SOFTWARE ENGINEERING-II
COMPUTER SCIENCE AND ENGINEERING

Design Document StudentXCompanies

Authors
Shaurya Aditya Singh
Ergun Gani Çalışkan
Mohammadali Khaledi

Academic year
2024-25

Deliverable: DD
Title: Design Document

Authors:

Shaurya Aditya Singh

Ergun Gani Çalışkan

Mohammadali Khaledi

Version: 1.0
Date: 07-January-2025

Download page:

Table of Contents

1| Introduction

1.1. Purpose

The Student&Companies (S&C) platform is a web-based application designed to streamline and improve the process of connecting students seeking internships with companies offering them. S&C leverages technology, including a sophisticated recommendation system, to make internship matching more efficient, personalized, and transparent. The platform aims to solve the following problems:

- Difficulty for students in finding relevant internships: Students often spend a significant amount of time searching through numerous job boards and company websites to find suitable internship opportunities.
- Lack of personalization: Traditional internship search methods often do not take into account a student's specific skills, interests, and academic background, resulting in irrelevant matches.
- Inefficient application processes: Applying for internships can be time-consuming, involving multiple platforms and inconsistent communication channels.
- Limited feedback and transparency: Students often receive little feedback on their applications, making it difficult to understand areas for improvement.

S&C aims to address these challenges by:

- Providing a centralized platform for students and companies to interact.
- Using a recommendation system to suggest relevant internships to students and suitable candidates to companies.
- Offering tools for efficient application management and communication.
- Facilitating feedback sharing to enhance transparency and improve the internship experience for both parties.

1.2. Scope

The S&C platform will include the following features and functionalities:

- Student Profiles: Students can create comprehensive profiles, including their academic background, skills, interests, and experience. They can also upload their CVs and portfolios.

- Company Profiles: Companies can create profiles to showcase their organization, internship programs, and company culture.
- Internship Postings: Companies can post detailed internship descriptions, including required qualifications, responsibilities, duration, location, and compensation.
- Recommendation System: An intelligent recommendation system will analyze student profiles and internship postings to suggest relevant matches for both parties.
- Application Management: Students can easily apply for internships and track the status of their applications. Companies can manage incoming applications, shortlist candidates, and communicate with applicants directly through the platform.
- Feedback and Rating System: Students and companies can provide feedback and ratings on their internship experiences. This feedback can help improve future matches and enhance the platform's effectiveness.
- Communication Tools: S&C will offer messaging and notification features to facilitate communication between students and companies.

The S&C platform is not intended to:

- Replace traditional job boards or recruitment agencies entirely.
- Guarantee internship placements for all students.
- Provide legal or financial advice related to internships.

1.3. Definitions, Acronyms, Abbreviations

Definitions:

- Recommendation System: A machine learning system that uses algorithms to analyze data and make personalized suggestions for users.
- CV (Curriculum Vitae): A document summarizing a student's education, skills, and experiences.

Acronyms:

- S&C: Students & Companies
- UI: User Interface
- UX: User Experience

Abbreviations:

- R*: functional requirement
- UC*: use case
- UI: user interface

1.4. Revision History

Version: 1
Date: 07/01/2025
Description:Initial draft of the DD.

1.5. Reference Documents

- Requirement Analysis Specification Document (RASD) for Student&Companies
- Software Engineering II Course Materials

1.6. Document Structure

This Design Document is organized as follows:

1. Introduction: Provides an overview of the S&C platform, its purpose, scope, and key definitions.
2. Architectural Design: Describes the technical structure and design of the platform.
3. Requirements Traceability: Maps the requirements outlined in the RASD to specific design elements.
4. User Interface Design: Showcases the user interface and design principles.
5. Implementation, Integration, and Test Plan: Outlines the development and testing approach.
6. Effort Spent: Documents the time and resources used in creating this document.
7. Bibliography: Lists the sources cited in the document.

2| Architectural Design

2.1. Overview

The S&C platform will use a 3-tier architecture, a common and scalable model for web applications. The three tiers are:

1. Presentation Tier (Client): Responsible for handling user interactions and displaying the user interface. This tier will be implemented using a combination of HTML, CSS, and JavaScript frameworks.
2. Application Tier (Application Server): Houses the business logic of the platform, processing user requests, managing data, and implementing the recommendation system. This tier can be built using languages like Python, Java, or Node.js and will interact with the database.
3. Data Tier (Database Server): Stores all the platform's data, including user profiles, internship postings, applications, and feedback. This tier can utilize a relational database management system (RDBMS) like PostgreSQL or MySQL for structured data storage.

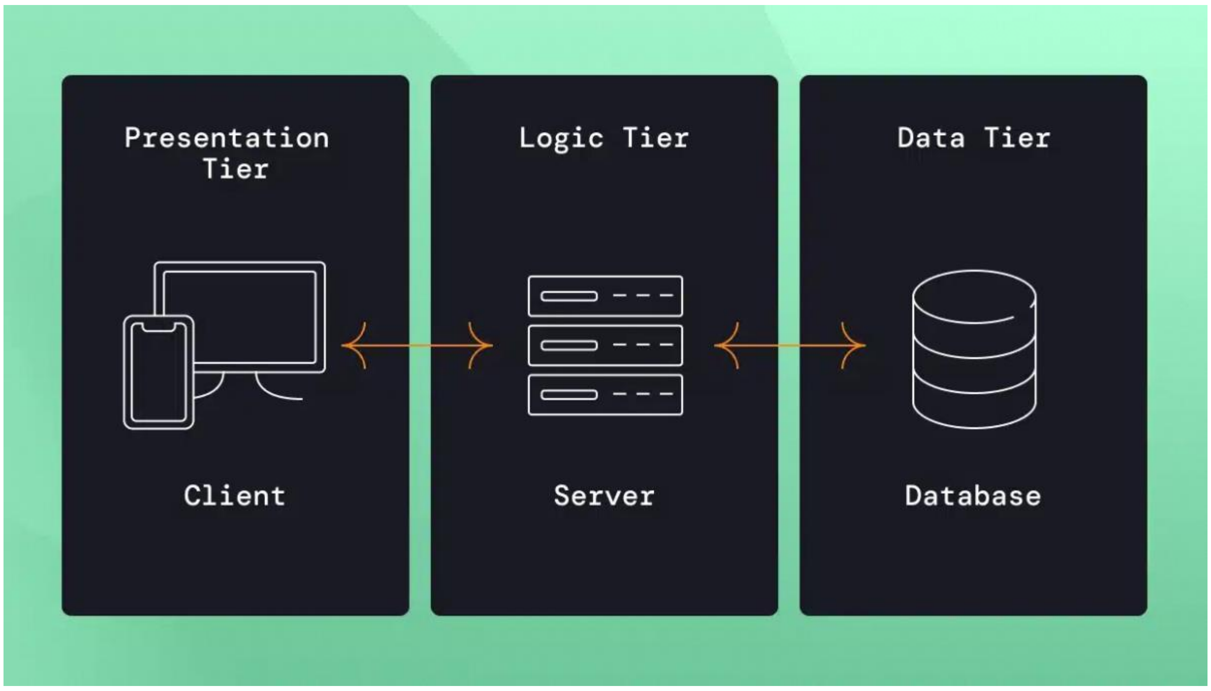


Fig. 2.1 3 Tier Architecture

Design Patterns:

- Model-View-Controller (MVC): The MVC pattern will be used to structure the application tier. This will help separate concerns, making the code more maintainable and testable.

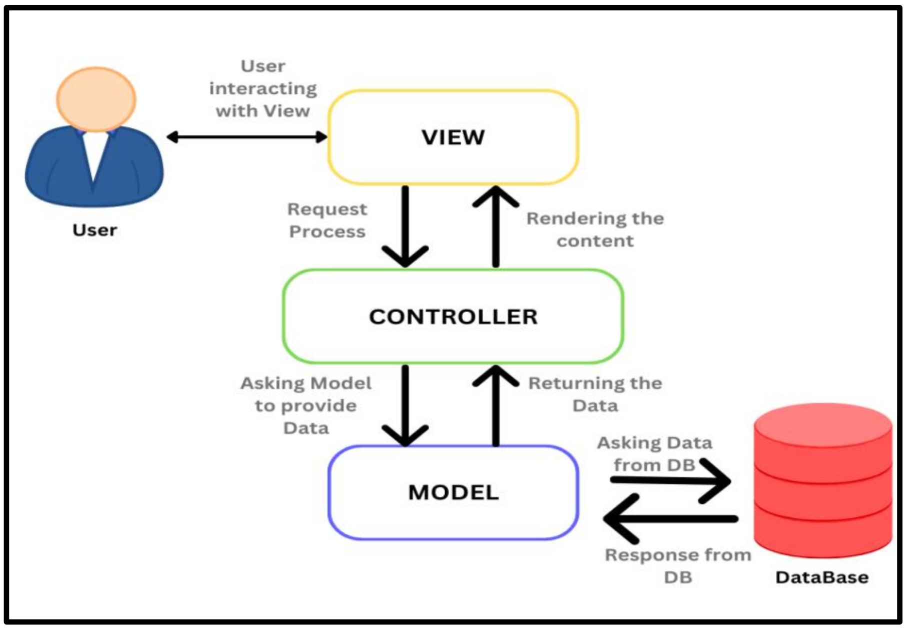


Fig.2.2 MVC model

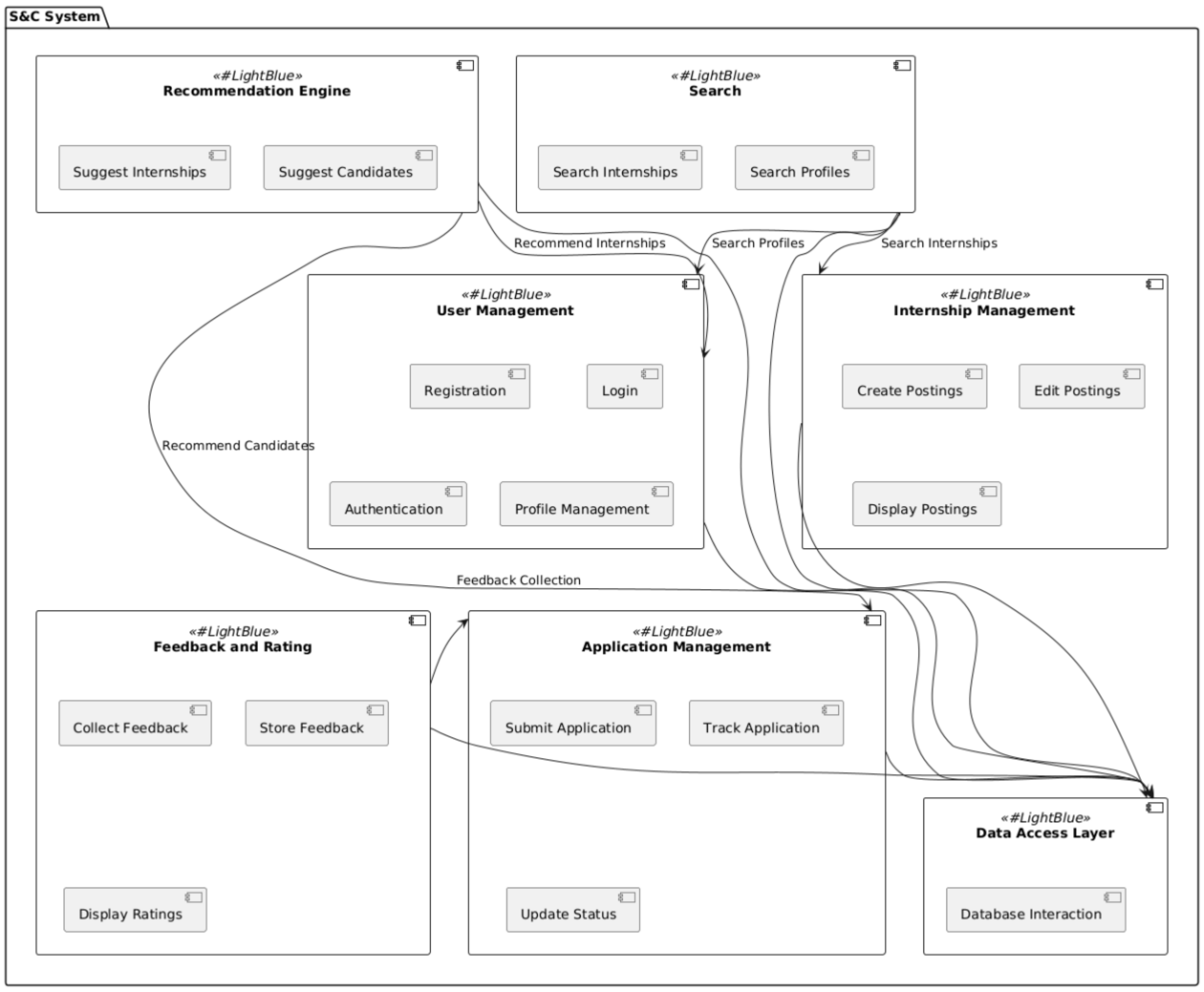


Fig. 2.3 Component view

2.2. Component View

- User Management Component: Handles user registration, login, authentication, and profile management for both students and companies.
- Internship Management Component: Responsible for creating, editing, managing, and displaying internship postings.
- Application Management Component: Manages the application process, including application submission, tracking, and status updates.
- Recommendation Engine Component: Implements the recommendation algorithms to suggest relevant internships to students and suitable candidates to companies.
- Feedback and Rating Component: Handles the collection, storage, and display of feedback and ratings for internships and user experiences.

- Search Component: Allows users to search for internships and profiles using various filters.
- Data Access Layer: Provides an abstraction layer to interact with the database, ensuring data integrity and security.

