

Florida International University
Knight Foundation School of Computing and Information Sciences

Software Engineering Focus

Final Deliverable

Project Title: Path Tracker

Team Members: Martin Archila, Nicolas Astros, Marco Chaparro, Alejandro Morales

Product Owner(s): Miguel Castro

Mentor(s): Masoud Sadjadi

Instructor: Masoud Sadjadi

Copyright (c) 2016 *Florida International University*

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Abstract

*This document provides a comprehensive overview of the development process for our capstone senior project, **Path Tracker**. It begins with an introduction to the project, followed by an analysis of the current system and the rationale for the proposed solution. The document then details the implemented user stories that define the application's core functionality, along with the pending user stories for future enhancements. The project plan is outlined, including the hardware and software resources utilized, as well as the sprint planning approach. System design aspects are thoroughly explained, covering architectural patterns, system and subsystem decomposition, and applied design patterns. Finally, the document concludes with an explanation of the system validation process, ensuring the application meets its intended goals and requirements.*

Table of Contents

Table of Contents

<i>Introduction</i>	<i>5</i>
<i>Current System</i>	<i>5</i>
<i>Purpose of New System.....</i>	<i>5</i>
<i>User Stories.....</i>	<i>6</i>
<i>Implemented User Stories</i>	<i>6</i>
<i>Pending User Stories</i>	<i>7</i>
<i>Project Plan</i>	<i>7</i>
<i>Hardware and Software Requirements.....</i>	<i>7</i>
<i>Hardware.....</i>	<i>7</i>
<i>Software: Front-End</i>	<i>7</i>
<i>Software: Back-End</i>	<i>7</i>
<i>Software: Database</i>	<i>8</i>
<i>Sprint Plan.....</i>	<i>8</i>
<i>System Design</i>	<i>15</i>
<i>Architectural Patterns</i>	<i>15</i>
<i>System and Subsystem Decomposition</i>	<i>15</i>
<i>Design Patterns.....</i>	<i>16</i>
<i>System Validation</i>	<i>16</i>
<i>Appendix</i>	<i>17</i>
<i>Appendix A – UML Diagrams</i>	<i>17</i>
<i>Appendix B – User Interface Design.....</i>	<i>18</i>
<i>Appendix C – Sprint Review Reports</i>	<i>21</i>

Introduction

The **Path Tracker**, a web-based application, was conceived to address a common challenge faced by students: managing the overwhelming and often chaotic process of tracking job and internship applications. For many, keeping tabs on application statuses, scheduled interviews, and other important updates can be stressful and disorganized. To tackle this issue, we developed a solution that simplifies and streamlines this process.

This application enables users to create personalized profiles and efficiently manage their job applications through a user-friendly dashboard. A key feature of the system is a Google Chrome extension, which allows users to seamlessly save application details directly from job postings. By serving as a centralized hub for all job-related information, the Job Application Organizer helps users stay on top of their applications, upcoming events, and status updates, promoting a more organized and less stressful job search experience.

Current System

Currently, the primary product on the market offering functionality similar to our proposed application is Simplify, a Google Chrome extension. Simplify allows users to autofill online job applications and track the jobs they have applied to. Additionally, it provides supplementary features such as resume assistance and job discovery tools. While Simplify aims to support users throughout their entire job application journey, its broader focus means it is not exclusively dedicated to organizing and managing application details in a centralized and streamlined manner, which is the core emphasis of our product.

Purpose of New System

The purpose of the proposed application is to provide users with a streamlined solution for managing their job applications and associated details. Users can save information about each application, including important updates such as scheduled interviews and status changes like "In Progress," "Offer Accepted," "Offer Extended," "Offer Declined," or "Rejected." Additionally, the system offers analytics to provide insights into the application process, such as the total number of applications submitted and a breakdown of statuses by category. Other features include the ability to search for specific applications, filter and organize entries, and schedule events using a built-in calendar. This comprehensive functionality ensures users can efficiently track and manage their job search in an organized and intuitive way.

User Stories

The following section provides the detailed user stories that were implemented in this iteration of the **Path Tracker** project. These user stories served as the basis for the implementation of the project's features. This section also shows the user stories that are to be considered for future development.

Implemented User Stories

- As a user, I want to be able to sign up to create an account and then log in securely.
- As a user, I want to save, track, and review details of each job application, including the date applied, company name, position, and personal notes.
- As a user, I want to view a list of all my job applications displayed on my dashboard.
- As a user, I want to manually update and track the status of my job applications to reflect the current stage of the process, with statuses such as "In Progress," "Interview Scheduled," "Rejected," and more.
- As a user, I want to receive important notifications about updates to my job applications.
- As a user, I want to be able to filter my job applications based on specific attributes
- As a user, I want to add and edit details about my job applications stored in my dashboard.
- As a user, I want to manually add job applications to my dashboard and view their important details.
- As a developer I want to implement a SQL database for persistent storage for the program.
- As a developer, I want to implement the back end of the application using Java and the Spring Boot framework, following the MVC design pattern, so that I can efficiently organize the application logic.
- As a user, I want to navigate the website securely, starting at a landing page that includes key details about the app.
- As a user, I want to interact with various app features and move seamlessly between tabs and functionalities.
- As a user, I want to log out securely from my profile.
- As a developer I want to be able to connect the front-end and the back-end of the application successfully through API endpoints.
- As a user, I would like to use a Google Chrome extension to save job postings automatically to my dashboard.
- As a user, I want to have analytics about the number of jobs I have applied to and a breakdown by categories.
- As a user, I want to use a calendar to create events and track them, such as scheduled interviews or phone calls.

