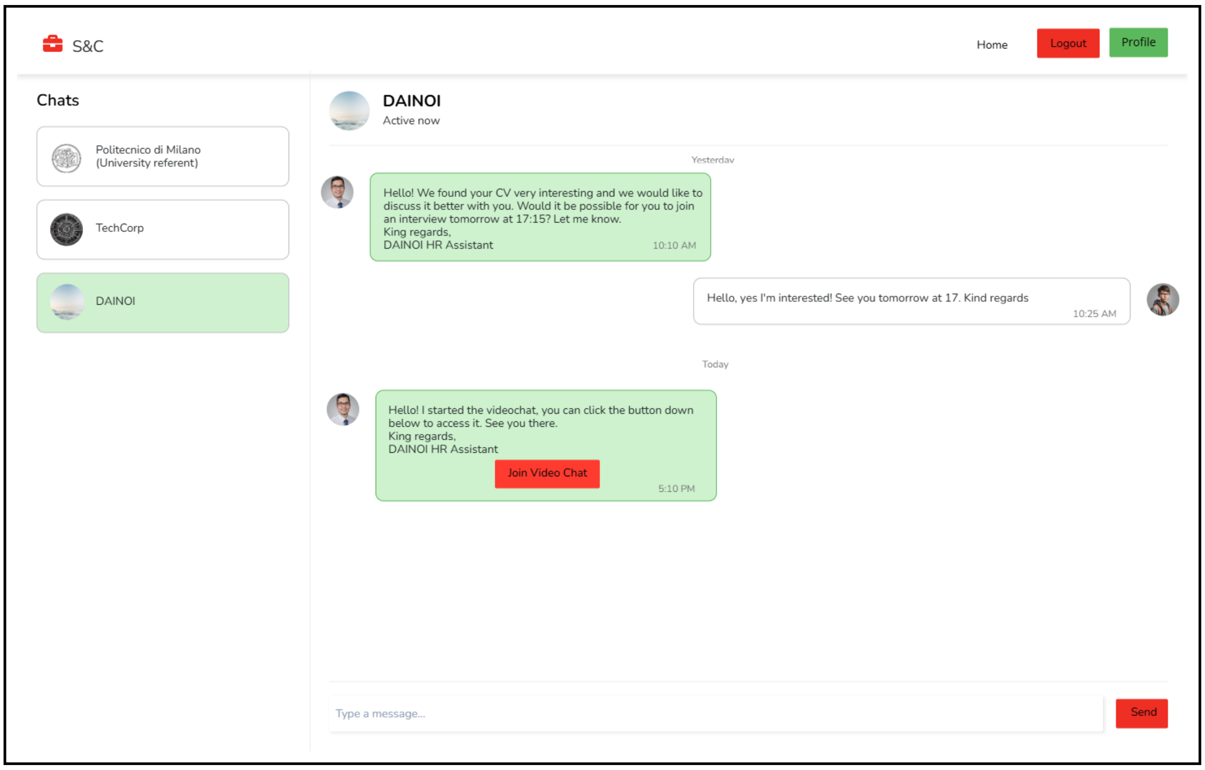
FinServe
Location: Milano
Duration: 3 Month
Job Title: Data Scientist

MarketMakers
Location: Lecco
Duration: 6 Month
Job Title: Data Scientist

DAINOI
Location: Milano
Duration: 6 Month
Job Title: Data Scientist

In this section a student can see his/her applications to the different companies. "Accept" means that the student has successfully passed the interviews, while "Pending" refers to the opportunities that a company has to approve yet.

3.1.7 Student chats with a company and joins an interview



Politecnico di Milano (University referent)

TechCorp

DAINOI

DAINOI
Active now

Yesterday

Hello! We found your CV very interesting and we would like to discuss it better with you. Would it be possible for you to join an interview tomorrow at 17:15? Let me know.
King regards,
DAINOI HR Assistant 10:10 AM