2.3. Deployment View

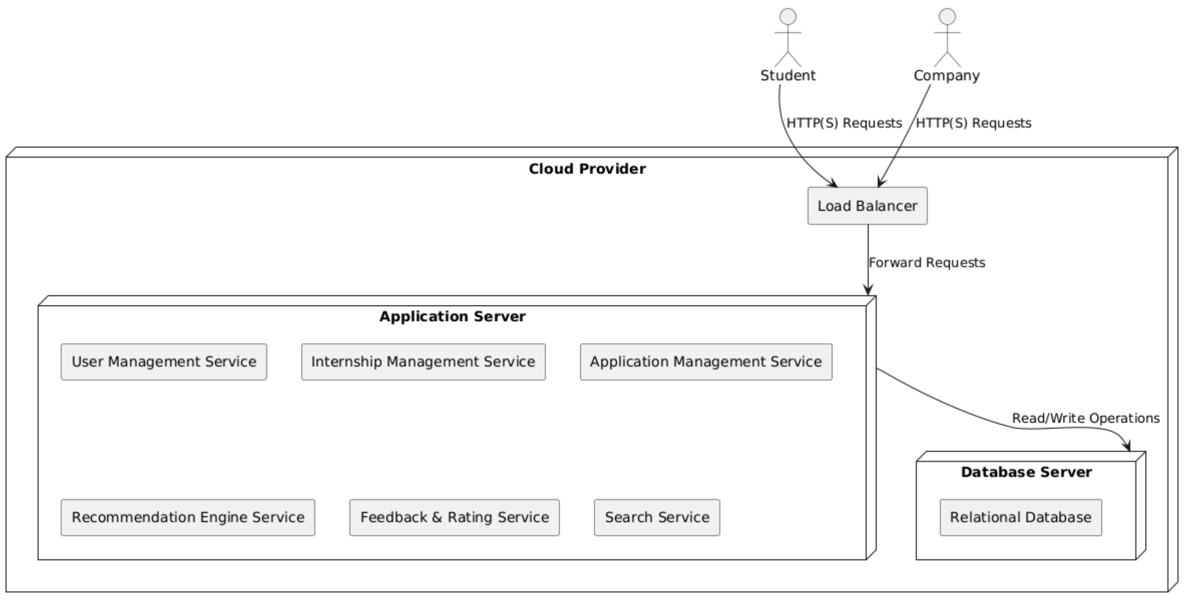


Fig. 2.4 Deployment View

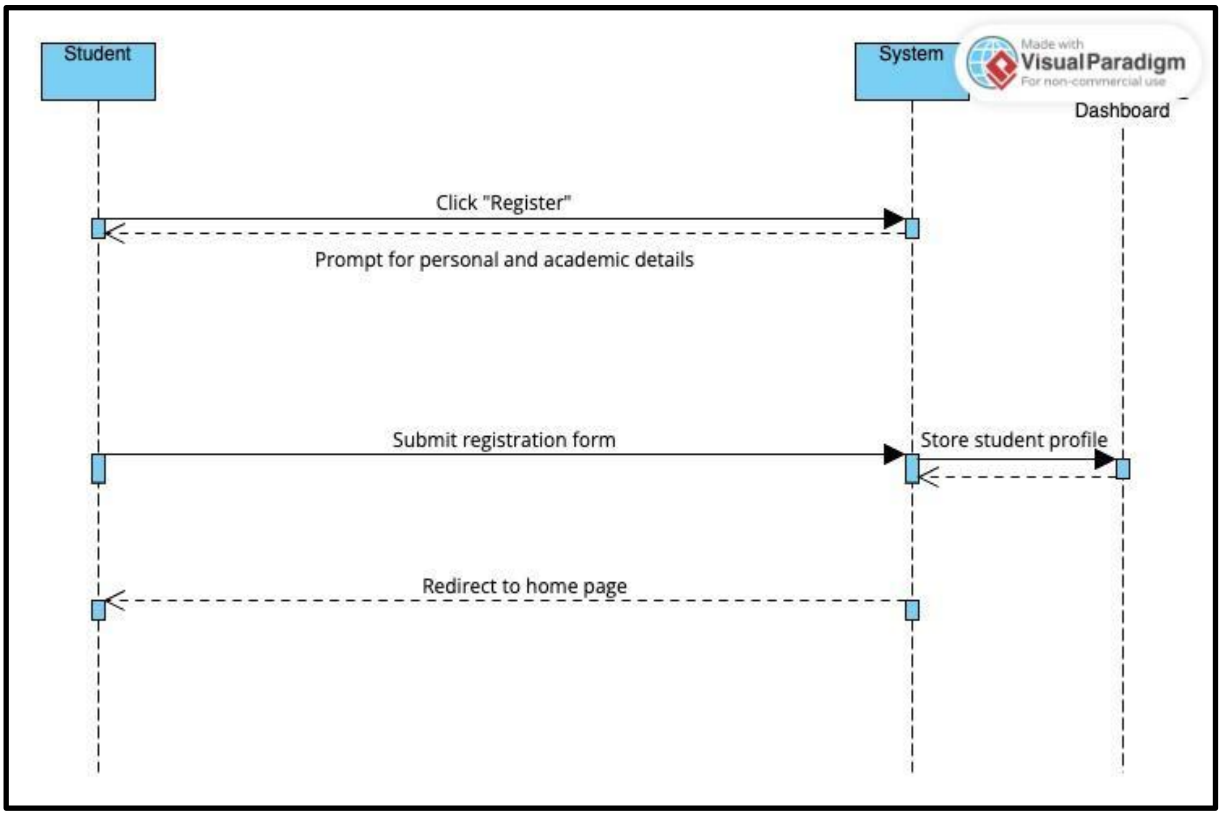
The S&C platform will be deployed on a cloud-based infrastructure, using services like AWS or Google Cloud Platform. This will ensure:

- Scalability: The ability to handle varying levels of user traffic and data storage needs.
- Availability: High uptime and resilience to failures.
- Security: Robust security measures to protect user data.

2.4. Runtime View

A typical user interaction with the platform might involve the following runtime steps:

[SD1] Student Registration



2.5 Student Registration Sequence diagram

Description: The Student Registration sequence diagram outlines the steps a student takes to register on the platform. The sequence begins with the student accessing the registration page and submitting personal and academic details. The system validates the information and creates the student's profile. Once the profile is successfully created, the system redirects the student to the login page. The database plays a key role in storing and retrieving data during the registration process.

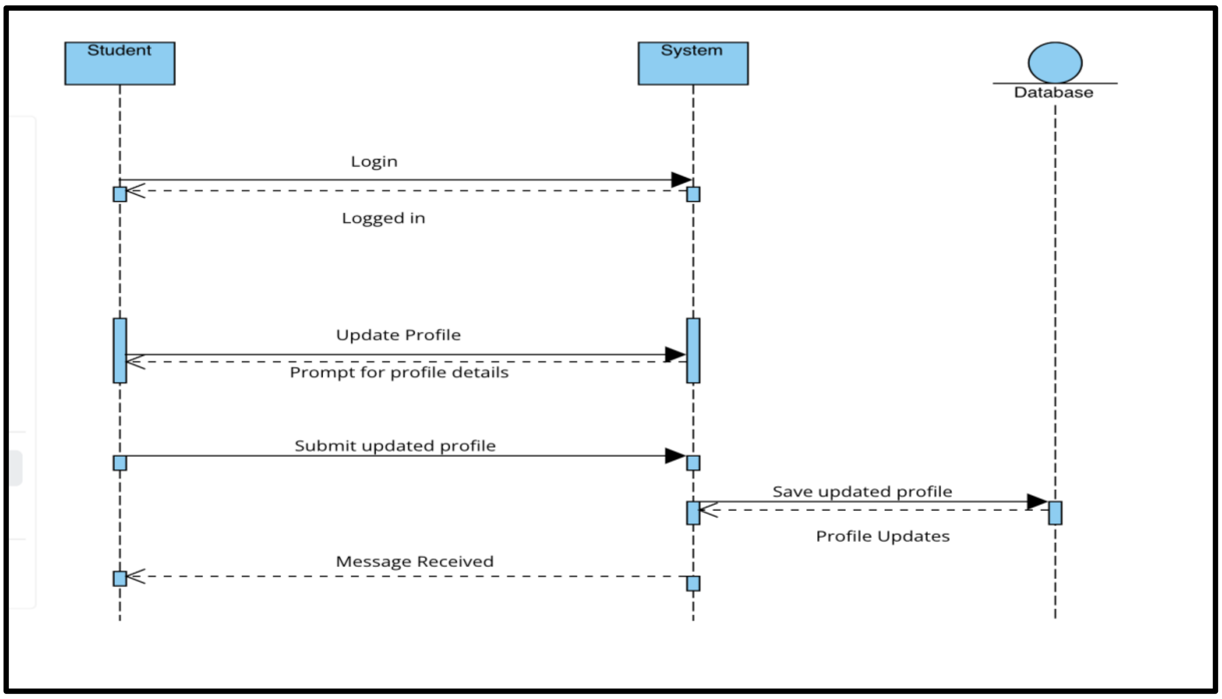
Actors Involved:

- Student
- System
- Database

Steps:

1. The student accesses the registration page and provides personal and academic details.
2. The system validates the data and stores it in the database.
3. After validation, the system creates a student profile.
4. The system notifies the student of successful registration and redirects them to the login page.

[SD2] Updating Profile



2.6 Update Profile diagram

Description: The Updating Profile sequence diagram details how users (students or companies) can update their profiles on the platform. The user submits an update request, and the system processes the changes. The updated information is stored in the database, and the user is notified of the successful update.

Actors Involved:

- User (Student/Company/University)
- System
- Database

Steps:

1. The user logs into the platform and navigates to the profile update section.
2. The user submits the changes to their profile.
3. The system processes the update request and validates the data.
4. The updated information is stored in the database.
5. The system notifies the user that the profile has been updated successfully.

[SD3] Recommend Internship

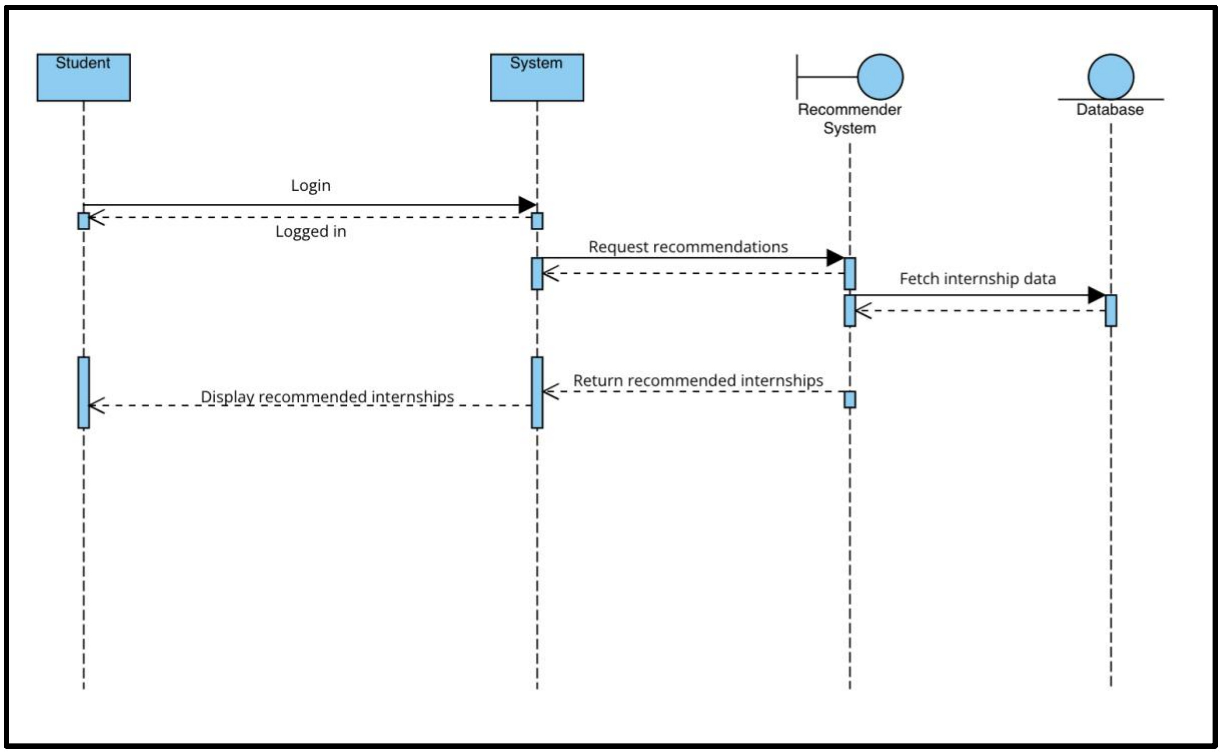


Fig. 2.7 Recommendation System

Description: The Recommend Internship sequence diagram describes the process through which the system suggests suitable internships to a student based on their resume. The system analyzes the student's profile and matches it with available internship listings, providing a tailored list of recommended internships.

Actors Involved:

- Student
- System
- Recommender system
- Database

Steps:

1. The student logs into the platform and requests internship recommendations.
2. The system analyzes the student's resume and profile.
3. The system compares the student's profile with available internships.
4. The system generates a list of recommended internships.
5. The recommended internships are displayed to the student.