Pending User Stories

- As a user, I would like the app to integrate with my Google or Microsoft Calendar to save important events.
- As a user, I want to group my job applications by categories such as "Company Name," "Date Applied," "Position," and others.
- As a user, I want to save job postings from any website using the Chrome extension.
- As a user, I want to save my resume in my profile for easy access.
- As a user, I would like the Chrome extension to autofill job applications.
- As a developer I want to migrate our current architecture to a microservice architecture that uses an API gateway and Eureka server.

Project Plan

This section describes the planning that went into the realization of this project. This project incorporated the agile development techniques and as such required the sprints to be planned. These sprint plannings are detailed in the section. This section also describes the components, both software and hardware, chosen for this project.

Hardware and Software Requirements

Hardware

- Personal laptops and computers with access to coding tools and the internet

Software: Front-End

- React.js
- HTML
- CSS
- JavaScript
- Angular
- Node
- Visual Studio Code

Software: Back-End

- Java
- SpringBoot
- Python
- Flask

Software: Database

- MySQL Server
- MySQL Workbench
- Azure SQL
- Azure Data Studio

Sprint Plan

Sprint 1

User Story #1: As a user, I want to be able to sign up to create an account and then log in securely.

- Acceptance Criteria:
 - Users can sign up with email and password.
 - Users can log in and out securely.
 - Passwords are hashed and stored securely to the database.

User Story #2: As a user, I want to save, track, and review details of each job application, including the date applied, company name, position, and personal notes.

- Acceptance Criteria:
 - Users can input job details (company name, position, application date, status).
 - Application entries are saved to the database.
 - Users receive confirmation when an application is successfully added.
 - Users can automatically save the application from the different platforms

User Story #3: As a user, I want to view a list of all my job applications displayed on my dashboard.

- Acceptance Criteria:
 - The dashboard displays a list of job applications with key details (company name, position, status).
 - The list is sortable by date, company, and status.
 - Users can click on an application to view more details.

User Story #4: As a user, I want to manually update and track the status of my job applications to reflect the current stage of the process, with statuses such as "In Progress," "Interview Scheduled," "Rejected," and more.

- Acceptance Criteria:
 - Users can select and update the status of an application (e.g., "Interview Scheduled," "Rejected," "Offer Received").
 - Status changes are saved and reflected on the dashboard.

User Story #5: As a user, I want to receive important notifications about updates to my job applications.

- Acceptance Criteria:
 - Users receive notifications for changes in application status.
 - Users can opt in or out of notifications.
 - Notifications are sent on app.

Sprint 2

User Story #1: As a user, I want to be able to sign up to create an account and then log in securely.

- Acceptance Criteria:
 - Users can sign up with email and password.
 - Users can log in and out securely.
 - Passwords are hashed and stored securely to the database.

User Story #2: As a user, I want to save, track, and review details of each job application, including the date applied, company name, position, and personal notes.

- Acceptance Criteria:
 - Users can input job details (company name, position, application date, status).
 - Application entries are saved to the database.
 - Users receive confirmation when an application is successfully added.
 - Users can automatically save the application from the different platforms

User Story #3: As a user, I want to view a list of all my job applications displayed on my dashboard.

- Acceptance Criteria:
 - The dashboard displays a list of job applications with key details (company name, position, status).
 - The list is sortable by date, company, and status.
 - Users can click on an application to view more details.

User Story #4: As a user, I want to manually update and track the status of my job applications to reflect the current stage of the process, with statuses such as "In Progress," "Interview Scheduled," "Rejected," and more.

- Acceptance Criteria:
 - Users can select and update the status of an application (e.g., "Interview Scheduled," "Rejected," "Offer Received").
 - Status changes are saved and reflected on the dashboard.

User Story #5: As a user, I want to receive important notifications about updates to my job applications.

- Acceptance Criteria:
 - Users receive notifications for changes in application status.
 - Users can opt in or out of notifications.
 - Notifications are sent on app.

Sprint 3

User Story #1: As a user, I want to be able to filter my job applications based on specific attributes

- Acceptance Criteria:
 - I am able to see all of my job application in my dashboard
 - I am able to see a subset of my job applications in my dashboard based on the filter applied
 - There is a drop-down menu with options for the filter

User Story #2: As a user, I want to add and edit details about my job applications stored in my dashboard.

- Acceptance Criteria:
 - The job application is still showing in the dashboard after refreshing the page.
 - Each application shows the date applied, name of the position, and status.
 - I am able to change the details about the job applications after saving them

User Story #3: As a user, I want to manually add job applications to my dashboard and view their important details.

- Acceptance Criteria:
 - Users can add the name of a company, job title, date applied, and status to a job manually from the dashboard
 - User can add details about the job applications to their dashboard

User Story #4: As a developer I want to implement a SQL database for persistent storage for the program.

- Acceptance Criteria:
 - The data from the front end is saved in tabular form in MySQL database
 - Each table in the database represents an object in the program
 - The attributes of each object are relevant to the context of the app

User Story #5: As a developer, I want to implement the back end of the application using Java and the Spring Boot framework, following the MVC design pattern, so that I can efficiently organize the application logic.

- Acceptance Criteria:
 - The architecture for the back end is build using Java and SpringBoot.
 - The back end runs and handles data with no errors.