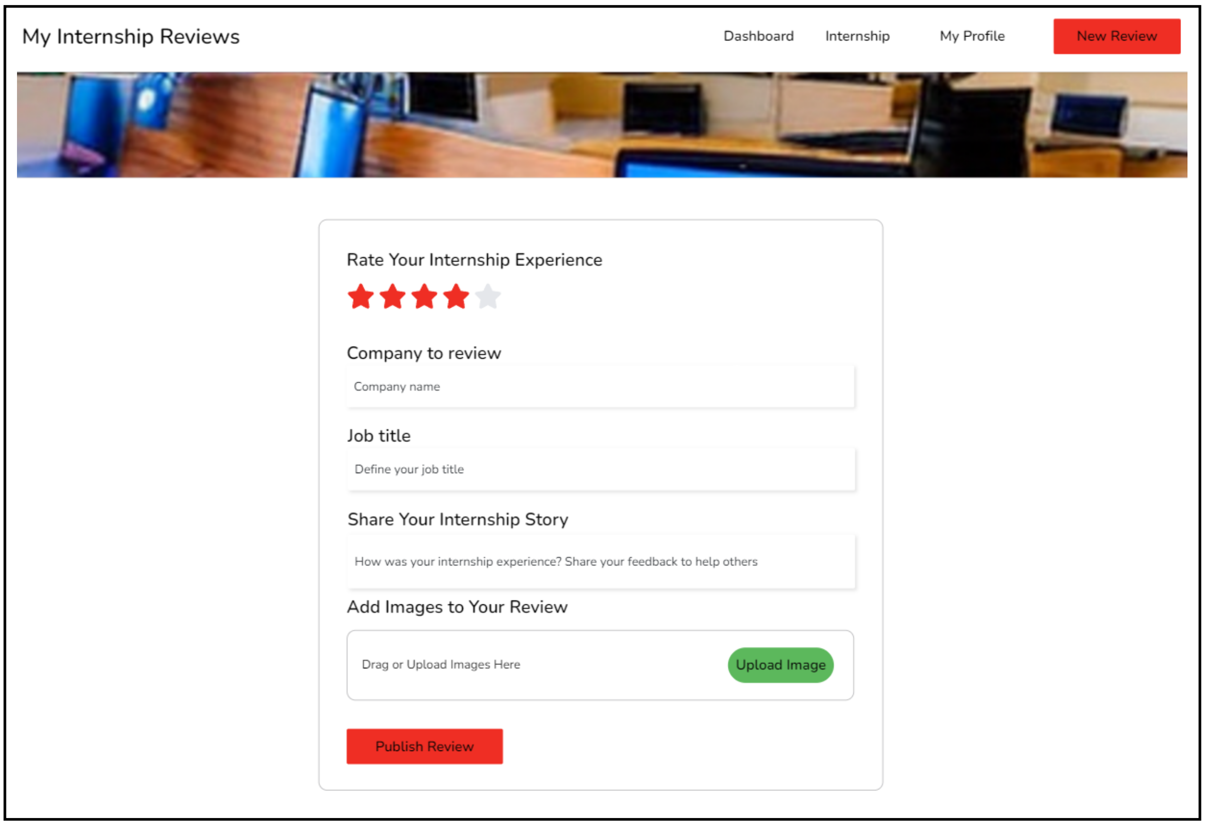
Hello, yes I'm interested! See you tomorrow at 17. Kind regards 10:25 AM

Today

Hello! I started the videochat, you can click the button down below to access it. See you there.
King regards,
DAINOI HR Assistant
5:10 PM

Here it is possible to observe the "Messages" section of a Student: he has a chat with his university, and 2 companies (included the one which is opened). The company invited the student for a video chat interview that he can join through the platform.