[SD4] Internship Application

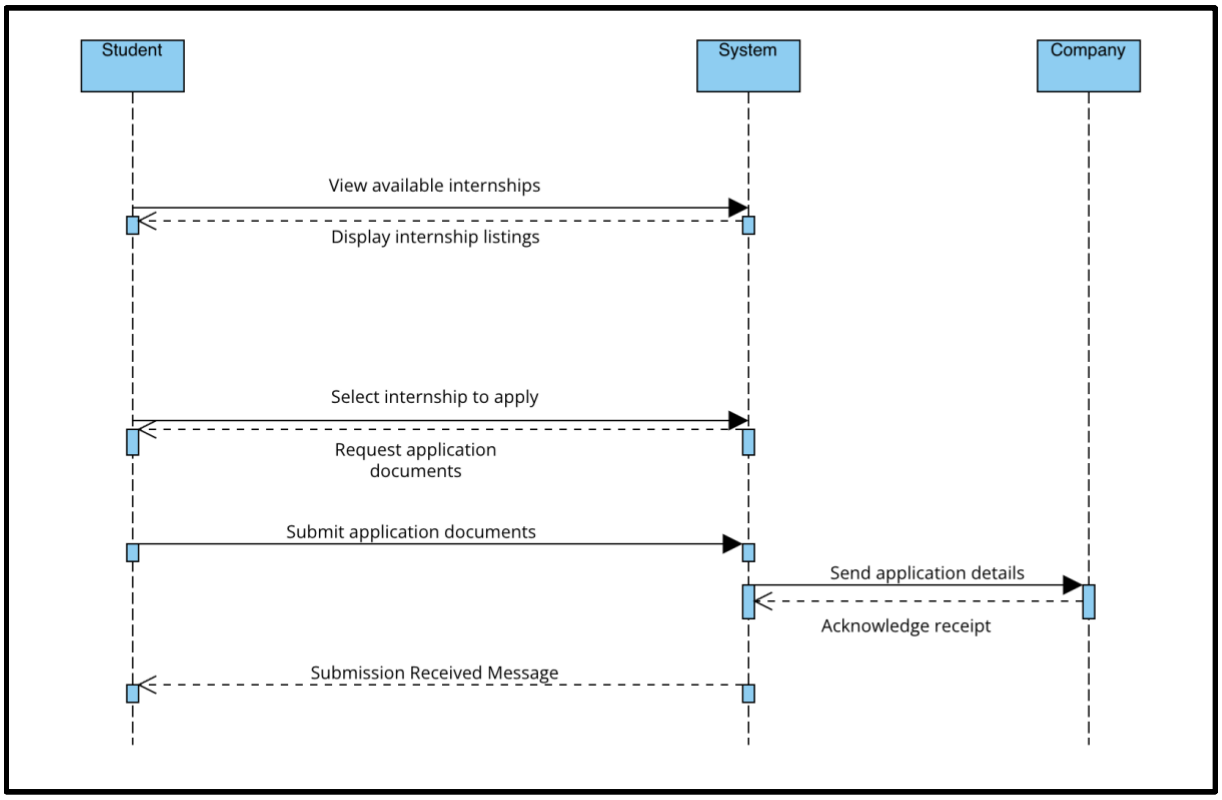

Student
System
Company
View available internships
Display internship listings
Select internship to apply
Request application documents
Submit application documents
Send application details
Acknowledge receipt
Submission Received Message

Fig.2.8 Internship Sequence diagram

Description: In the Internship Application sequence diagram, a student applies for an internship. After browsing available internships, the student selects the desired internship and uploads the necessary documents. The system processes the application and notifies the company about the new application. The student is also notified about the status of their application submission.

Actors Involved:

- Student
- System
- Company

Steps:

1. The student browses internship listings and selects one to apply for.
2. The system prompts the student to upload required documents.
3. The student submits the application along with the documents.
4. The system notifies the company of the new application.
5. The student receives a confirmation of the successful submission.

[SD5] Internship Posting

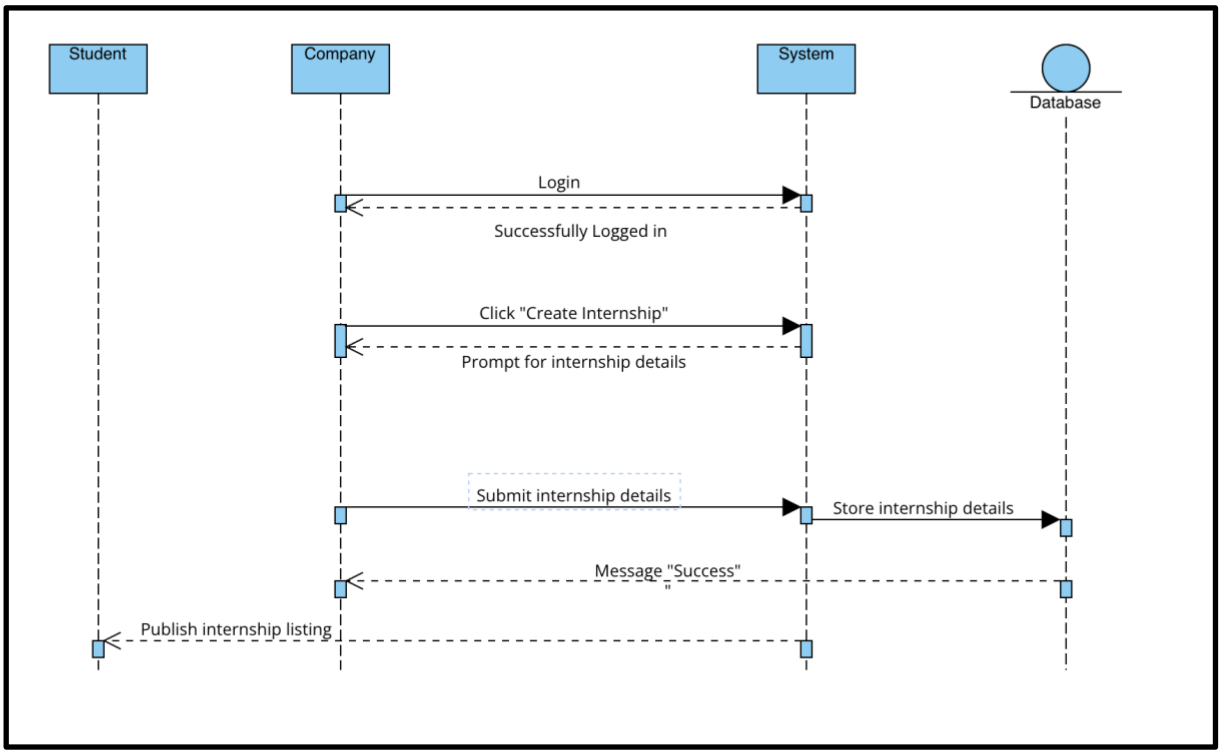

Student
Company
System
Database
Login
Successfully Logged in
Click "Create Internship"
Prompt for internship details
Submit internship details
Store internship details
Message "Success"
Publish internship listing

Fig. 2.9 Internship posting by Company

Description: The Internship Posting sequence diagram describes the process in which a company posts an internship on the platform. The company logs in, clicks on the "Create Internship" button, and fills out the necessary details like position, qualifications, location, etc. The system processes the details and publishes the internship listing, making it available for students to apply.

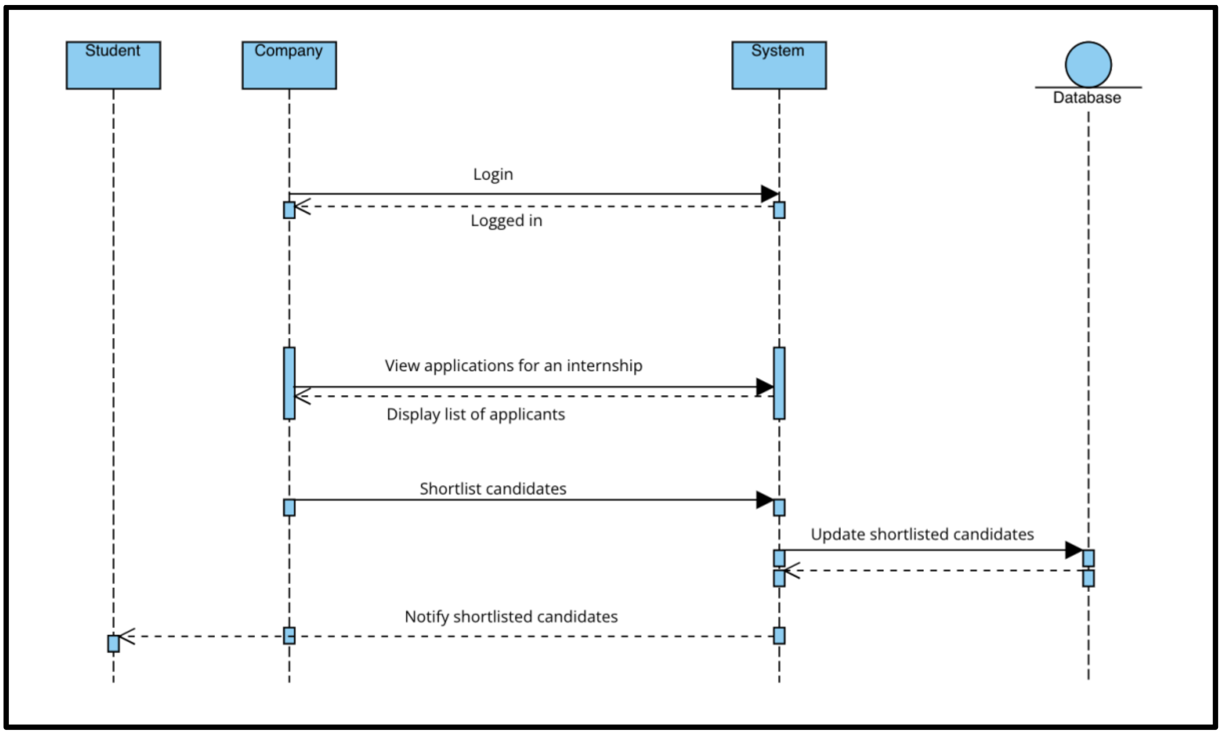
Actors Involved:

- Company
- System
- Student
- Database

Steps:

1. The company logs into the platform and clicks the "Create Internship" button.
2. The system prompts the company to input the internship details.
3. The company submits the form with the internship details.
4. The system processes the details and posts the internship on the platform.

[SD6] Shortlisting Candidates


Student
Company
System
Database
Login
Logged in
View applications for an internship
Display list of applicants
Shortlist candidates
Update shortlisted candidates
Notify shortlisted candidates

2.10 Shortlisting candidates diagram

Description: In the Shortlisting Candidates sequence diagram, the company reviews the applications for an internship. The company shortlists candidates based on their resumes and application materials. The system then updates the application status and sends notifications to the shortlisted candidates.

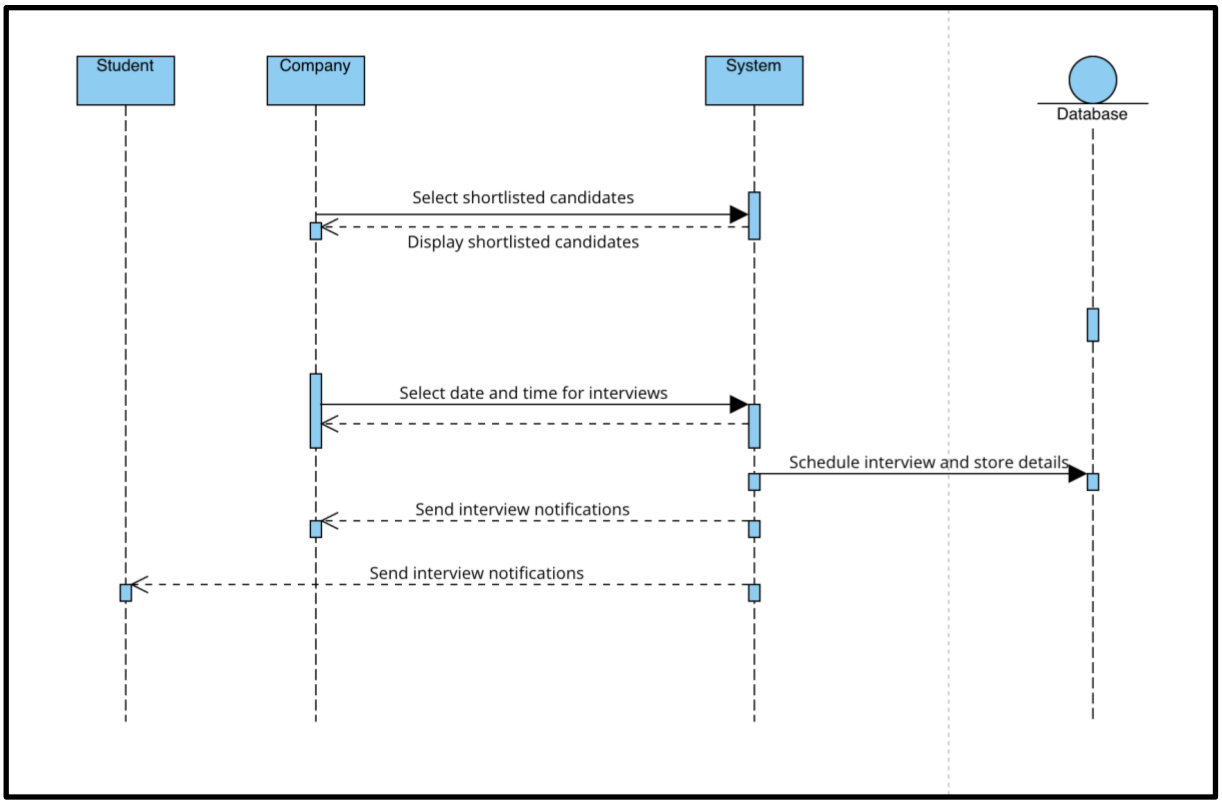
Actors Involved:

- Company
- System
- Database
- Student

Steps:

1. The company logs into the platform and views the list of applicants.
2. The company reviews the application materials and shortlists candidates.
3. The system updates the application status for the shortlisted candidates.
4. The system sends notifications to the shortlisted candidates.