Sprint 4

User Story #1: As a user, I want to be able to filter my job applications based on specific attributes

- Acceptance Criteria:
 - I am able to see all of my job application in my dashboard
 - I am able to see a subset of my job applications in my dashboard based on the filter applied
 - There is a drop-down menu with options for the filter

User Story #2: As a user, I want to add and edit details about my job applications stored in my dashboard.

- Acceptance Criteria:
 - The job application is still showing in the dashboard after refreshing the page.
 - Each application shows the date applied, name of the position, and status.
 - I am able to change the details about the job applications after saving them

User Story #3: As a user, I want to manually add job applications to my dashboard and view their important details.

- Acceptance Criteria:
 - Users can add the name of a company, job title, date applied, and status to a job manually from the dashboard
 - User can add details about the job applications to their dashboard

User Story #4: As a developer I want to implement a SQL database for persistent storage for the program.

- Acceptance Criteria:
 - The data from the front end is saved in tabular form in MySQL database
 - Each table in the database represents an object in the program
 - The attributes of each object are relevant to the context of the app

User Story #5: As a developer, I want to implement the back end of the application using Java and the Spring Boot framework, following the MVC design pattern, so that I can efficiently organize the application logic.

- Acceptance Criteria:
 - The architecture for the back end is build using Java and SpringBoot.
 - The back end runs and handles data with no errors.

Sprint 5

User Story #1: As a user, I want to navigate the website securely, starting at a landing page that includes key details about the app.

- Acceptance Criteria:
 - The first page that the users see is the landing page
 - I can navigate to sign up and log in from the landing page
 - I can navigate between different tabs inside the app after logging in

User Story #2: As a user, I want to interact with various app features and move seamlessly between tabs and functionalities.

- Acceptance Criteria:
 - Once logged in, all of the features from the app work properly
 - I can move from one feature or functionality to other easily
 - I am able to go back to any feature or tab however many times I want

User Story #3: As a user, I want to log out securely from my profile.

- Acceptance Criteria:
 - After logging in, I can securely log out of my profile whenever I want
 - After logging out there is no way to access my profile without logging in again

User Story #4: As a developer I want to be able to connect the front-end and the back-end of the application successfully through API endpoints.

- Acceptance Criteria:
 - Data and requests from the front end are successfully communicated to the back end
 - The back end successfully communicates with the database and the data is stored

User Story #5: As a developer I want to migrate our current architecture to a microservice architecture that uses an API gateway and Eureka server.

- Acceptance Criteria:

- The architecture from the app is made up of an API gateway, Eureka server and loosely coupled microservices
- All of the components successfully communicate with each other.
- The microservices successfully communicate with the database

Sprint 6

User Story #1: As a user, I want to navigate the website securely, starting at a landing page that includes key details about the app.

- Acceptance Criteria:
 - The first page that the users see is the landing page
 - I can navigate to sign up and log in from the landing page
 - I can navigate between different tabs inside the app after logging in

User Story #2: As a user, I want to interact with various app features and move seamlessly between tabs and functionalities.

- Acceptance Criteria:
 - Once logged in, all of the features from the app work properly
 - I can move from one feature or functionality to other easily
 - I am able to go back to any feature or tab however many times I want

User Story #3: As a user, I want to log out securely from my profile.

- Acceptance Criteria:
 - After logging in, I can securely log out of my profile whenever I want
 - After logging out there is no way to access my profile without logging in again

User Story #4: As a developer I want to be able to connect the front-end and the back-end of the application successfully through API endpoints.

- Acceptance Criteria:
 - Data and requests from the front end are successfully communicated to the back end
 - The back end successfully communicates with the database and the data is stored

User Story #5: As a developer I want to migrate our current architecture to a microservice architecture that uses an API gateway and Eureka server.

- Acceptance Criteria:
 - The architecture from the app is made up of an API gateway, Eureka server and loosely coupled microservices
 - All of the components successfully communicate with each other.
 - The microservices successfully communicate with the database

Sprint 7

User Story #1: As a user, I would like to use a Google Chrome extension to save job postings automatically to my dashboard.

- Acceptance Criteria:
 - I can save job applications to my dashboard using the chrome extensions
 - The details from the job posting are correctly saved in the job application
 - I can edit the saved details from my dashboard later.

User Story #2: As a user, I want to have analytics about the number of jobs I have applied to and a breakdown by categories.

- Acceptance Criteria:
 - In my dashboard I can see the total number of jobs I applied
 - There is a visual chart that shows me my job applications broken down by their status
 - When I update a job application, the analytics update as well

User Story #3: As a user, I want to use a calendar to create events and track them, such as scheduled interviews or phone calls.

- Acceptance Criteria:
 - The app has a built-in calendar
 - I can add and remove events to the calendar in the app
 - The events display correctly in the calendar

User Story #4: As a developer I want to be able to connect the front-end and the back-end of the application successfully through API endpoints.

- Acceptance Criteria:
 - Data and requests from the front end are successfully communicated to the back end
 - The back end successfully communicates with the database and the data is stored

User Story #5: As a developer I want to migrate our current architecture to a microservice architecture that uses an API gateway and Eureka server.