3.1.8 Feedback form



Rate Your Internship Experience

Company to review

Company name

Job title

Define your job title

Share Your Internship Story

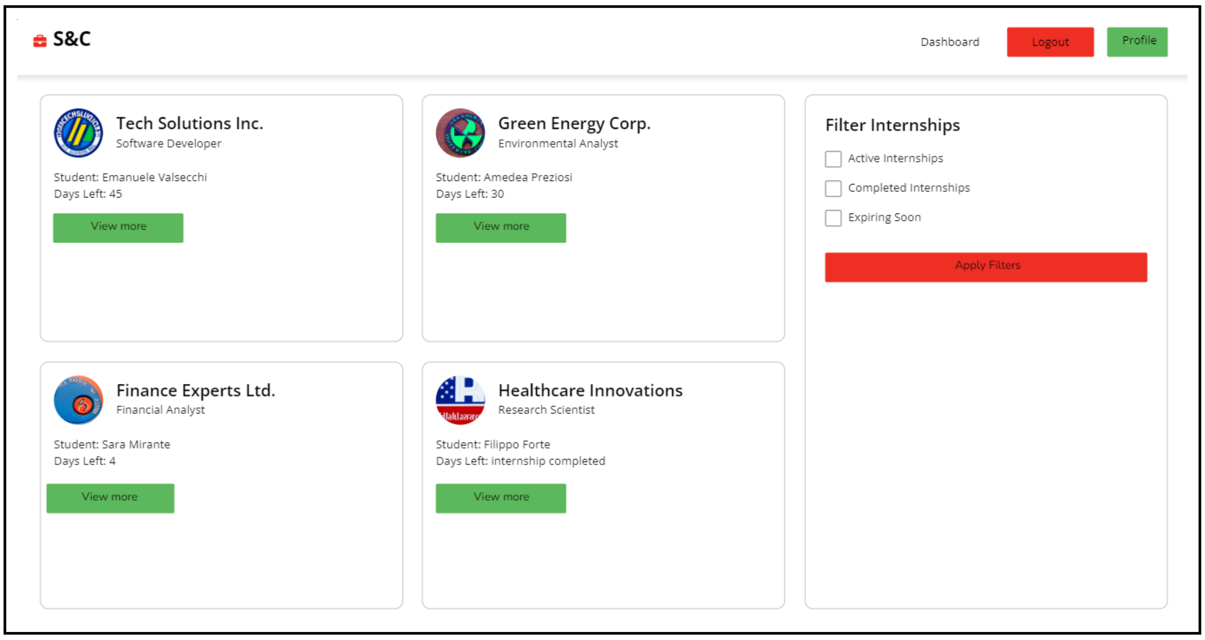
How was your internship experience? Share your feedback to help others

Add Images to Your Review

Drag or Upload Images Here

In this page an intern can review his/her experience when finished (gracefully or not).

3.1.9 University dashboard (ongoing internships)



Tech Solutions Inc.
Software Developer
Student: Emanuele Valsecchi
Days Left: 45

Green Energy Corp.
Environmental Analyst
Student: Amedea Preziosi
Days Left: 30

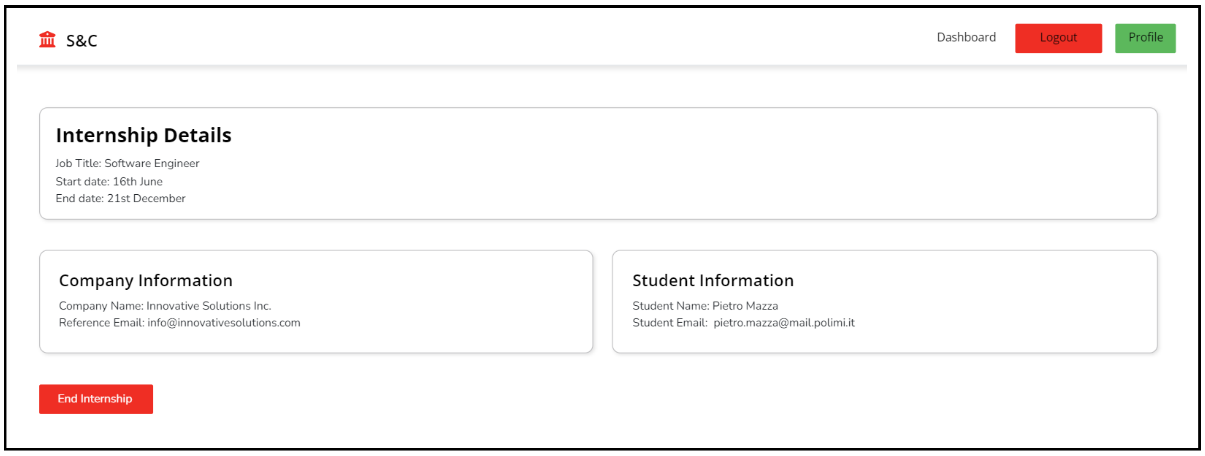
Finance Experts Ltd.
Financial Analyst
Student: Sara Mirante
Days Left: 4

Healthcare Innovations
Research Scientist
Student: Filippo Forte
Days Left: internship completed

Filter Internships
- [ ] Active Internships
- [ ] Completed Internships
- [ ] Expiring Soon

From this page the university representative can monitor its students internship. Obviously it's not accessible from other users.