[SD7] Scheduling Interview


Student
Company
System
Database
Select shortlisted candidates
Display shortlisted candidates
Select date and time for interviews
Schedule interview and store details
Send interview notifications
Send interview notifications

2.11 Scheduling interview diagram

Description: The Scheduling Interview sequence diagram outlines the process of scheduling interviews for shortlisted candidates. Once a candidate is selected, the company initiates the scheduling process by selecting an interview time. The system handles the scheduling and sends confirmation notifications to both the company and the candidate.

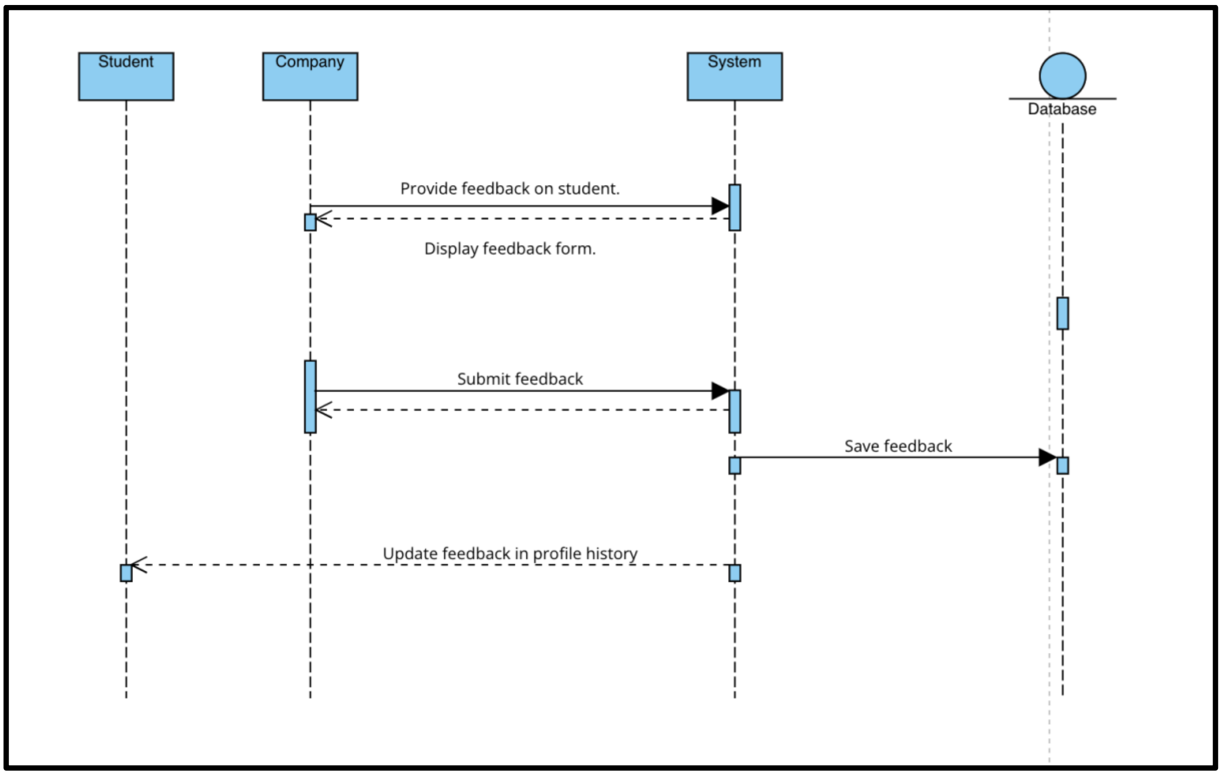
Actors Involved:

- Company
- Student
- System

Steps:

1. The company shortlists candidates and selects an interview date and time.
2. The system verifies the available slots and schedules the interview.
3. The system sends a confirmation notification to both the company and the candidate.
4. The candidate acknowledges the interview schedule.

[SD8] Submit Feedback



2.12 Submit Feedback diagram

Description: The Submit Feedback sequence diagram describes the process through which students and companies provide feedback after the completion of an internship. The company submits feedback regarding the student's performance, and the student provides feedback on the internship experience. The system stores and displays the feedback for both parties.

Actors Involved:

- Company
- Student
- System
- University

Steps:

1. The company submits feedback on the student's performance during the internship.
2. The student submits feedback on their experience during the internship.

3. The system stores the feedback and makes it accessible for review.
4. The system displays the feedback to both the company and the student.
5. Feedback may also be shared with the student's college.

2.5. Component Interfaces

The interfaces between the components will primarily be based on APIs (Application Programming Interfaces) using RESTful principles and JSON (JavaScript Object Notation) for data exchange. This will ensure:

- Loose Coupling: Components can be developed and updated independently.
- Flexibility: The platform can easily integrate with other systems or services.

User Management Component

- `request(register_user(username, password))`
- `authenticate_user(credentials)`
- `manage_profile(user_id, profile_data)`
- `logout(user_id)`
- `retrieve_user(user_id)`

Internship Management Component

- `create_internship_post(company_id, internship_details)`
- `edit_internship_post(post_id, updates)`
- `delete_internship_post(post_id)`
- `list_internships(filters)`
- `get_internship_details(post_id)`

Application Management Component

- `submit_application(student_id, internship_id, application_data)`
- `update_application_status(application_id, status)`
- `track_application(student_id)`
- `get_application_details(application_id)`

Recommendation Engine Component

- `get_recommended_internships(student_id, preferences)`
- `get_recommended_candidates(company_id, job_requirements)`

Feedback and Rating Component

- `submit_feedback(user_id, internship_id, feedback_data)`
- `get_feedback(internship_id)`
- `submit_rating(user_id, internship_id, rating)`
- `get_average_rating(internship_id)`

Communication Component

- `send_message(sender_id, recipient_id, message_content)`
- `get_messages(thread_id)`
- `mark_message_as_read(message_id)`
- `send_notification(user_id, notification_data)`

Search Component

- `search_internships(query, filters)`
- `search_profiles(query, filters)`
- `get_search_suggestions(query)`

Data Access Layer (DAL)

- `execute_query(query_string)`
- `fetch_data(query_parameters)`
- `insert_data(table_name, data_object)`
- `update_data(table_name, data_object)`
- `delete_data(table_name, record_id)`

2.6. Selected Architectural Styles and Patterns

3-Tier Architecture

Chosen for its:

- **Scalability:**
  Each tier can be scaled independently to meet demand, allowing the system to handle varying loads in different layers (e.g., database, application logic, and presentation). This is essential for cloud-based platforms where traffic can fluctuate.
- **Maintainability:**
  The separation of concerns makes code easier to manage and refactor. Updates or fixes in one tier can be applied without affecting other layers, reducing the risk of bugs.
- **Security:**
  Clear boundaries between the tiers (e.g., separating the database from the application logic) help enforce security policies. For instance, the database can be isolated from direct access by the front-end, requiring authentication via an application layer.

Additional Considerations:

- **Performance:**
  The 3-tier architecture can be optimized for performance by caching frequently accessed data in the application layer and using load balancing between servers in the application tier.

MVC Pattern (Model-View-Controller)

Provides:

- **Organized Structure:**
  Separates concerns into three distinct components (Model, View, Controller), making the codebase more organized and manageable. This is particularly useful for complex platforms like S&C, where the business logic, user interface, and database interactions are distinct.
- **Reusability:**
  Individual components (e.g., models or views) can be reused across different parts of the application. This increases development speed and consistency.
- **Testability:**
  The MVC pattern makes it easier to write unit tests for individual components. For instance, models can be tested independently of the user interface (view),

and controllers can be tested without worrying about the specific implementation of the view.

Facade Pattern

Offers:

- **Simplified Interface:**
  The facade pattern hides the complexity of subsystems (e.g., communication with the database, external APIs) from the user. This results in a clean, simplified interface for interactions between components.
- **Improved Readability:**
  By consolidating calls to multiple subsystems into a single method, the code becomes easier to understand and maintain. This is especially useful for complex workflows like internship management or user registration, where multiple backend processes need to be executed.

Event-Driven Architecture:

An event-driven approach could be useful for decoupling actions, such as sending notifications or triggering updates in the recommendation engine, based on user actions (e.g., submitting an internship application).

2.7. Other Design Decisions

Data Storage:

- Relational database (PostgreSQL or MySQL) will store structured data like user profiles, internship postings, and applications.

Security:

- Implemented industry-standard security practices, including secure user authentication (e.g., OAuth 2.0), data encryption (both at rest and in transit), and protection against common web vulnerabilities (e.g., cross-site scripting, SQL injection).
- Regular security audits and penetration testing will be conducted to identify and address potential vulnerabilities.
- Adherence to GDPR and other relevant data protection regulations is paramount.

API Design:

- RESTful APIs will be used for communication between the platform's components and external systems.
- Well-documented API specifications will be provided to facilitate integration and third-party development.

3| Requirements Traceability

This section maps the requirements defined in the RASD to the components and design elements outlined in this Design Document.

The matrix should demonstrate how the design choices address each requirement, ensuring a complete and consistent system development process.

Component	**Requirements**	**Description**
User Management Component	[R1] The system allows students to register using their email and academic information.	Handles user registration, login, and profile management for both students and companies.
	[R2] The system allows companies to register and create internship opportunities.	
	[R3] Students can log in to the platform to view internship listings, apply for internships, and track application progress.	
	[R4] Companies can log in to manage internship postings and review student applications.	
	[R13] Students can update their profiles, including personal details, resume, and academic information.	

	[R15] Companies can create and manage their profiles, listing information such as industry, size, and available internship programs.	
Internship Management Component	[R8] Companies can create internship listings with required qualifications, internship duration, location, and application deadline.	Responsible for creating, editing, managing, and displaying internship postings.
	[R9] Companies can update internship listings, change the status (open, closed, filled), and remove listings if necessary.	
	[R17] Companies can track which internships are still open and how many applications have been received for each.	
Application Management Component	[R5] Students can search for internships based on filters like skill requirements, location, and duration.	Manages the application process, including application submission, tracking, and status updates.
	[R6] Students can apply to multiple internships by submitting their resume, cover letter, and any required documents.	

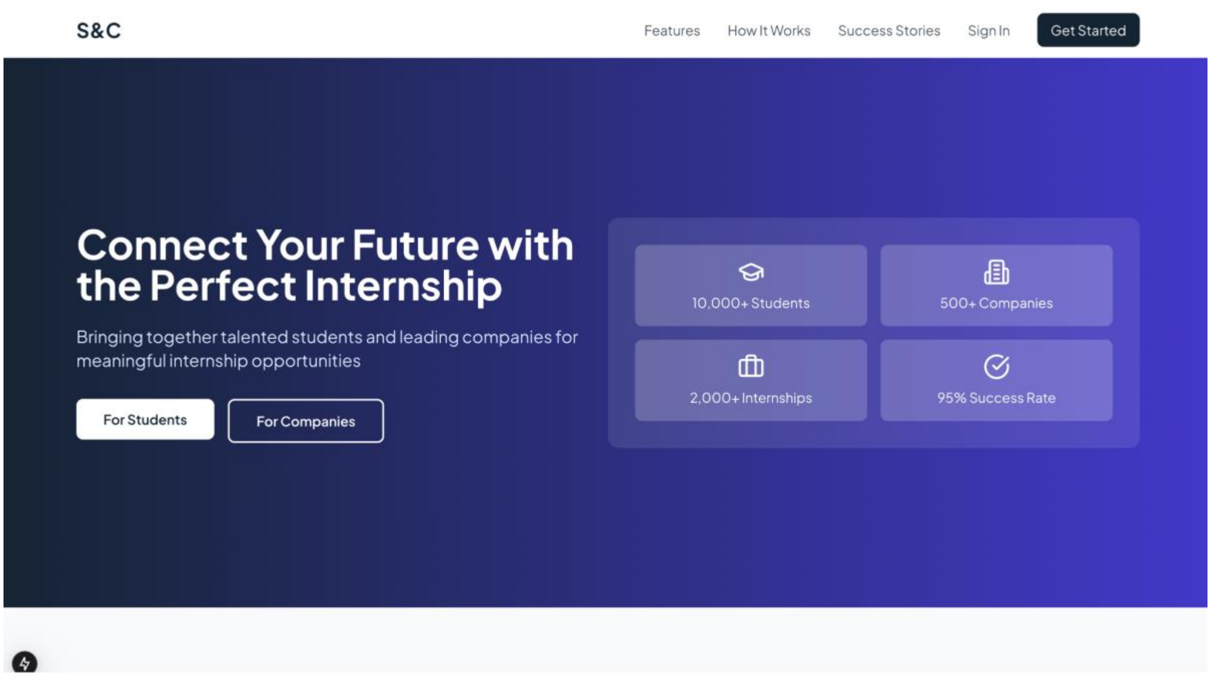
	[R7] Companies can view applications, filter candidates, and shortlist students for interviews.	
	[R10] The system sends notifications to students about the status of their applications (shortlisted, rejected, interview scheduled, etc.).	
	[R11] Students receive reminders about application deadlines and upcoming interviews.	
	[R14] The system tracks students' application history and feedback from companies, providing a complete view of each student's activities.	
Recommendation Engine Component	Sections 2.1.1 and 2.3, which describe the AI-based matching system to recommend suitable internships to students and potential candidates to companies.	Implements the recommendation algorithms to suggest relevant internships to students and suitable candidates to companies.
Feedback and Rating Component	[R12] Companies can send interview invites or requests for additional information to shortlisted candidates.	Handles the collection, storage, and display of feedback and ratings for internships and user experiences.
	[R16] Companies can review student profiles, view resumes, and provide	

	feedback or rejection notices.	
	Scenarios 6 and 7 in the Overall Description, which mention feedback and rating mechanisms.	
Search Component	[R5] Students can search for internships based on filters like skill requirements, location, and duration.	Allows users to search for internships and profiles using various filters.
Data Access Layer	Software System Attributes section, which highlights the importance of data security, emphasizing encryption and protection against cyber threats.	Provides an abstraction layer to interact with the database, ensuring data integrity and security. The source also stresses reliable performance and availability, implying the need for a robust Data Access Layer.