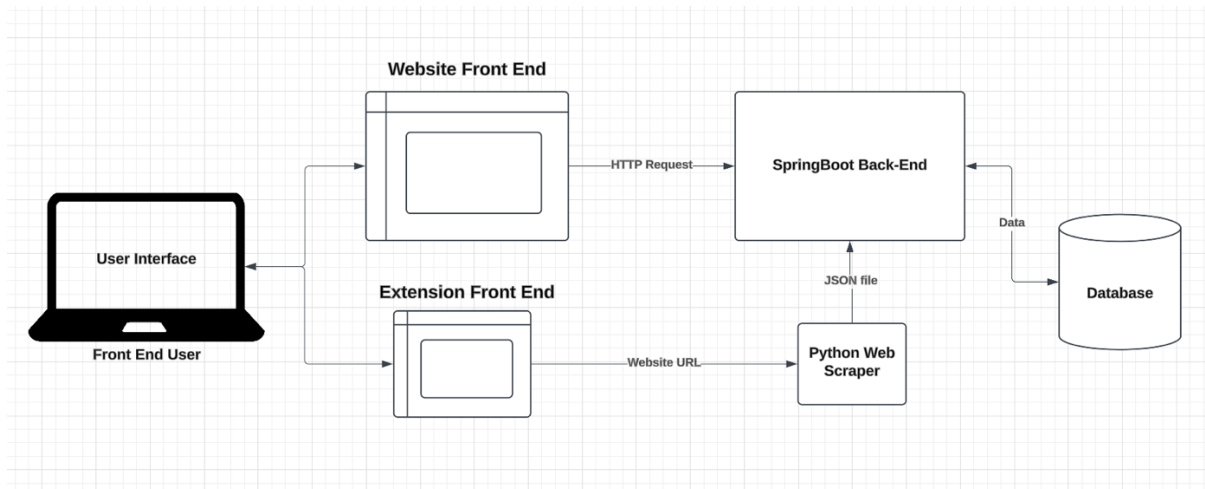
- Acceptance Criteria:
 - The architecture from the app is made up of an API gateway, Eureka server and loosely coupled microservices
 - All of the components successfully communicate with each other.
 - The microservices successfully communicate with the database.

System Design

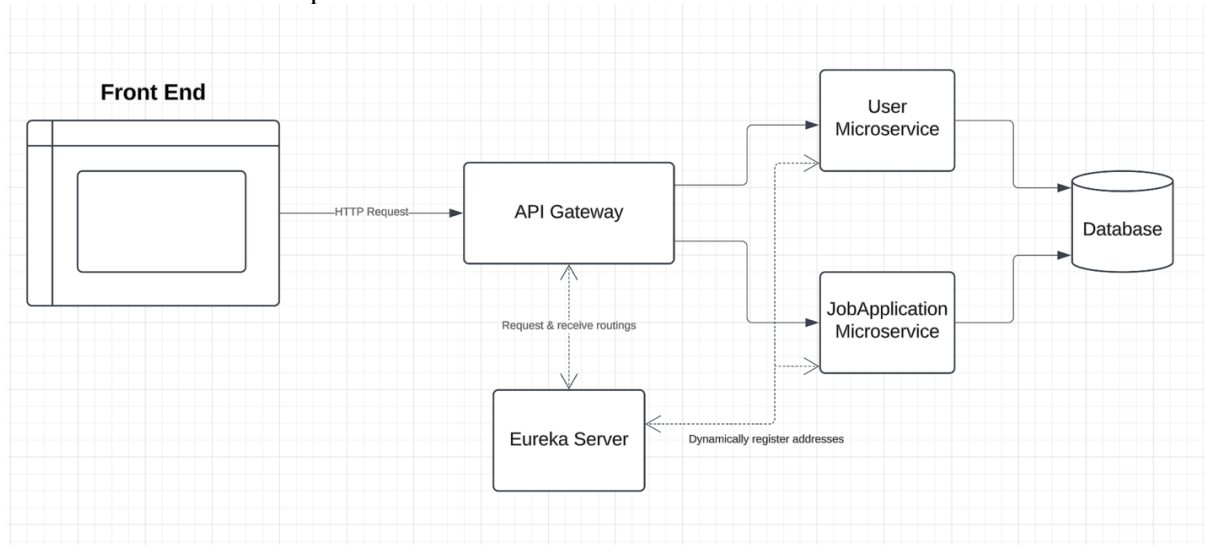
This section contains information on the design decisions that went into this project. The architecture patterns are outlined and explained. The entire system is shown in a package diagram and the subsystems are explained. Finally, the design patterns used in the project are discussed.