3.1.10 University can end one of its student internship



Internship Details

Job Title: Software Engineer
Start date: 16th June
End date: 21st December

Company Information

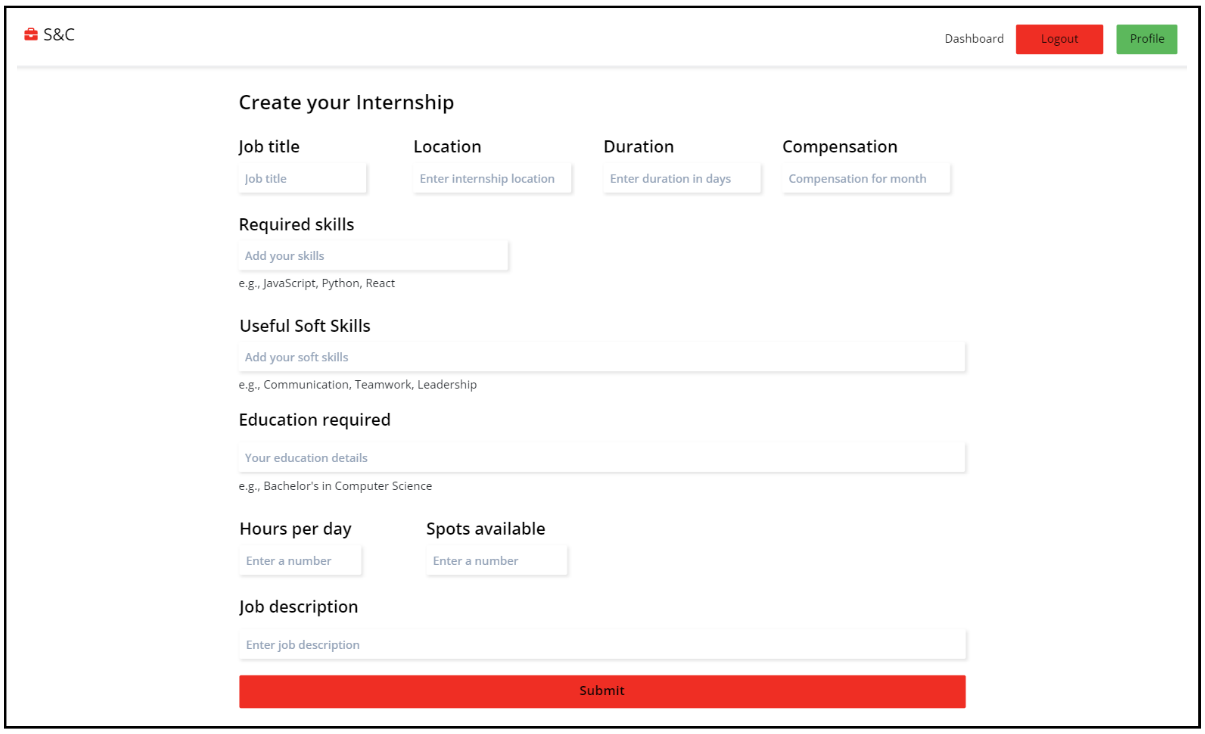
Company Name: Innovative Solutions Inc.
Reference Email: info@innovativesolutions.com

Student Information

Student Name: Pietro Mazza
Student Email: pietro.mazza@mail.polimi.it

This is another section for universities that allows them to see the internship details and eventually take action, if the internship has to be terminated.

3.1.11 Company creates an internship announcement



Create your Internship

Job title: Job title
Location: Enter internship location
Duration: Enter duration in days
Compensation: Compensation for month

Required skills: Add your skills
e.g., JavaScript, Python, React

Useful Soft Skills: Add your soft skills
e.g., Communication, Teamwork, Leadership

Education required: Your education details
e.g., Bachelor's in Computer Science

Hours per day: Enter a number
Spots available: Enter a number

Job description: Enter job description

This section is where the company representative can create an internship post, inserting all the required details.

4 Requirement Traceability

This section shows how the requirements we defined in the RASD map to the design elements defined before in this document.
The **Model** component is mapped to each requirement so it will not be listed below to avoid not needed repetition. **Web Application** is often listed since lots of the requirements are satisfied through it.

4.1 General Requirements (GR)

[GR1] Platform must **prevent** multiple accounts from using the same E-Mail address.
- Registration service

[GR2] Platform must **permit** unregistered users only to access the registering function.
- Registration Service
- Auth Service

[GR3] Platform must **allow** unregistered users to register as Student, Company or University.
- Registration Service

[GR4] Platform must **ensure** to send all the necessary notifications to the correct users right after something relevant occurs.
- Notification Service
- SMTP Server
- Web Application

[GR5] Platform must **find** optimal matching between Students and Companies, based on localization and a match between Company's internship requested skills and Student's abilities.
- Matching Engine

4.2 Student Platform Requirements (SR)

[SR1] Platform must **prevent** Students to access the platform as Company or University account.
- Auth Service