Table-1.1 Requirement mapping

This table demonstrates how the requirements outlined in RASD can be mapped to the components of a system designed to connect students seeking internships with companies offering them.

4| User Interface Design



Connect Your Future with the Perfect Internship

Bringing together talented students and leading companies for meaningful internship opportunities

Fig.4.1 Home page

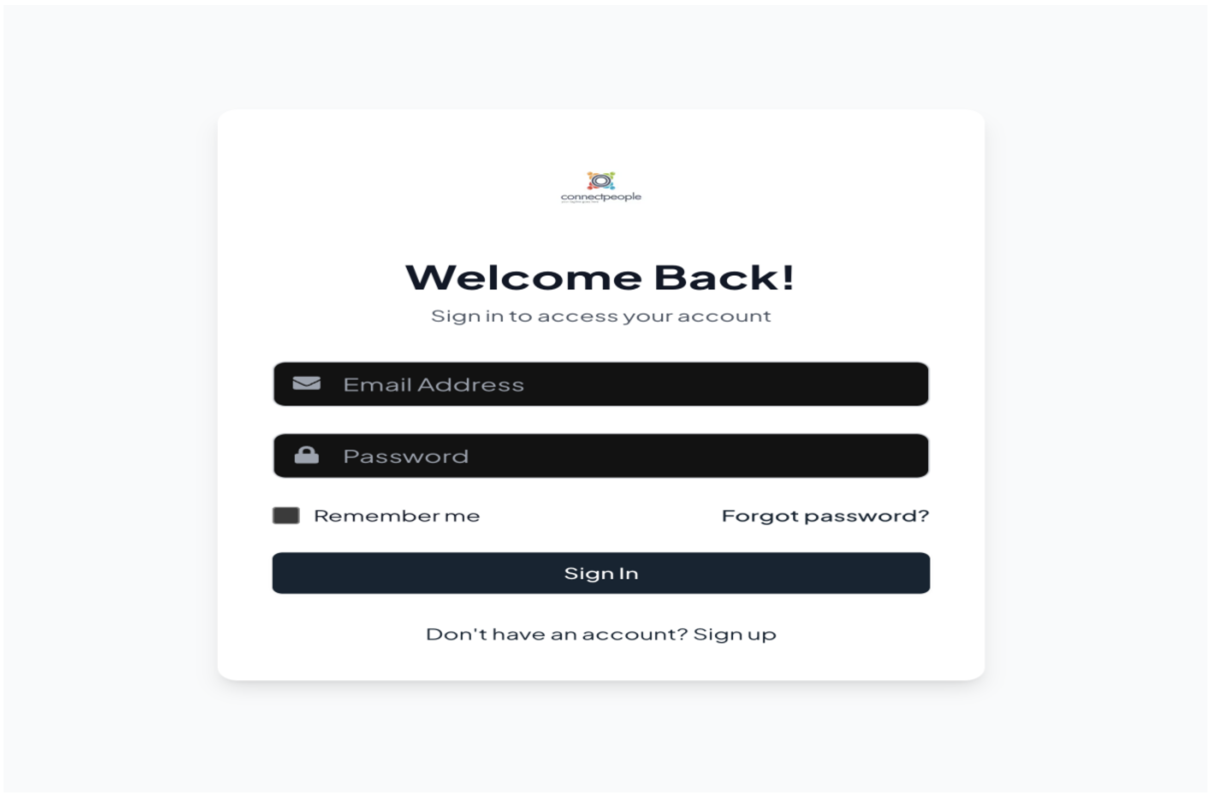
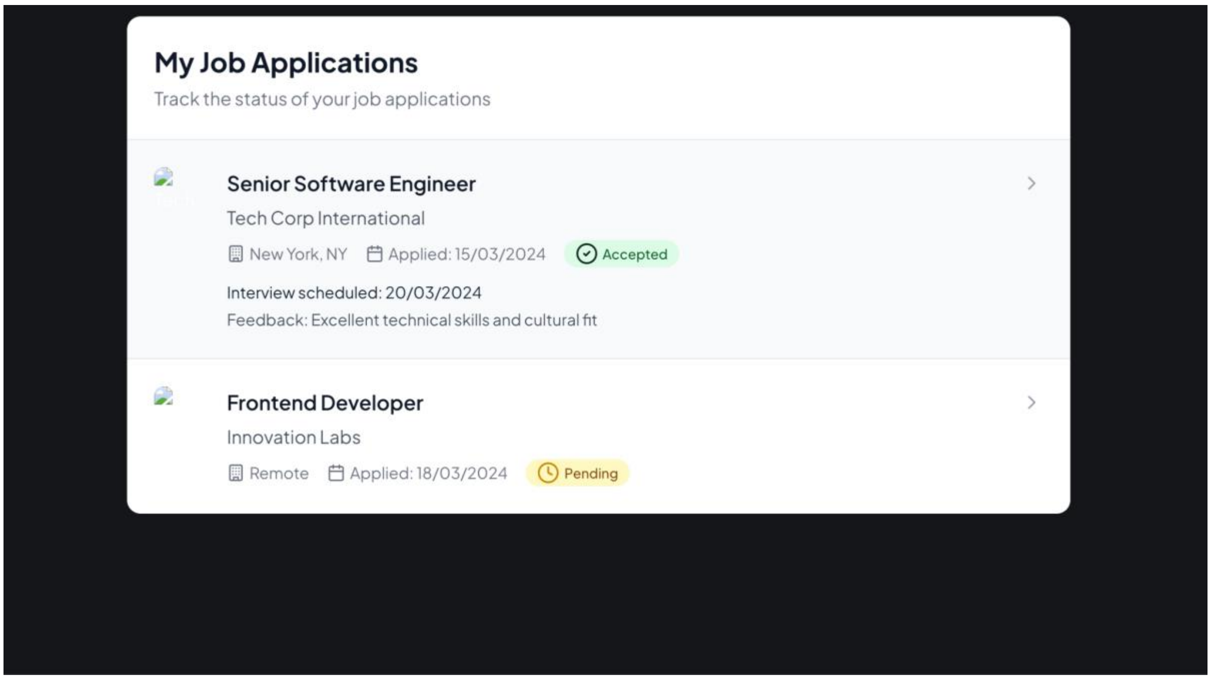


Fig.4.2 Login Page



My Job Applications

Track the status of your job applications

Senior Software Engineer

Tech Corp International

New York, NY | Applied: 15/03/2024 | Accepted

Interview scheduled: 20/03/2024

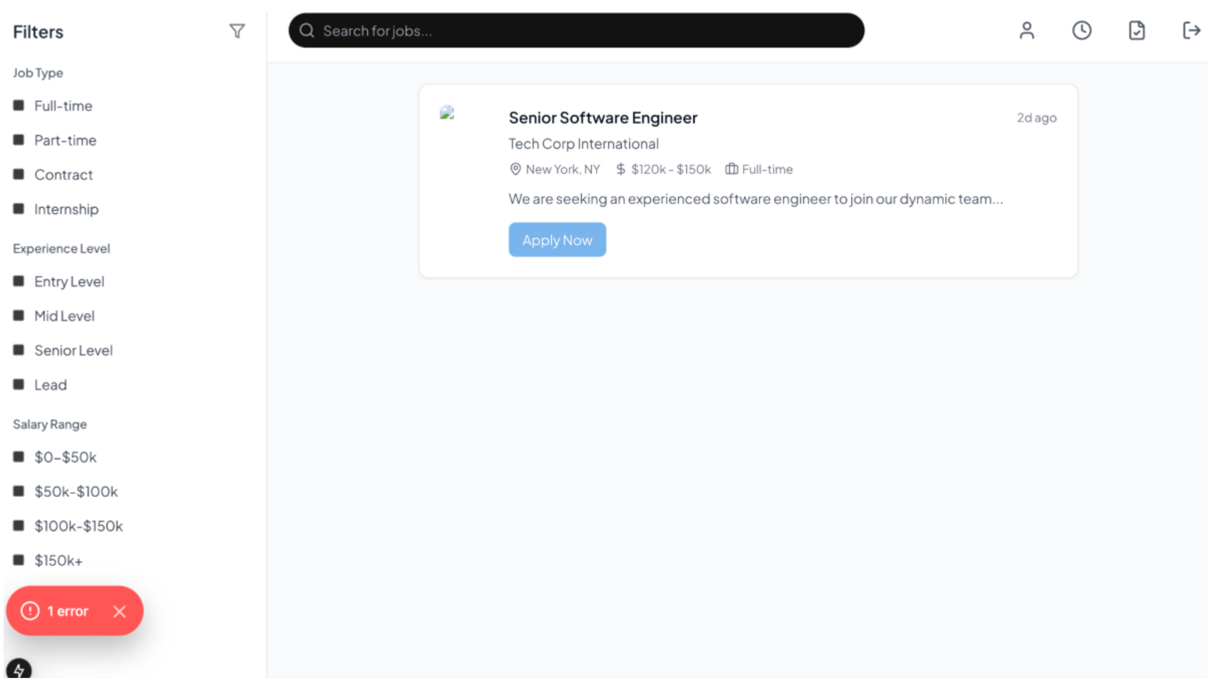
Feedback: Excellent technical skills and cultural fit

Frontend Developer

Innovation Labs

Remote | Applied: 18/03/2024 | Pending

Fig.4.3 Student Internship Applications



Filters

Job Type

- Full-time
- Part-time
- Contract
- Internship

Experience Level

- Entry Level
- Mid Level
- Senior Level
- Lead

Salary Range

- $0-$50k
- $50k-$100k
- $100k-$150k
- $150k+

1 error

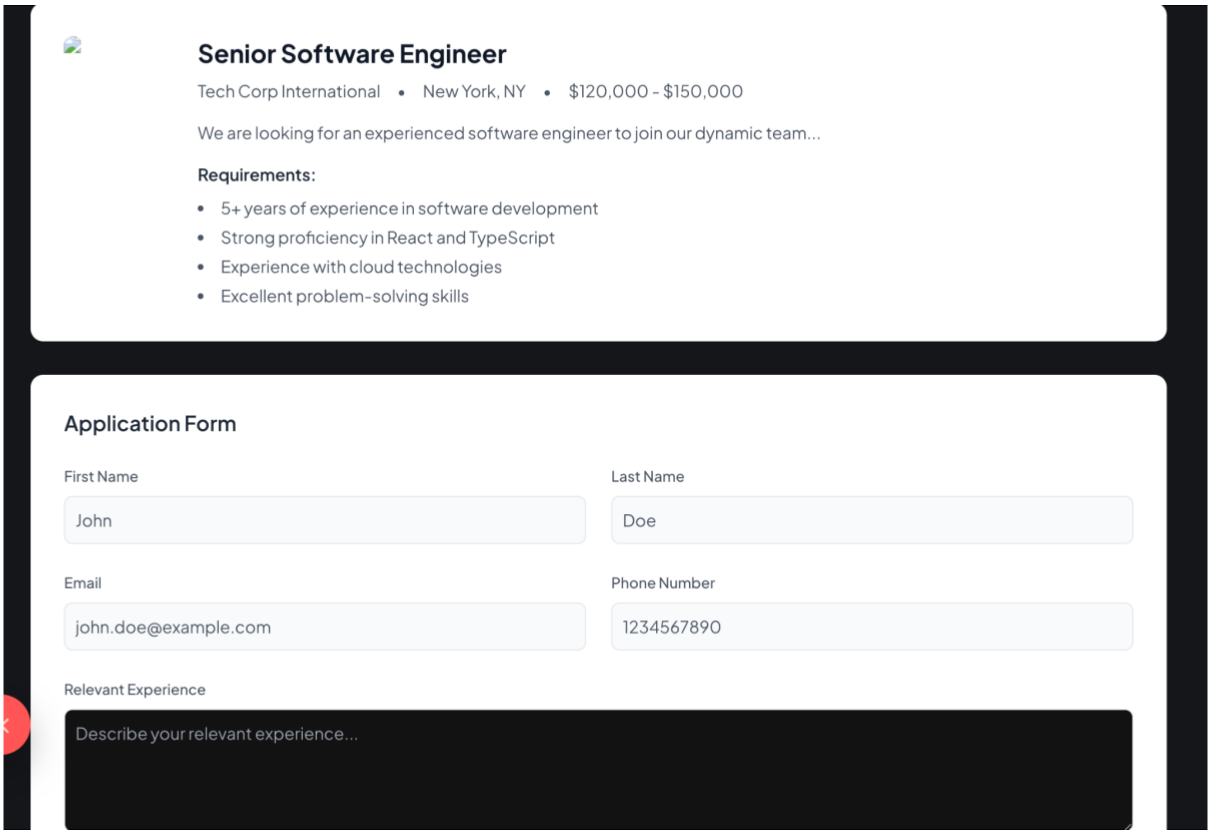
Search for jobs...

Senior Software Engineer
Tech Corp International
New York, NY · $120k - $150k · Full-time
2d ago

We are seeking an experienced software engineer to join our dynamic team...

Apply Now

Fig.4.4 Student Dashboard



Senior Software Engineer

Tech Corp International • New York, NY • $120,000 - $150,000

We are looking for an experienced software engineer to join our dynamic team...