Architectural Patterns

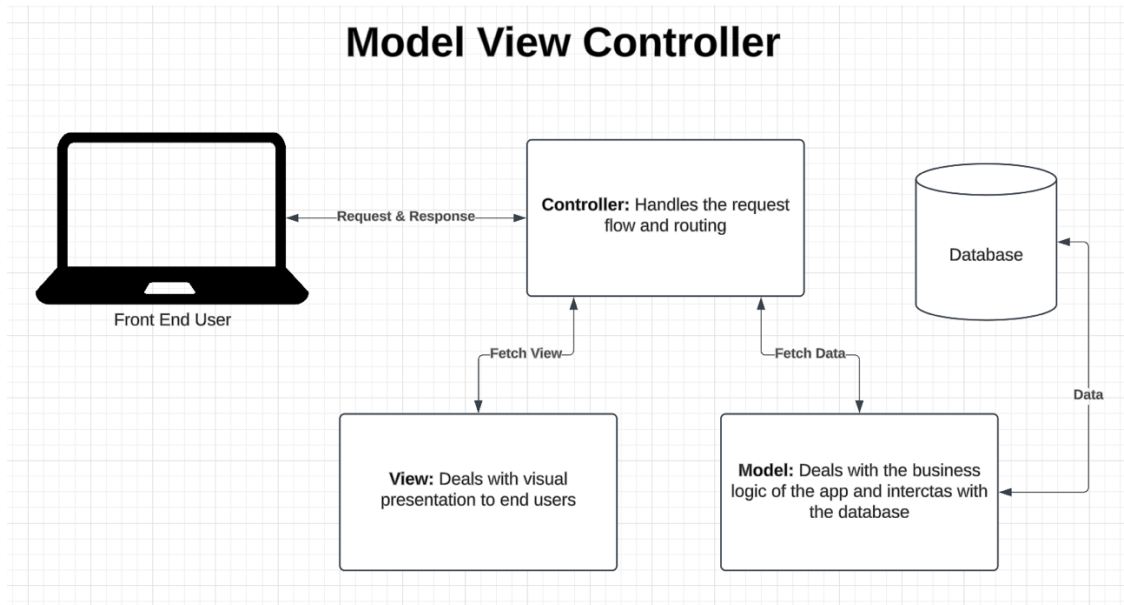
Current Structure:



Future Microservice Implementation:



System and Subsystem Decomposition



Design Patterns

The architecture of this application follows the Model-View-Controller (MVC) design pattern, which organizes the application into three main components: the controller, model, and view. The controller manages the routing of the application, the model handles business logic and database interactions, and the view is responsible for rendering the user interface that users interact with.

In addition to the MVC framework, the back end implements a monolithic architecture using Java SpringBoot. This back end is able to manage all of the routing of the application as well as the necessary business logic and the communication with the database. The MVC framework is implemented in the back end by creating controller classes and model classes for each object of the system.

For future iterations of the project a microservices architecture wants to be implemented. The microservices would be divided into two main services: a user microservice and a job application microservice. These microservices communicate through an API gateway, which acts as the central entry point for the back end, and they are registered with a Eureka server for service discovery and registry. Both microservices also establish connections to the database, ensuring a modular and scalable architecture.

System Validation

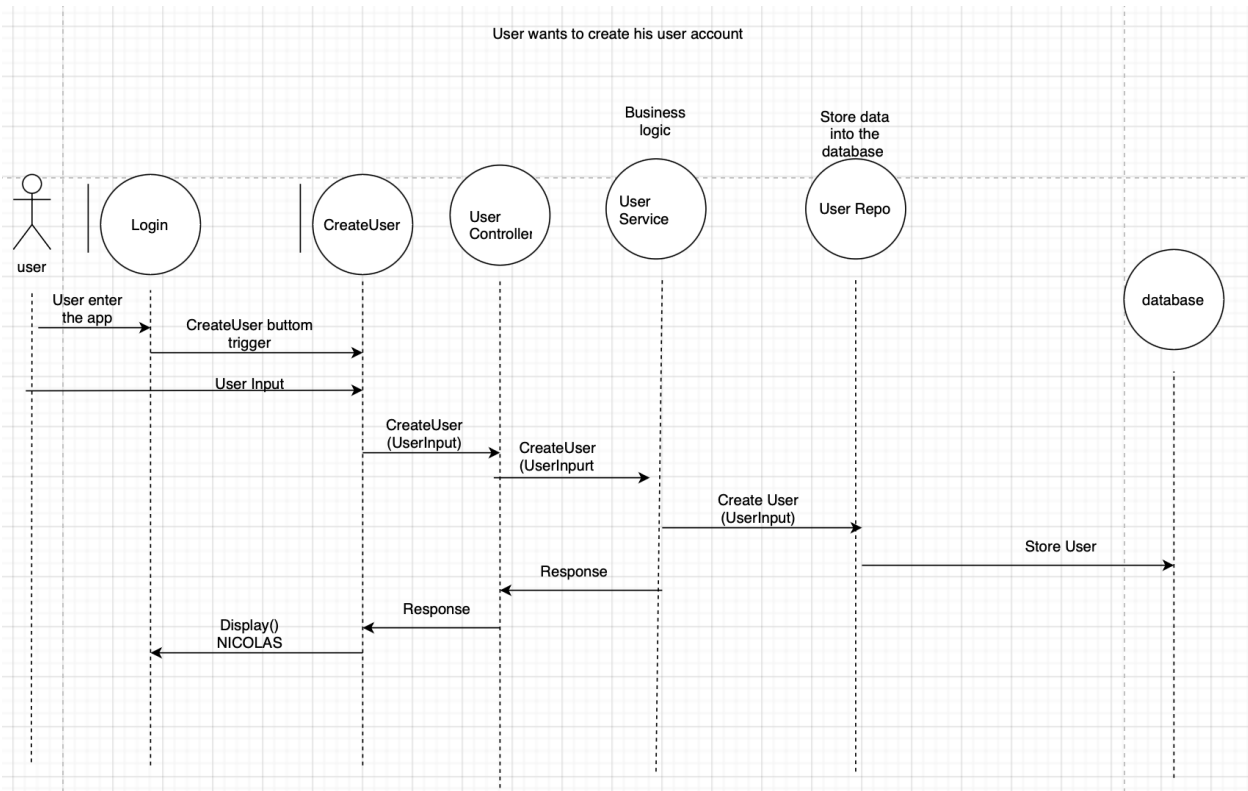
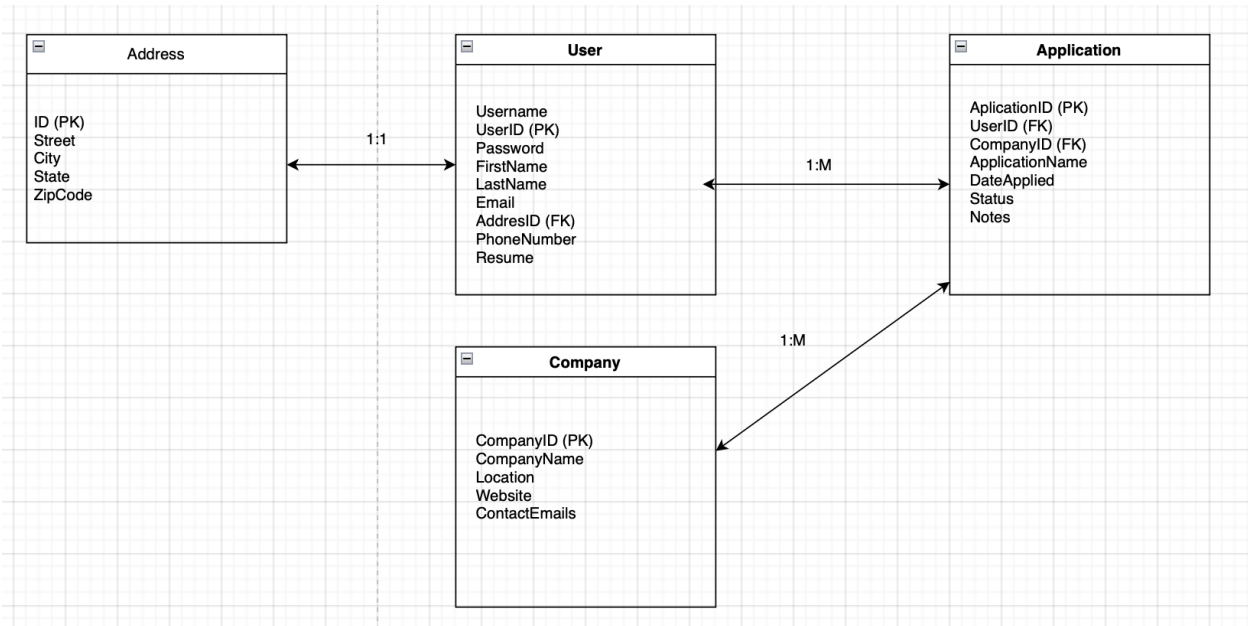
The system was validated through comprehensive testing of its features to ensure proper functionality and user experience. Each front-end feature was thoroughly tested, including