[SR2] Platform must **allow** Students to pro-actively search for internships through the platform itself.
- Web Application
- DBMS
- Data Displayer

[SR3] Platform must **allow** Students to filter internships' announcements based

on their specific likings.
- Web Application
- Data Displayer
- DBMS

[SR4] Platform must **allow** Students to apply for any internship they find interesting.
- Web Application
- DBMS

[SR5] Platform must **prevent** Students to apply more than once for the same internship.
- Web Application

[SR6] Platform must **allow** Students to complain or provide any type of information about the ongoing internship through the platform itself.
- Web Application
- Complaints Handler

[SR7] Platform must **suggest** Students on how to get their CVs more appealing to Companies.
- CV analyzer

[SR8] Platform must **notify** Students about new suitable internships for them.
- Notification Service
- SMTP Server
- Matching Engine

[SR9] Platform must **allow** Students to monitor their ongoing internship's situation.
- Web Application
- Data Displayer

[SR10] Platform must **allow** Students to participate to interviews if selected by any Company.
- Web Application
- Chat service

[SR11] Platform must **allow** Students to provide feedback after their internship experience to improve the platform's matching algorithm.
- Feedback handler
- Web Application

[SR12] Platform must **allow** Students to chat with their University's account and with the Companies they are interested working in or with which they are already doing an internship with.
- Chat service
- Web Application

[SR13] Platform must **show** Students only available internships.
- Web Application
- Data Displayer

[SR14] Platform must **allow** Students to upload their CV and update it whenever they want to.
- CV service
- Web Application

[SR15] Platform must **notify** Students when they are accepted for an internship experience by a Company.
- Notification service
- SMTP Server

[SR16] Platform must **deny** Students to upload a non European Format CV (see *Reference*)
- CV service

[SR17] Platform must **notify** Students about the result of their complaint.
- Notification service
- SMTP Server

[SR18] Platform must **notify** Students when the Company of a recommended internship they tried to apply for, match with their profile.
- Notification service
- SMTP Server

4.3 Company Platform Requirements (CR)

[CR1] Platform must **prevent** Companies to access the platform as Student or University account.
- Auth Service

[CR2] Platform must **allow** Companies to upload their internship announcements whenever they want to.
- Internship Service
- Web Application

[CR3] Platform must **allow** Companies to contact recommended Students whose CVs appear interesting to them.
- Chat service
- Web Application
- Matching Engine

[CR4] Platform must **show** Companies only available Students, who are not already on an internship.
- Data Displayer
- Web Application

[CR5] Platform must **allow** Companies to chat with Students who are working with them as interns.
- Chat service
- Web Application

[CR6] Platform must **allow** Companies to chat with their specific interns' Universities in case of any problem.
- Chat service
- Web Application

[CR7] Platform must **allow** Companies to provide feedback after Students' internship experiences to improve the platform's matching algorithm.
- Feedback handler
- Web Application

[CR8] Platform must **allow** Companies to monitor their interns situation.
- Web Application
- Data Displayer

[CR9] Platform must **prevent** Companies to publish more than once the same internship's announcement.
- DBMS
- Internship Service

[CR10] Platform must **allow** Companies to complain or provide any type of information about the ongoing internships of its interns through the platform itself.
- Web Application
- Complaints handler

[CR11] Platform must **notify** Companies about new suitable student candidates for their internships.
- Notification Service
- SMTP Server
- Matching Engine

[CR12] Platform must **notify** Companies if an internship is interrupted by the student's university.
- Notification Service
- SMTP Server

[CR13] Platform must **notify** Companies about the result of their complaint.
- Notification Service
- SMTP Server

[CR14] Platform must **allow** Companies to plan and carry out interviews to select candidates for their internships
- Chat service
- Web Application

[CR15] Platform must **notify** Companies when a recommended Student they tried to contact, match with one of their internships
- Notification Service
- SMTP Server

[CR16] Platform must **suggest** Companies on how to make their Internship post more appealing to Students.
- Internship Analyzer

4.4 University Platform Requirements (UR)

[UR1] Platform must **prevent** University to access the platform as Student or Company account.
- Auth Service

[UR2] Platform must **allow** Universities' accounts to monitor and access the information on all the specific University's students and their ongoing internships.
- Web Application
- Data Displayer

[UR3] Platform must **allow** Universities' accounts to chat with the specific University's students and the companies its students are working as interns in.
- Chat Service
- Web Application

