# JobQuest Navigator – Week 5 Sprint Document

Maria Soto, Shruti Amit Vasanwala, Zhihuai Wang, Ishan Aakash Patel

Team 9

The Zombies of CAA

Seneca Polytechnic

Course Code: CAA900

David Chan

## Sprint Goals

Sprint: **Sprint 2**

**Sprint Dates:**31 May 2025 to 07 June 2025

**Team Participation:**All team members are actively working on Sprint 2. Each member is assigned tasks across different Epics, ensuring comprehensive coverage of project objectives.

**Tracking and Assignments:**Task assignments and progress can be monitored via the Jira board:[**https://myseneca-team-pi6s3gm8.atlassian.net/jira/software/projects/SM/list?sortBy=key&direction=ASC**](https://myseneca-team-pi6s3gm8.atlassian.net/jira/software/projects/SM/list?sortBy=key&direction=ASC)

**Documentation and Review:**

* Jira: Used for task management, sprint tracking, and assignment visibility.
* GitHub: Weekly progress and key decisions are documented in the project’s repository for transparency and accountability.

## Tasks

List of the tasks Sprint 2:

|  |  |
| --- | --- |
| **Epic** | **Task** |
| 1 | IU Design |
| 1 | User Story 1.3 (A): Sync job listings with map pins based on geolocation. |
| 3 | User Story 3.1 (A): Implement AI suggestions for resume alterations. |

## Key Features Being Developed

* **IU Desing: Figma UI link:** [**JobQuestUI**](https://www.figma.com/design/3VvLCvPZR7dZGnxCRk26GK/JobQuest?node-id=0-1&m=dev)

As part of this sprint, the frontend development progressed significantly. Based on our finalized Figma UI, the following pages were implemented using React.js, HTML, CSS, and JavaScript:

* **Login & Signup Page** – Fully styled forms with validation and navigation logic.
* **Dashboard Page** – Candidate dashboard layout with placeholder sections for saved jobs, profile, and recent activity.
* **Job Search Page** – Dynamic layout to display job cards fetched from mock data; includes filters and search functionality.
* **Map Interface Page** – Integrated Google Maps with job pins. Pins are clickable and show job title, company name, and "Apply" button in a popup.
* **Navigation Header/Footer** – Common layout components applied across pages for consistency.

All pages follow responsive design principles and maintain a clean, minimal UI in alignment with the JobQuest theme.

Working Interface: [Video](https://drive.google.com/file/d/17fj3c1MrCXwAiNEJVIKHqM4nyo3jUsZl/view?usp=sharing)

## Workflow and Tools Used

**1. Project Setup**

* We started by creating a new React project using Create React App, which provides a solid foundation for building modern web applications.

**2. Version Control**

* We used Git for version control to track changes and collaborate efficiently.
* The project is hosted on GitHub, allowing for easy sharing, branching, and pull requests.

**3. Development Tools**

* **React:** The main JavaScript library used to build the user interface. React allows us to create reusable components and manage the UI efficiently.
* **HTML & CSS:** Used for structuring and styling the web pages. We wrote custom CSS for layout and design, ensuring the UI matches the Figma design.
* **JavaScript (ES6+):** Used for logic, interactivity, and connecting components.
* **React Router:** Used for navigation between different pages (like Login, Signup, Dashboard, etc.) without reloading the page.
* **Context API:** Used for managing global state, such as user authentication and job data, across the app.

**4. External Integrations**

* **Google Maps API:** Integrated to display job locations on a map, allowing users to visualize job opportunities geographically.
* **Adzuna API:** Used to fetch real-time job listings, making the job search experience dynamic and up-to-date.

**5. Workflow Steps**

* We broke down the UI into reusable components (e.g., NavBar, Job Listings, Application Form).
* Each page (Login, Signup, Dashboard, etc.) was developed as a separate component for clarity and maintainability.
* We used live reloading (via yarn start or npm start) to see changes instantly during development.
* All changes were committed to Git and pushed to GitHub, ensuring a backup and version history.
* We used branches (like dev and shruti) to manage features and updates without disrupting the main codebase.

**6. Testing and Debugging**

* We used the browser’s developer tools and React’s built-in error messages to debug and test the UI.
* The app was tested for responsiveness to ensure it works well on different devices.

**7. Deployment (if applicable)**

* The project can be easily deployed to platforms like Vercel, Netlify, or GitHub Pages for public access.