signing up and logging in, creating and saving job applications, as well as editing and deleting job applications and their associated details. The accuracy of the analytics on the dashboard, including the correct count and breakdown by category, was verified, along with the seamless navigation between tabs and the ability to create events in the calendar.

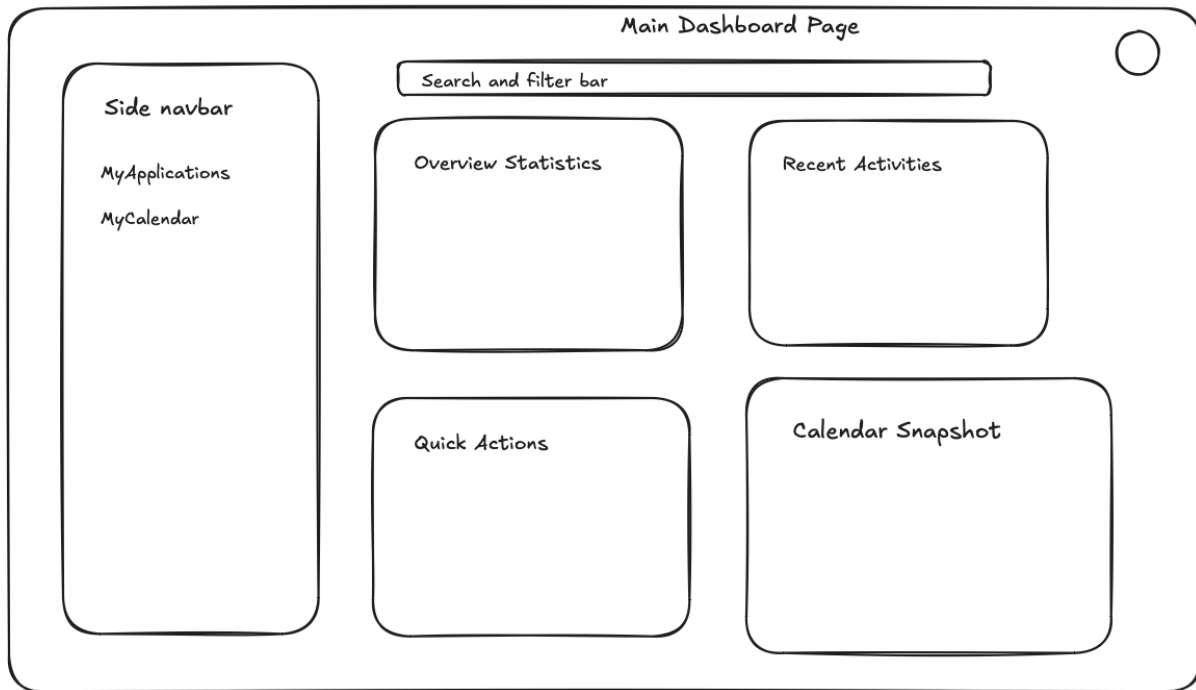
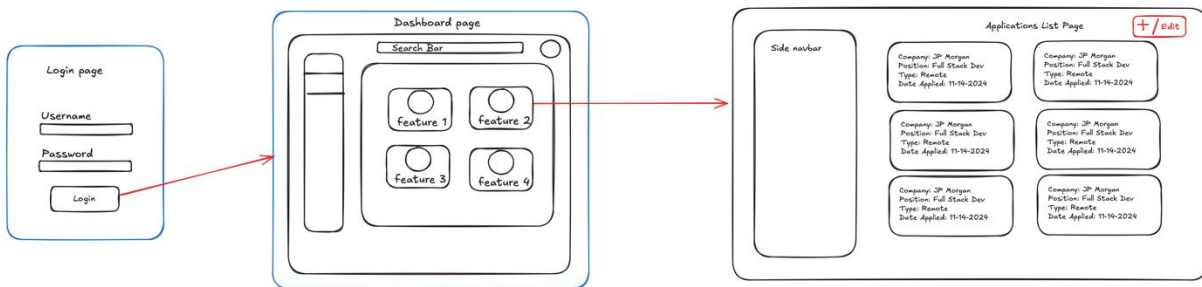
Furthermore, the creation of objects in the database was validated by inspecting the contents of the entries directly during development. The connection of APIs and endpoints was tested using tools such as Postman to ensure reliable communication between the front-end and back-end components. These extensive validation efforts confirm the system's functionality and readiness for deployment.