[UR4] Platform must **prevent** Universities' accounts to monitor other Universities' students internships and situation.
- Web Application
- Auth Service
- Data Displayer

[UR5] Platform must **allow** Universities' accounts to formally interrupt its students' ongoing internships if any kind of problem happens.
- Web Application
- Internship Service

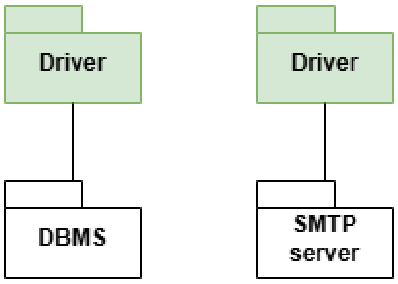
[UR6] Platform must **notify** Universities when a complaint is filled out by one of its students or one of the companies its students are working in as interns.
- Notification Service
- SMTP Server

[UR7] Platform must **allow** Universities' accounts to manage complaints coming from its students or the companies its students are working in as interns.
- Complaints handler
- Web Application

5 Implementation, Integration and Test Plan

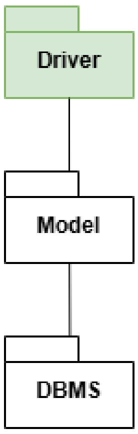
This section provides an overview on implementation and integration that will follow a bottom-up approach to also have a subsequential incremental testing phase.

5.1 Integration with DBMS and SMTP



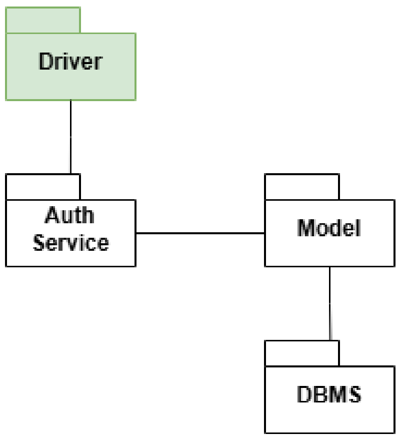
We start integrating and testing the "lower" components that are the DBMS and the SMTP server.

5.2 Integration with the Model



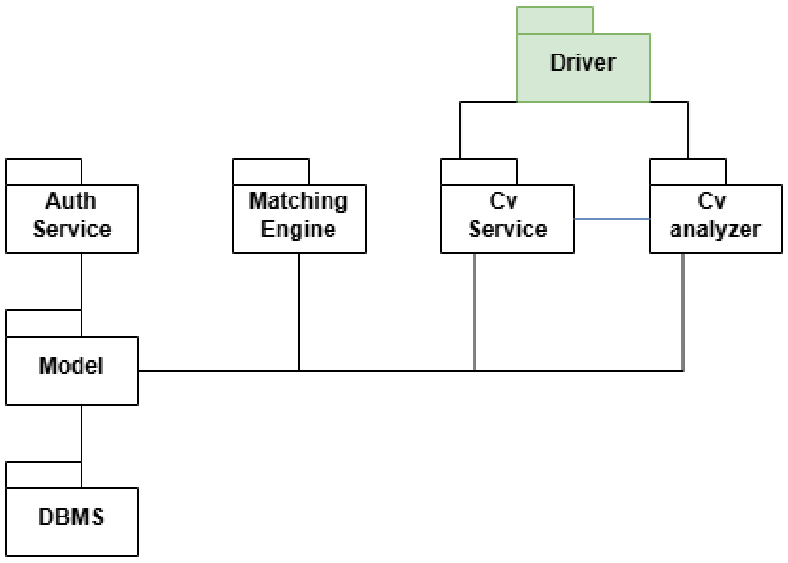
The model is the component we add right after since it will be the "base" of our platform.