Requirements:

- 5+ years of experience in software development
- Strong proficiency in React and TypeScript
- Experience with cloud technologies
- Excellent problem-solving skills

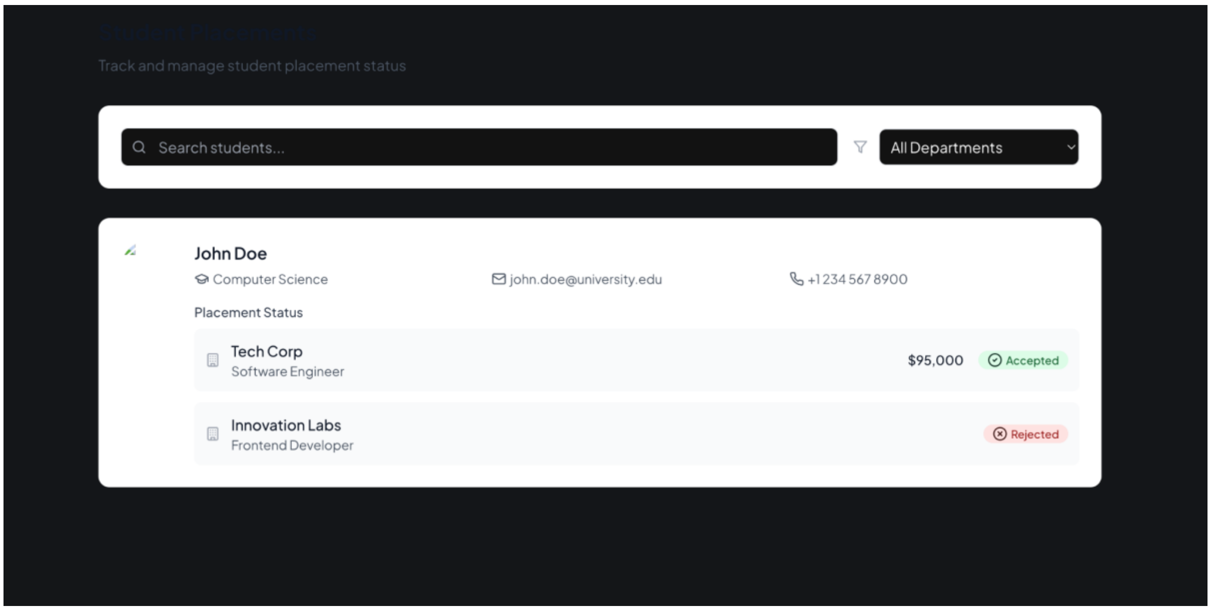
Application Form

First Name: John	Last Name: Doe
Email: john.doe@example.com	Phone Number: 1234567890

Relevant Experience

Describe your relevant experience...

Fig.4.5 Internship Application Page


Track and manage student placement status

Search students...

All Departments

John Doe

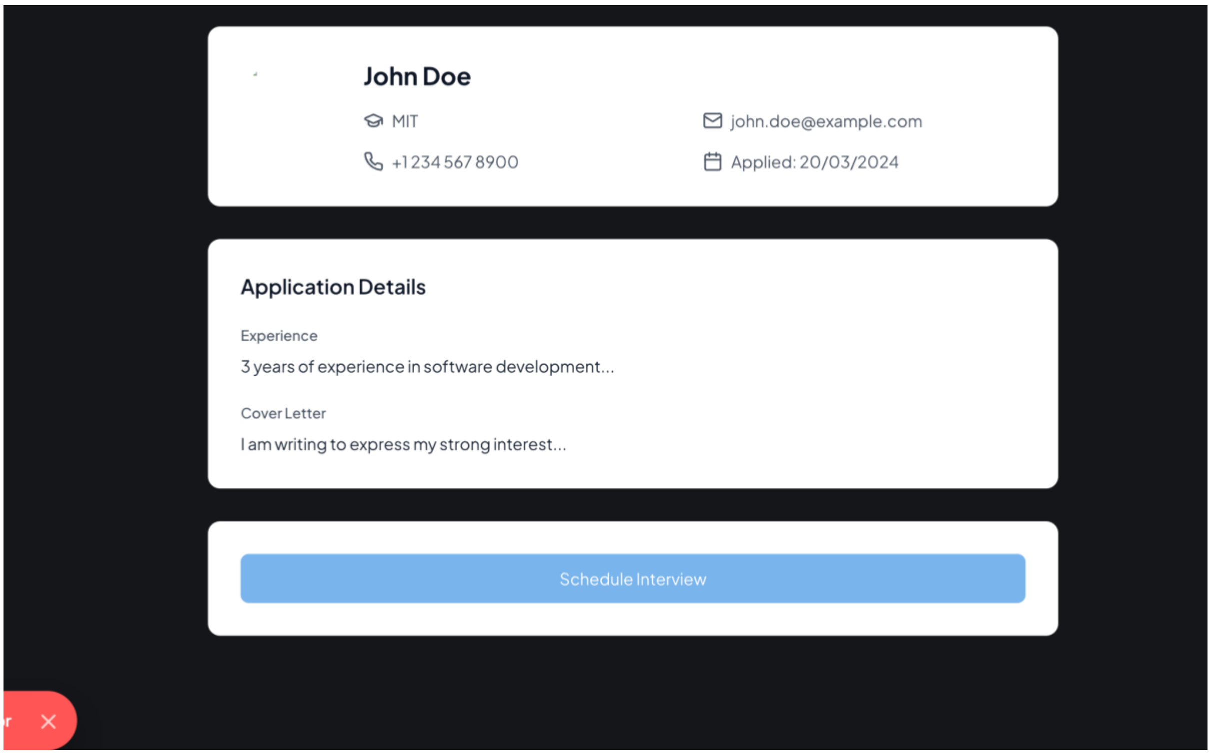
Computer Science | john.doe@university.edu | +1 234 567 8900

Placement Status

Tech Corp
Software Engineer
$95,000 Accepted

Innovation Labs
Frontend Developer
Rejected

Fig.4.6 University Students


John Doe

MIT | john.doe@example.com

+1 234 567 8900 | Applied: 20/03/2024

Application Details

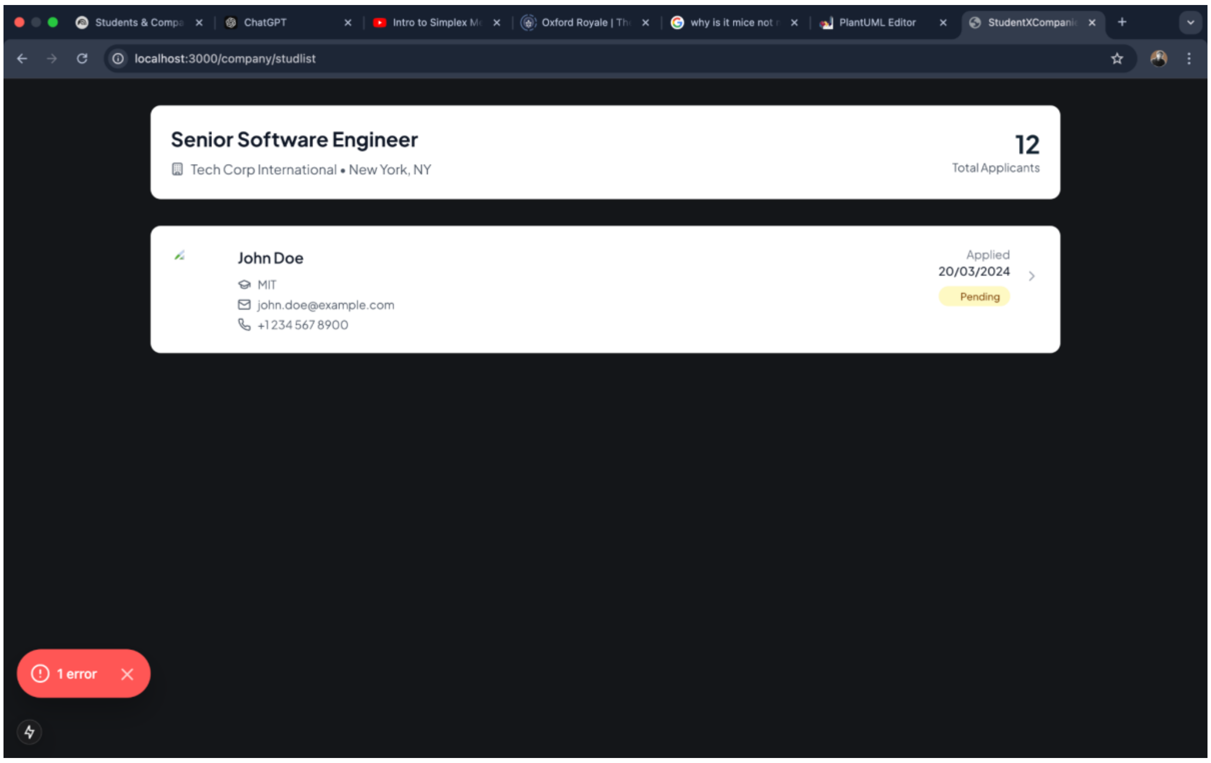
Experience

3 years of experience in software development...

Cover Letter

I am writing to express my strong interest...

Fig.4.7 Company view of Student Profile


Senior Software Engineer

Tech Corp International • New York, NY

12

Total Applicants

John Doe

MIT

john.doe@example.com

+1234 567 8900

Applied

20/03/2024

Pending

Fig.4.7 Company view of Internship page

5| Implementation, Integration and Test Plan

5.1. Overview

The S&C platform will be developed using an agile methodology, with an iterative and incremental approach. This allows for flexibility, continuous feedback, and adaptation to changing requirements.

Technologies:

- Programming Languages: Python (with frameworks like Django or Flask)
- Frontend Frameworks: Next.JS for building interactive and responsive user interfaces.
- Database: PostgreSQL for relational data storage.
- Cloud Infrastructure: AWS, Google Cloud Platform, or Azure for hosting and scalability.
- Version Control: Git for code management and collaboration.
- Machine learning

5.2. Implementation Plan

Here is a more detailed **implementation plan** in paragraph format with the features and their respective development steps:

User Registration and Authentication

The user registration and authentication process will begin with the creation of a form for both students and companies to register. This form will capture necessary personal and academic details for students and business details for companies. The form will include validation to ensure the data entered is accurate and complete. Once validated, user profiles will be created, and authentication will be handled using JWT tokens to secure sessions. The login system will allow users to access their accounts securely and manage their sessions throughout the platform.

Internship Posting by Companies

For companies to post internship opportunities, a dedicated form will be created where they can enter internship details such as position, qualifications, location, and other relevant information. This form will go through validation to ensure the data is correct and consistent. Once submitted, the internship details will be stored in the database and displayed on the platform for students to view and apply. The companies will also have the ability to edit or delete their internship postings as required.

Internship Application by Students

Students will have the ability to apply for internships through a submission form attached to each internship listing. The form will include fields for uploading necessary documents, such as resumes and cover letters. Once students submit their applications, the system will link the application with the respective internship listing and notify the companies of the new application. Students will also receive notifications about the status of their application, such as whether it was successfully submitted.

Recommendation Engine

The recommendation engine will be implemented to suggest relevant internships to students based on their profile data, including academic background, skills, and preferences. The system will analyze these attributes and match them with available internships, providing personalized suggestions. This will be done using a machine learning algorithm, and Python-based libraries like Scikit-learn will be used for training and deploying the recommendation model. Students will see these recommendations on their dashboard, making the platform more intuitive and user-friendly.

Shortlisting Candidates

Companies will be able to shortlist candidates after reviewing the applications they receive. The system will provide a streamlined interface for companies to filter applications and select the most suitable candidates based on criteria such as qualifications, experience, and application materials. Once candidates are shortlisted, the system will send notifications to inform them about their status. This feature ensures that companies can efficiently narrow down the pool of applicants and move forward with the hiring process.

Interview Scheduling

Once candidates are shortlisted, companies can schedule interviews directly through the platform. A dedicated scheduling interface will allow them to choose available time slots and send invitations to selected candidates. The system will integrate with calendar APIs, such as Google Calendar, to sync interview schedules and manage availability. Both students and companies will receive notifications with the interview details, ensuring smooth communication and coordination.

Profile Management and Update

Both students and companies will have the ability to update their profiles at any time. This will include fields for updating personal information, academic qualifications, work experience (for students), or business details (for companies). The profile update functionality will ensure that the platform remains dynamic and up-to-date, allowing users to modify their information as needed. Form validation will be implemented to ensure that all fields are correctly filled out before the changes are submitted and saved.

Feedback and Rating System

After completing an internship, students will be able to submit feedback and rate their experience. Similarly, companies will be able to rate the students they hired. This will help in improving the quality of matches between students and companies, as well as providing insights to future users. The feedback and rating system will also allow students and companies to leave comments on each other’s profiles and experiences. These ratings will be visible on internship listings to help guide future decisions.

Search Functionality

The platform will feature a robust search functionality that will allow users to search for internships and profiles using various filters. Students will be able to filter internships by location, duration, skills required, and other factors. Companies will also be able to search for candidates based on specific qualifications, skills, or availability. This search functionality will be optimized to ensure fast and accurate results, helping users find the best matches easily.

5.3. Component Integration and Testing

A combination of unit testing, integration testing, and system testing will be employed to ensure the quality and functionality of the S&C platform:

1.1 Unit Testing: Individual components will be tested in isolation to verify their functionality.

1. Frontend Testing (NextJS)

- **Objective**: Test individual components and functions to ensure they perform correctly in isolation.
- **Tools**: Mocha & tea, React Testing Library
- **Test Cases**:
    1. **Component Rendering**: Ensure that each component renders correctly with various props.
    2. **State Management**: Test whether the component's state updates correctly when triggered by user actions.
    3. **Event Handlers**: Test event handlers (e.g., button clicks, form submissions) to ensure they trigger the expected actions.
    4. **API Calls**: Mock and test API calls to check if components handle the responses correctly.
    5. **Error Handling**: Ensure the component handles edge cases and displays appropriate error messages.