Appendix

Appendix A – UML Diagrams



Appendix B – User Interface Design



Side navbar

Applications List Page

+ / Edit

Company: JP Morgan
Position: Full Stack Dev
Type: Remote
Date Applied: 11-14-2024

Company: JP Morgan
Position: Full Stack Dev
Type: Remote
Date Applied: 11-14-2024

Company: JP Morgan
Position: Full Stack Dev
Type: Remote
Date Applied: 11-14-2024

Company: JP Morgan
Position: Full Stack Dev
Type: Remote
Date Applied: 11-14-2024

Company: JP Morgan
Position: Full Stack Dev
Type: Remote
Date Applied: 11-14-2024

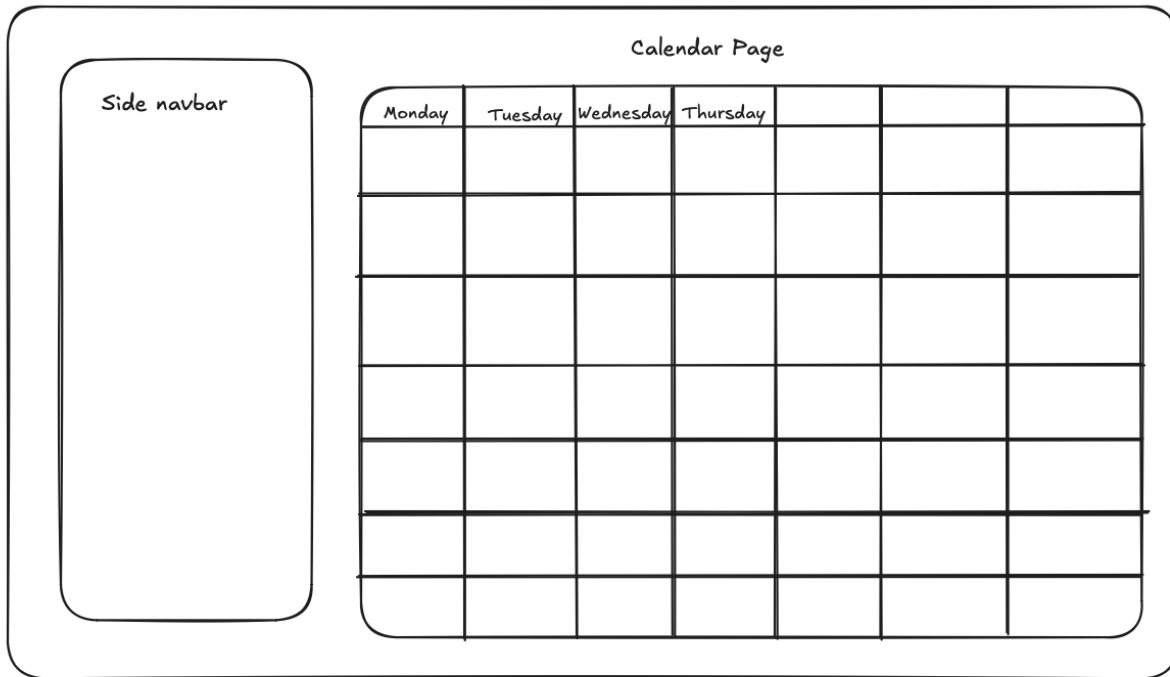
Company: JP Morgan
Position: Full Stack Dev
Type: Remote
Date Applied: 11-14-2024

Side navbar

Job Applications Details Page



Company: JP Morgan Chase
Position: Full Stack Engineer
Type: Remote
Date Applied: 11-14-2024
Notes:
Status: Interview Scheduled



Appendix C – Sprint Review Reports

Sprint 1: Review Meeting Minutes

- Attendees: Nicolas Astros, Martin Archila, Marco Chaparro, Cairo Crawford, Alejandro Morales
- Start time: 3:30PM
- End time: 4:30PM

After a show-and-tell presentation, we consolidated our findings as follows.