5.3 Authentication Service



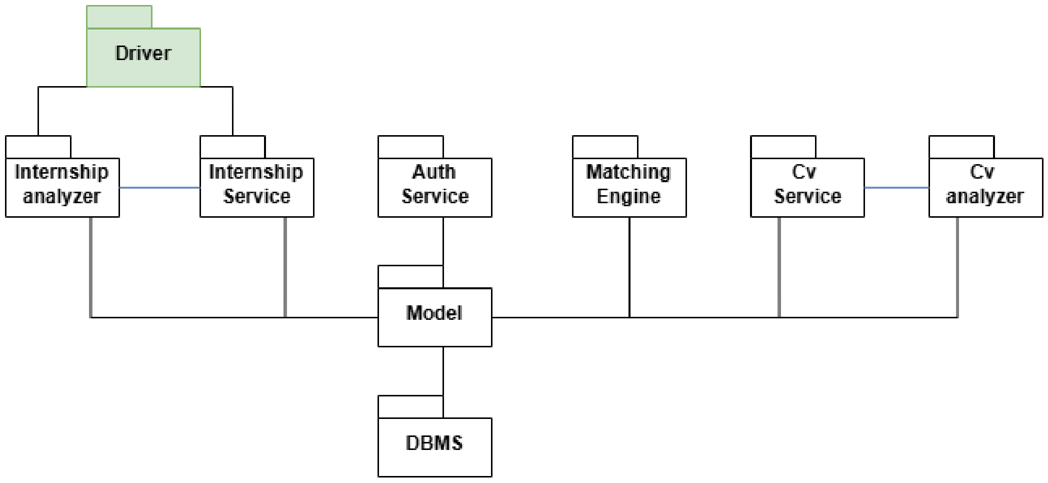
As the 3rd step we include the Authentication service that is needed to include all the login and authentication procedures.

5.4 CV components



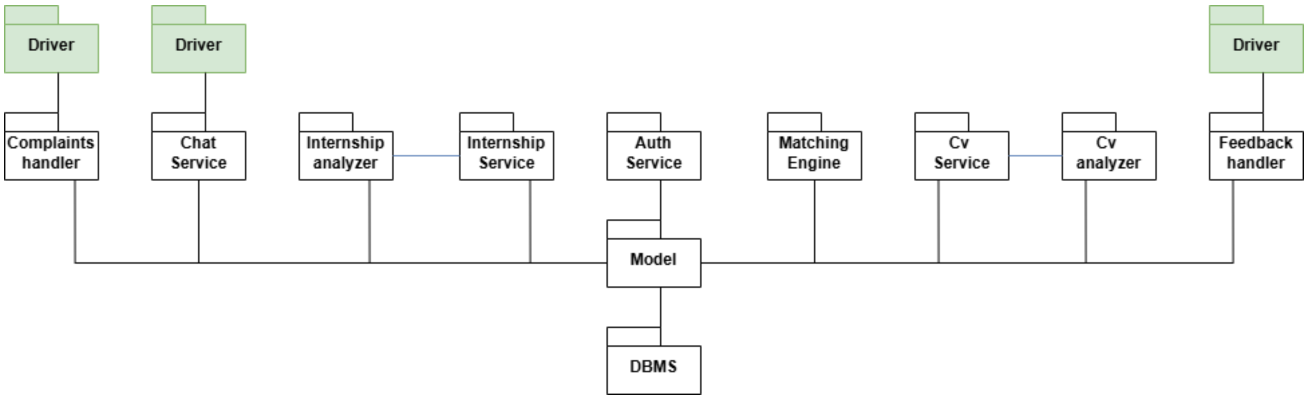
Then, we include and test together the 2 components related to the CV since they are strictly related.

5.5 Internship components



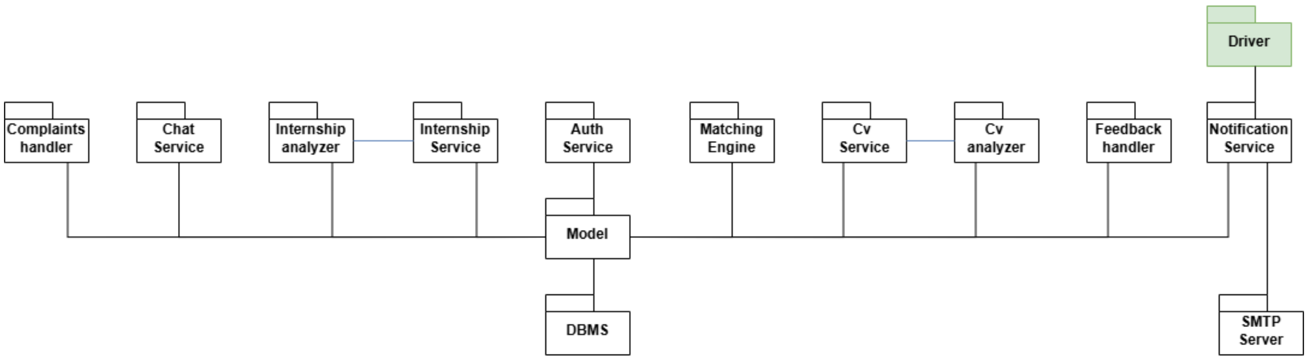
Same as stated in the previous integration, but for the internship related components.