2. Backend Testing (Django with pytest)

- **Objective**: Test individual components of the backend (e.g., models, views, serializers, forms) to ensure that they work correctly.
- **Tools**: pytest, pytest-django, FactoryBoy (for data creation)
- **Test Cases**:
    1. **Model Tests**:

- Test model methods and properties (e.g., `__str__`, custom methods).
- Ensure database constraints (e.g., uniqueness, nullability) are enforced.
    1. **View Tests**:
        - Test that views return correct status codes and data (e.g., 200 OK for successful requests, 404 for not found).
        - Test that permissions and authentication are correctly enforced.
    2. **Serializer Tests**:
        - Test that serializers correctly validate incoming data.
        - Test that they serialize/deserialize data accurately.
    3. **Form Tests**:
        - Test form validation and data saving functionality.

Feature Testing Plan for S&C Platform:

Feature testing is crucial to ensure that specific functionalities of the system are thoroughly tested. It ensures that the system behaves as expected and meets user requirements for each feature. For each feature, I'll provide the PlantUML code for the respective diagram and describe the feature testing process.

Frontend (NextJS) Feature Testing

Feature 1: Student Registration

Objective: Test the registration process for new users, ensuring the system creates a valid student profile and redirects the user to the login page.

Feature Testing Steps:

1. Fill out the registration form with valid data.
2. Submit the form and verify the student profile is created in the database.
3. Validate that the user is redirected to the login page.
4. Test the error handling for invalid data entry.

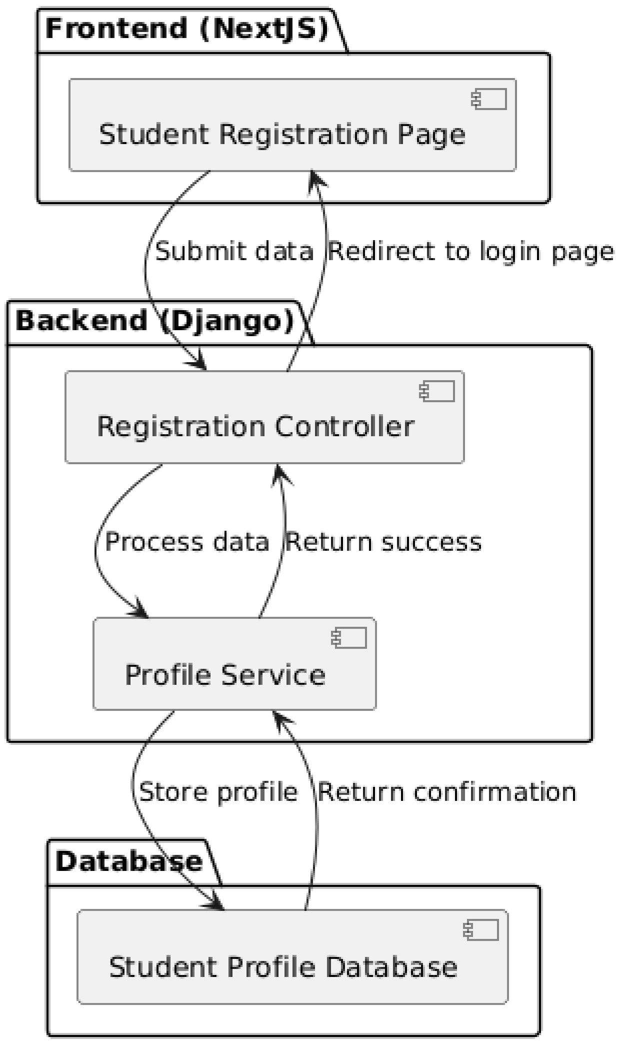


Fig. 5.1- End to End Student Registration

Feature 2: Internship Application

Objective: Test the process where a student applies for an internship and the internship application is submitted to the company.

Feature Testing Steps:

1. Browse available internships and select one.
2. Upload necessary documents and submit the application.
3. Verify that the system notifies the company of the new application.

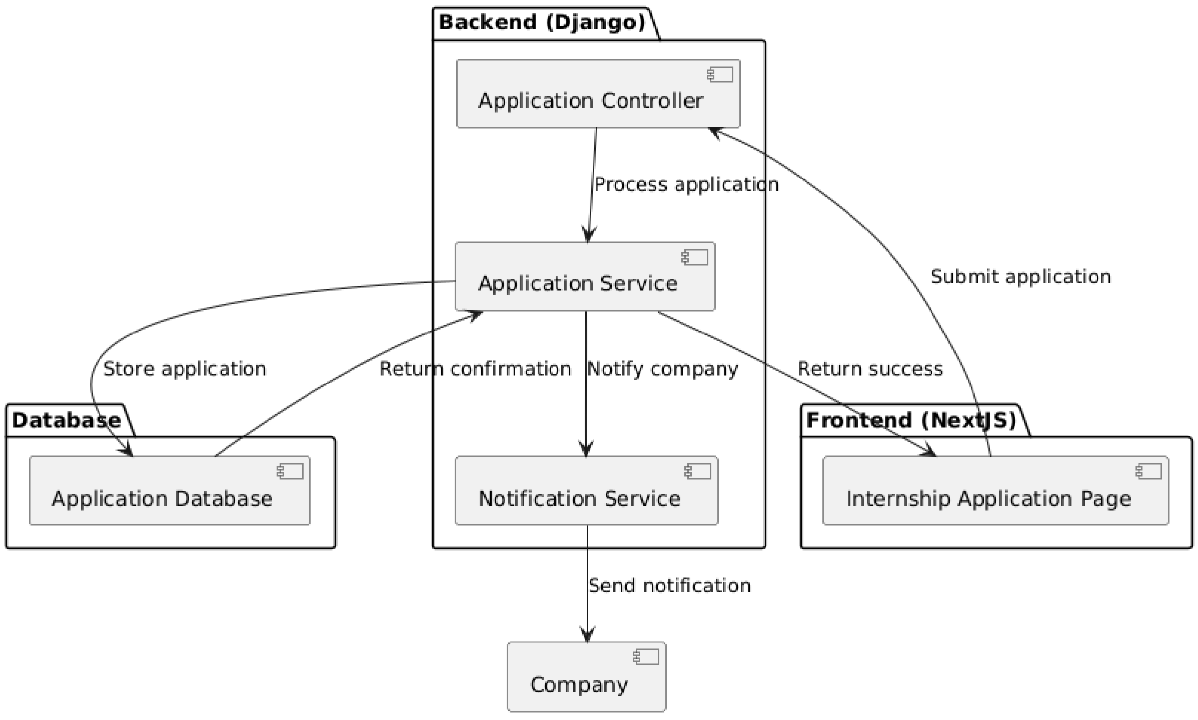


Fig. 5.2- End to End Internship Application

Feature 3: Profile Update

Objective: Test the functionality of updating the profile for both students and companies.

Feature Testing Steps:

1. Log in and access the profile page.
2. Modify profile details (e.g., name, email, etc.).
3. Submit the updated details and ensure that the system saves the changes.
4. Verify that the updated profile is correctly displayed.

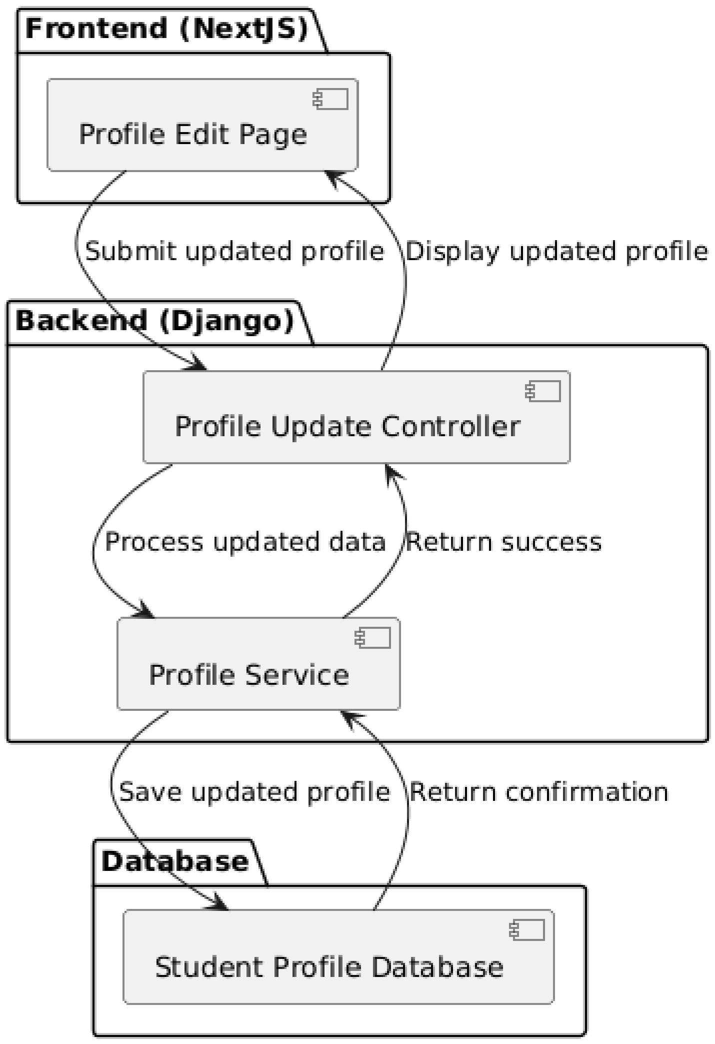


Fig. 5.3- End to End Profile Update

Backend (Django with pytest) Feature Testing

Feature 1: Internship Posting

Objective: Test the functionality of posting an internship by a company, verifying that the internship details are stored and displayed correctly.

Feature Testing Steps:

1. Log in as a company and navigate to the internship posting page.
2. Fill in the internship details.
3. Submit the internship and verify that the system stores it in the database.
4. Ensure the internship is visible to students on the platform.

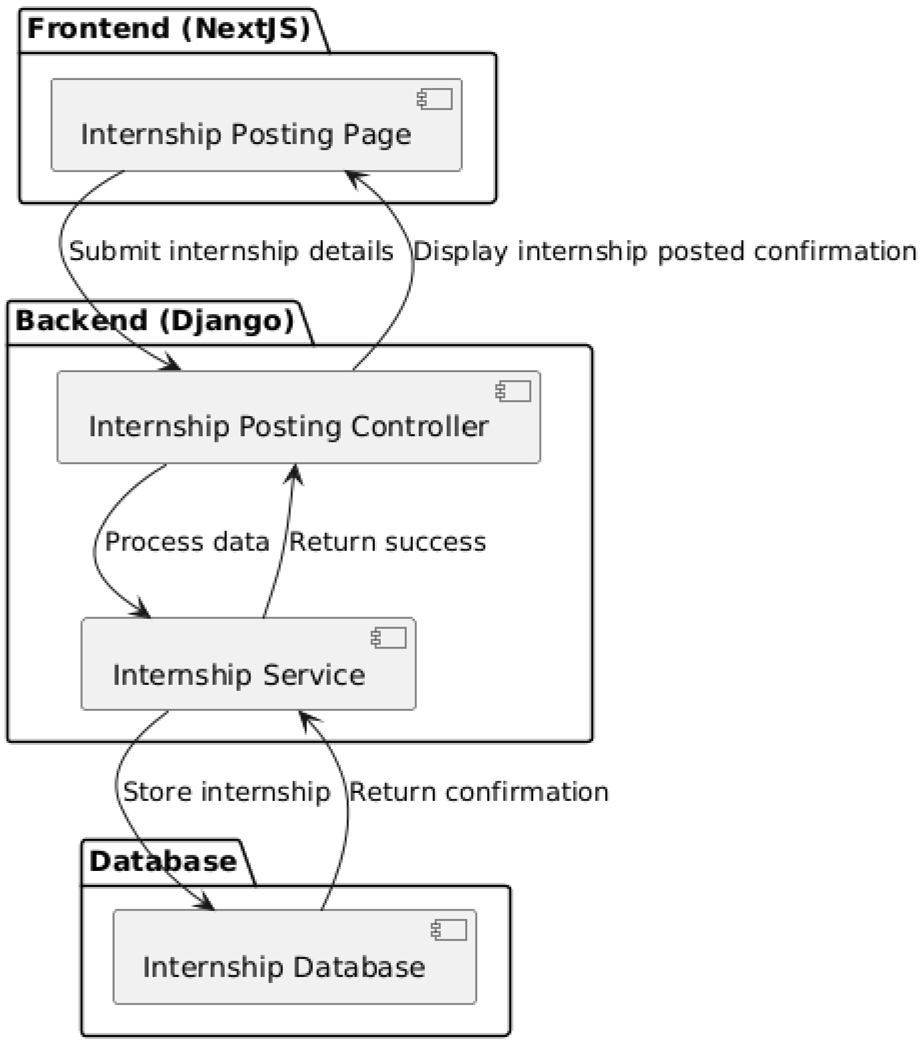


Fig. 5.4- End to End Internship posting

Feature 2: Shortlisting Candidates

Objective: Test the process where a company shortlists candidates for an internship and the system notifies the selected candidates.

Feature Testing Steps:

1. Log in as a company and access the list of internship applications.
2. Review applications and select candidates to shortlist.
3. Ensure that the system updates the candidates' status and sends a notification.