- Features/functions that were tested thoroughly and worked as expected without any errors/issues:
 - We haven't started testing the features of our product
- Features/functions that were tested thoroughly and did NOT work as expected. Here is the list of issues/errors observed.
 - Since we haven't started the testing of our features, none of the features are listed.
- These are features/functions that were tested thoroughly and would benefit from some improvement/enhancement. Here is the list of suggested enhancements/improvements.

- Since we haven't started the testing of our features, none of the features are listed.

Sprint 2: Review Meeting Minutes

- Attendees: Nicolas Astros, Martin Archila, Marco Chaparro, Cairo Crawford, Alejandro Morales
- Start time: 6:30PM
- End time: 7:30PM

After a show-and-tell presentation, we consolidated our findings as follows:

- Features/functions that were tested thoroughly and worked as expected without any errors/issues:
 - The navbar, dashboard, settings, login and signup inputs, and backend tasks,
- Features/functions that were tested thoroughly and did NOT work as expected. Here is the list of issues/errors observed:
 - None
- These are features/functions that were tested thoroughly and would benefit from some improvement/enhancement. Here is the list of suggested enhancements/improvements:
 - The login and signup page since they still need to be styled and connected to the backend.

Sprint 3: Review Meeting Minutes

- Attendees: Nicolas Astros, Martin Archila, Marco Chaparro, Cairo Crawford, Alejandro Morales
- Start time: 7:30PM
- End time: 8:30PM

After a show-and-tell presentation, we consolidated our findings as follows:

- Features/functions that were tested thoroughly and worked as expected without any errors/issues:
 - The navbar, simple dashboard page, and back-end endpoints for user login and entering a job application through the dashboard.
- Features/functions that were tested thoroughly and did NOT work as expected. Here is the list of issues/errors observed:

- The login/signup connection with the database for user authentication
- These are features/functions that were tested thoroughly and would benefit from some improvement/enhancement. Here is the list of suggested enhancements/improvements:
 - Fixing the issues occurring with the backend connection.

Sprint 4: Review Meeting Minutes

- Attendees: Nicolas Astros, Martin Archila, Marco Chaparro, Cairo Crawford, Alejandro Morales
- Start time: 7:30PM
- End time: 8:30PM

After a show-and-tell presentation, we consolidated our findings as follows:

- Features/functions that were tested thoroughly and worked as expected without any errors/issues:
 - The dashboard page, and back-end endpoints for user login and entering a job application through the dashboard.
 - Back-end security enhancement through JWT token
 - Front end, jobs analytics page
- Features/functions that were tested thoroughly and did NOT work as expected. Here is the list of issues/errors observed:
 - Hosting SQL database server for development
- These are features/functions that were tested thoroughly and would benefit from some improvement/enhancement. Here is the list of suggested enhancements/improvements:
 - Fixing the issues occurring with the backend connection.

Sprint 5: Review Meeting Minutes

- Attendees: Nicolas Astros, Martin Archila, Marco Chaparro, Cairo Crawford, Alejandro Morales
- Start time: 7:30PM
- End time: 8:30PM

After a show-and-tell presentation, we consolidated our findings as follows:

- Features/functions that were tested thoroughly and worked as expected without any errors/issues:

- Secure routing and authentication and authorization with every request to the backend
- Database deployment in Azure
- Features/functions that were tested thoroughly and did NOT work as expected. Here is the list of issues/errors observed:
 - Web scrapper tool to retrieve job application information from LinkedIn, Handshake, etc.
- These are features/functions that were tested thoroughly and would benefit from some improvement/enhancement. Here is the list of suggested enhancements/improvements:
 - Landing page design
 - Dashboard information display

Sprint 6: Review Meeting Minutes

- Attendees: Nicolas Astros, Martin Archila, Marco Chaparro, Cairo Crawford, Alejandro Morales
- Start time: 7:30PM
- End time: 8:30PM

After a show-and-tell presentation, we consolidated our findings as follows:

- Features/functions that were tested thoroughly and worked as expected without any errors/issues:
 - Log in, landing, dashboard, routing, sign out, create user and many other features of the front end.
- Features/functions that were tested thoroughly and did NOT work as expected. Here is the list of issues/errors observed:
 - Connection of chrome extension and microservice architecture of the back end.
- These are features/functions that were tested thoroughly and would benefit from some improvement/enhancement. Here is the list of suggested enhancements/improvements:
 - Connection between extension and app, cleaning up the front end and microservice architecture.

Sprint 7: Review Meeting Minutes

- Attendees: Nicolas Astros, Martin Archila, Marco Chaparro, Alejandro Morales

- Start time: 7:30PM
- End time: 8:30PM

After a show-and-tell presentation, we consolidated our findings as follows:

- Features/functions that were tested thoroughly and worked as expected without any errors/issues:
 - Front end landing page, log in, sign up, routing, calendar, add job applications, google chrome extension
 - Back-end connectivity and architecture, database connectivity
- Features/functions that were tested thoroughly and did NOT work as expected. Here is the list of issues/errors observed:
 - Google chrome extension does not work for all job postings
- These are features/functions that were tested thoroughly and would benefit from some improvement/enhancement. Here is the list of suggested enhancements/improvements:
 - Cloud SQL server