5.6 User communication components



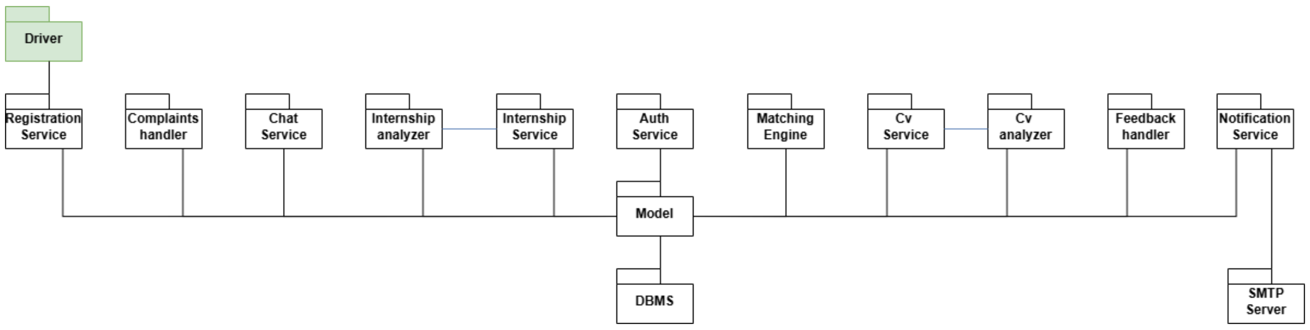
Here we include and test 3 new components that are all related to user's communication: the chat service, and the 2 handler (complaints and feedback).

5.7 Notification component



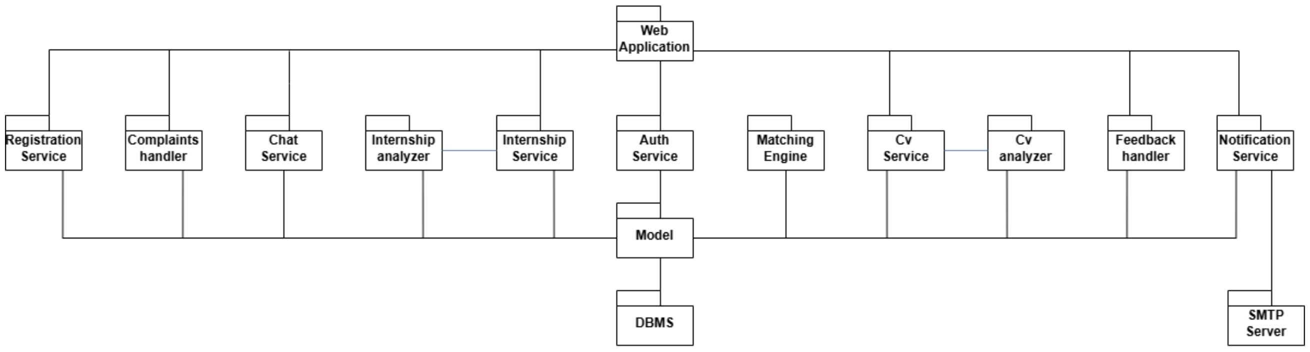
This integration is very important since is about the notification service that handles all the notification to the users, a key part of our platform.

5.8 Registration service component



As one of the last components to be added it's the registration one, that handles the user registration procedures.

5.9 Web Application integration



As the last step, the web application (user's end) is included completing the process of integration and testing.

5.10 Testing

Testing shall take place whenever new functionalities are introduced to uncover potential bugs. While developing the system, users and stakeholder feedback remains indispensable to meet their needs: alpha versions are distributed to collect these feedback.
Since we chose to adopt a bottom-up approach, drivers will be designed to test new components when added to the system. Different types of testing will be involved, firstly Functional testing to verifies that the system works as stated in the use cases and meets the requirements, then Performance, Stress, Load Testings should be performed in order to assure a valuable experience to the

users. It is also important to test components together to verify the correctness of the whole implementation.

6 Effort Spent

Here we include a table indicating how many hours each component has worked in every section.

Student	**Sect. 1**	**Sect. 2**	**Sect. 3**	**Sect. 4**	**Sect. 5**
Alessandro Vitobello	3h	12h	14h	3h	5h
Jurij Diego Scandola	7h	12h	4h	3h	3h
Francesco Raimondi	3h	10h	3h	8h	7h

7 References

- `Course slide from Webeep.it and notes`
- `Uizard.io for the UI mock-ups`