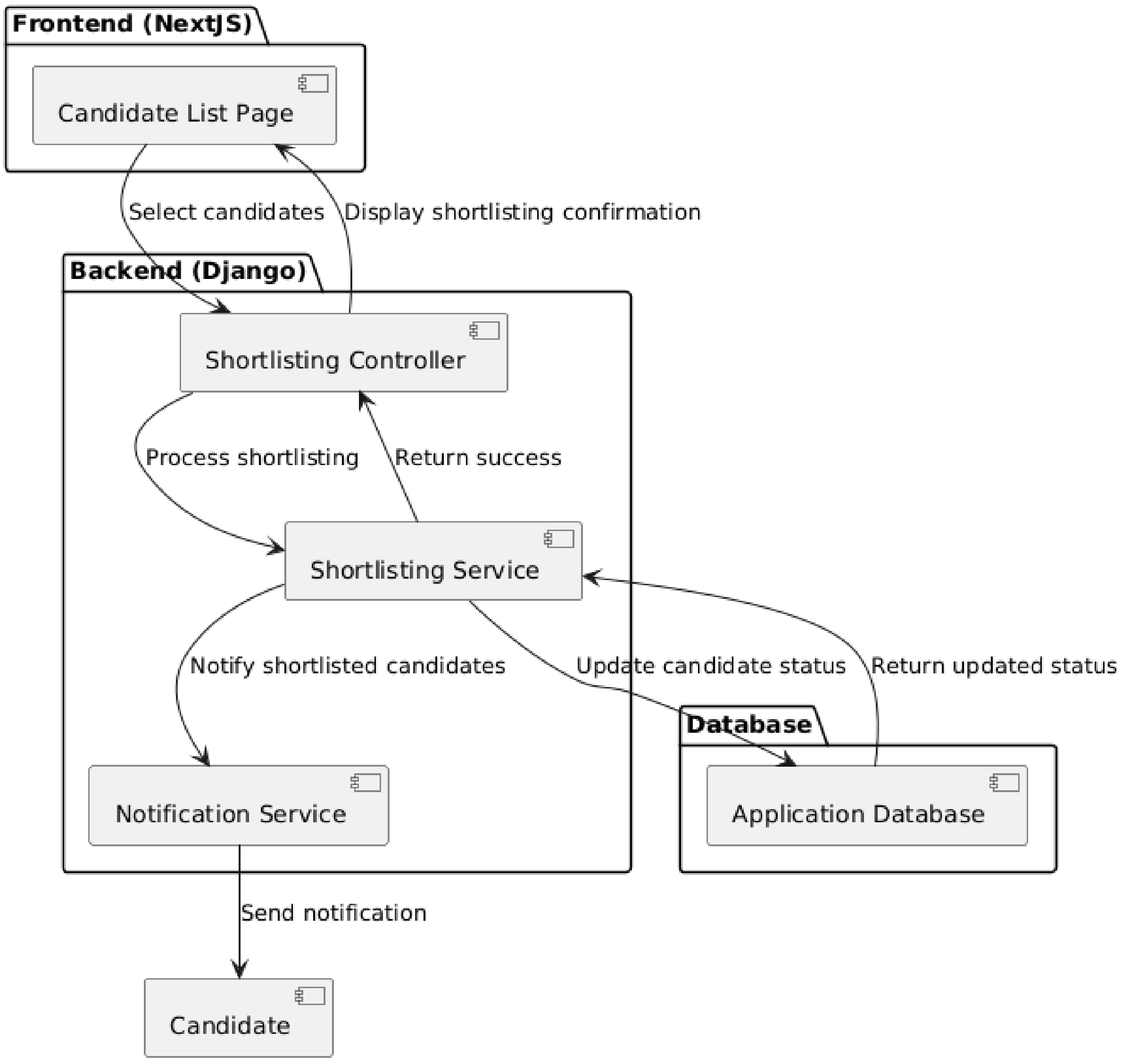


Fig. 5.5- End to End Candidate Shortlisting

Feature 3: Scheduling Interview

Objective: Test the functionality of scheduling an interview for shortlisted candidates, ensuring both the company and candidate are notified.

Feature Testing Steps:

1. Log in as a company and access the list of shortlisted candidates.
2. Select candidates for interview scheduling.
3. Set the interview time and submit the schedule.
4. Verify that both the company and the candidates are notified of the interview details.

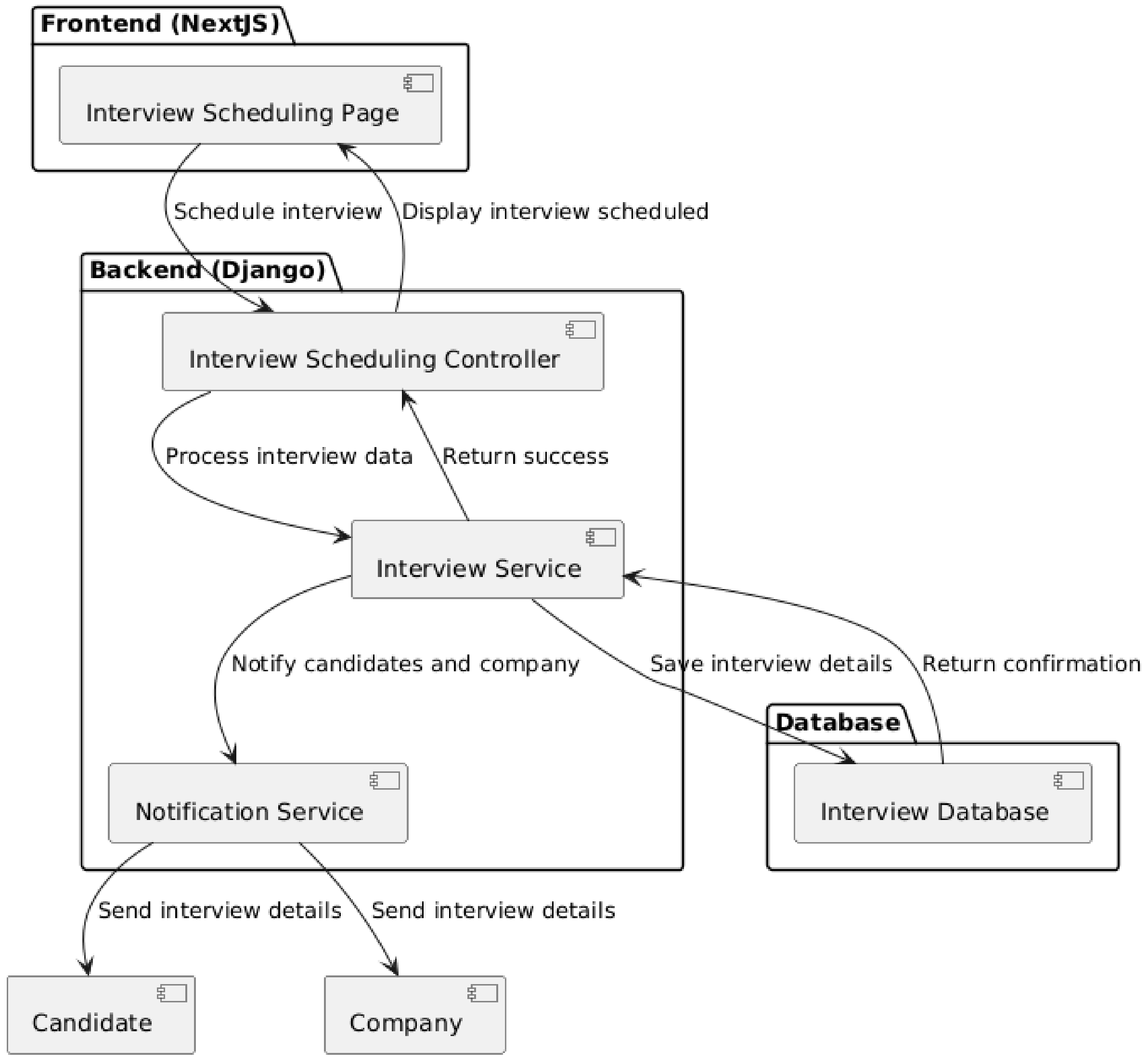


Fig. 5.6- End to End Scheduling Interviews

Feature 4: Submit Feedback

Objective: Test the functionality of submitting feedback after an interview, ensuring that the company, student, and college can submit and view feedback.

Feature Testing Steps:

1. After an interview, the company submits feedback about the student.
2. The student can also submit feedback about the interview process.
3. Colleges can view the feedback submitted by both parties.

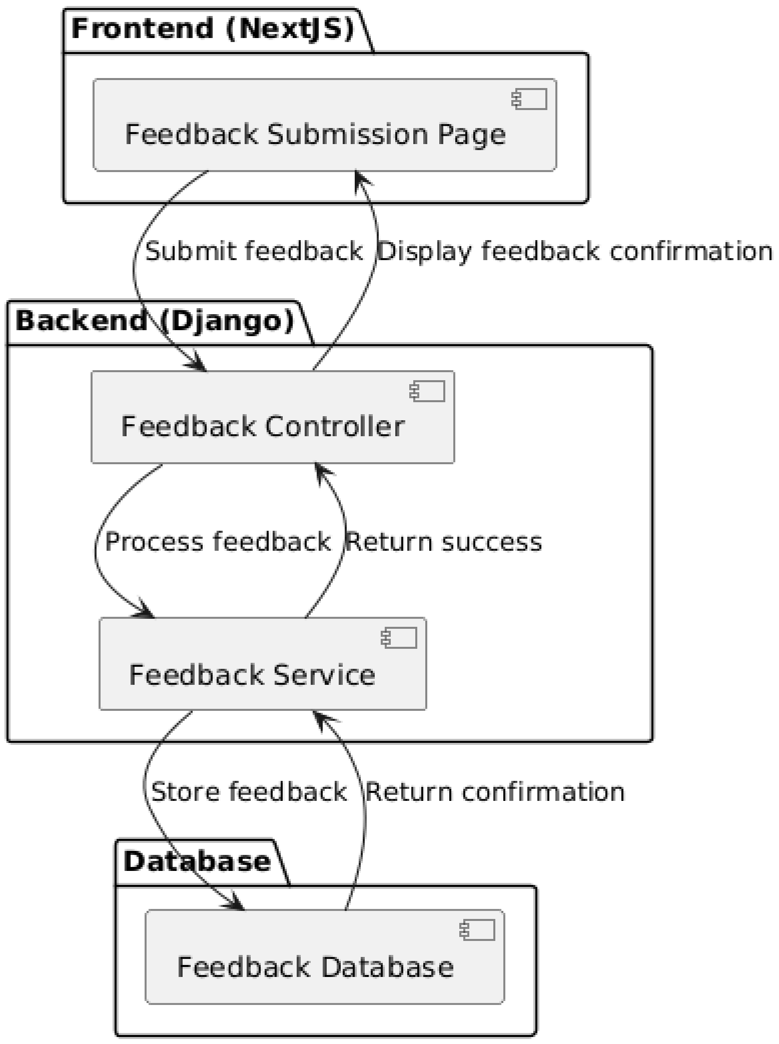


Fig. 5.7- End to End Submit Feedback

1.2 Integration Testing: Components will be tested together to ensure they interact correctly.

1. Frontend Testing (NextJS)

- **Objective**: Ensure different components work together as expected.
- **Tools**: Mocha & Tea, React Testing Library
- **Test Cases**:
    1. **Component Interactions**: Test interactions between components, such as parent-child interactions or sibling components.
    2. **State Propagation**: Test how state is propagated between components and if changes in one component reflect correctly in others.
    3. **Form Submissions**: Test end-to-end form submission to ensure the data is collected and sent to the backend.

4. **Routing**: Ensure navigation between different pages works as expected using NextJS routing features (e.g., `next/link`, `next/router`).

2. Backend Testing (Django with pytest)

Objective: Test how different components (models, views, serializers) interact with each other.

Tools: pytest, pytest-django

Test Cases:

1. **User Registration**:
   - Test that the registration API correctly creates a user and returns the expected response.
2. **Internship Application**:
   - Test the process of submitting an internship application, ensuring that data is correctly saved and notifications are sent.
3. **Internship Posting**:
   - Test the internship creation flow, ensuring that all required fields are processed and stored correctly.
4. **Shortlisting Candidates**:
   - Test that the company can shortlist candidates and that notifications are sent correctly.

System Testing: The entire system will be tested as a whole to verify that it meets the requirements and performs as expected under various scenarios.

5.4. System Testing

System testing will include:

- Functional Testing: Verifying that all features work as specified in the requirements.
- Performance Testing: Testing the platform's performance under expected load conditions.
- Usability Testing: Evaluating the user interface with real users to identify areas for improvement.
- Security Testing: Identifying and mitigating security vulnerabilities.
- Compatibility Testing: Ensuring the platform works correctly on different browsers and devices.

5.5. Additional Specifications on Testing

- Alpha Testing: Internal testing will be conducted with a limited group of users to gather initial feedback.
- Beta Testing: A public beta release will allow a wider audience to test the platform and provide feedback before the official launch.

6 | Effort Spent

In this part there is an overview of the time effort spent by each member of this team. Everyone has spent some time writing each section of this document and here it is visible the amount of time.

- Student 1

chapter	Effort(In hours)
1	10
2	10
3	10
4	5

- Student 2

chapter	Effort(In hours)
1	10
2	10
3	10
4	10

- Student 3

chapter	Effort(In hours)
1	10
2	15
3	10
4	8

7 | Bibliography

[1] **Unified Modeling Language (UML) - Object Management Group (OMG).** (2024).
UML 2.5 Superstructure Specification.
URL: https://www.omg.org/spec/UML/2.5/

- Provides official specifications for UML, including use case diagrams, class diagrams, sequence diagrams, and more, which are essential for documenting the S&C system.

[2]**Visual Paradigm.** (2024). *Visual Paradigm UML Tool.*
URL: https://www.visual-paradigm.com/

- A UML tool that can be used to create the various diagrams (use case, class, sequence, etc.) for the S&C platform.

8 | List of Figures

9 | List of Tables