# JobQuest Navigator – Week 6 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 3**

**Sprint Dates:**08 June 2025 to 15 June 2025

**Team Participation:**All team members are actively working on Sprint 3. 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 3:

|  |  |
| --- | --- |
| **Epic** | **Task** |
| 1 | IU Design |
| 1 | User Story 1.4: Implement filtering and prioritization of nearby jobs. |
| 1 | User Story 1.2 (A): Design and implement a visual map interface. |
| 2 | User Story 2.2: Integrate Jobscan’s resume library |
| 3 | User Story 3.1 (B): Implement AI suggestions for resume alterations. |
| 4 | User Story 4.1: Integrate Careerflow’s skill gap analysis. |

## Key Features Being Developed

* **Epic 1:**

**Frontend visuals link:** https://drive.google.com/file/d/1op8dki1Ods0qWlG77uCcAThdGOTCFn6A/view?usp=sharing

1. Completed Features

a) Page Development

- 404 Error Page

- Navigation System

- Resume Builder

- Interview Preparation

- Company Profile

- Job Details

- Application History

- Settings Page

b) UI/UX Improvements

- Consistent design system implementation

- Responsive layouts for all pages

- Enhanced navigation experience

- Improved user flow

2. Technical Achievements

- Implemented React Router v6 for routing

- Created reusable components

- Established consistent styling patterns

- Enhanced code maintainability

- Improved performance optimization

3. Design System Implementation

- Color scheme standardization

- Typography hierarchy

- Component spacing rules

- Responsive breakpoints

- Interactive states

4. Challenges & Solutions

- Navigation organization: Implemented logical grouping

- Responsive design: Created flexible layouts

- Code maintainability: Established component structure

- Performance: Optimized image loading

5. Metrics

- Pages Completed: 8

- Components Created: 12

- Code Coverage: TBD

- Performance Score: TBD

* **Epic 2**

Resume Versioning & Suggestion

- Analyzed and confirmed Django project structure.

- Supplemented missing files: README.md, frontend/index.html, tests/test\_api.py.

- Created a basic frontend skeleton for resume management.

- Created a Django test skeleton for API endpoints.

- Initialized Task Master for task management.

- Fixed Django settings (INSTALLED\_APPS) for proper test execution.

- Verified and documented the test environment.

* **Epic 3**

Epic 3, designated as 'Resume Suggestion Feedback,' is designed to provide AI-driven suggestions for resume improvement and establish feedback loops. This epic directly builds upon the functionalities introduced in Epic 2. The primary documentation for Epic 3 includes `EPIC3\_PRD.md`, which outlines the product requirements, and `Epic3\_Certification\_Roadmap.md`, providing additional context on its development and certification process. Furthermore, `swagger\_epic3\_certification.yaml` serves as the OpenAPI/Swagger specification for Epic 3, detailing its API endpoints and data models. The core objective of Epic 3 is to enhance the user's resume by offering intelligent suggestions, likely based on job descriptions (potentially leveraging data from Epic 1) and the user's existing resume versions (from Epic 2).

1. Complete Features:

- Epic 3 provides AI-driven suggestions for improving resumes and includes mechanisms for user feedback.

- It enhances user resumes with intelligent suggestions and continuously improves AI models based on feedback.

- The system consumes resume data from Epic 2 and can use job data from Epic 1 to generate tailored suggestions.

- Frontend components are designed to display suggestions, collect user feedback, and integrate with resume editing workflows.

2. Technical Achievements:

- Epic 3 is built as a set of microservices, including feedback, model training, and resume suggestion services.

- It uses OpenAPI/Swagger for API specifications and has detailed product requirements documentation.

- Architectural diagrams, such as Data Flow Diagrams, are proposed to visualize system interactions.

- A comprehensive testing strategy is in place, covering unit, integration, AI/ML model evaluation, end-to-end, and performance testing.

- The project emphasizes an API-first approach for seamless frontend integration, aligning data models and handling errors effectively.

3. Challenges & Solutions:

- Challenge: Ensuring alignment and seamless integration with Epic 1 and Epic 2.

Solution: Epic 3 directly builds on Epic 2's functionalities and can leverage Epic 1's data, with shared data models and API interoperability being key.

- Challenge: Guaranteeing the accuracy and reliability of AI-driven suggestions.

Solution: A robust testing strategy includes human-in-the-loop validation and bias detection for AI/ML models.

- Challenge: Integrating with the Figma-designed frontend.

Solution: An API-first approach, clear UI/UX flow mapping, data model alignment, and strong collaboration between design and development teams are going to implement.

4. Metrics:

- The accuracy and effectiveness of AI/ML models are evaluated using metrics like precision, recall, F1-score, and RMSE during training.

- Performance is assessed by monitoring responsiveness and scalability under various loads.

- Key performance indicators include response times, resource utilization (CPU, memory), and error rates.

* **Epic 4**

Automation Tools Used & How They Were Applied

* spaCy (Python NLP): Used for AI-powered skill extraction from resume text. The Python script (`extract\_skills.py`) processes user input and returns a filtered, accurate list of skills.
* Node.js/TypeScript Backend: Orchestrates the workflow, calls the Python script, and maps extracted skills to certifications using a local JSON map.

Deliverable Files:

* README.md
* USER\_GUIDE.md
* MICROSERVICES\_USED.md
* swagger\_epic4\_skillgap.yaml

Progress:

* spaCy-based skill extraction and certification mapping is complete.
* All documentation and microservice overviews are up to date.
* Codebase is clean and ready for final enhancements (hybrid API integration in progress).
* **Epic 6**

Company Research & Interview Prep

- Created a new Django project and app (company\_research, api).

- Generated requirements.txt for Django REST API development.

- Created README.md with project overview, setup, and structure.

- Created a basic frontend skeleton (frontend/index.html).

- Created a Django test skeleton (tests/test\_api.py).

- Initialized Task Master for task management.

- Ensured all project scaffolding aligns with best practices.