JobLaunch Project – Academic Test Plan

1. Introduction

This document presents the Test Plan for the JobLaunch project, a web-based job hunting and application platform developed as part of an academic software engineering project. The purpose of this test plan is to define a systematic approach to verifying and validating the system against its specified functional and non-functional requirements, as outlined in the Software Requirements Specification (SRS) and design documentation. The plan aims to ensure that the system operates correctly, reliably, and securely within the scope of the project.

2. Testing Objectives

The primary objectives of testing for the JobLaunch system are as follows:

- To confirm that all core functional features, including user authentication, job posting, job searching, and application management, operate according to the defined requirements.
- To evaluate key non-functional aspects of the system, such as performance, security, and usability.
- To identify, document, and support the resolution of defects discovered during testing.
- To establish confidence in the stability and readiness of the system prototype for academic evaluation and final demonstration.

3. Scope of Testing

In Scope:

- Functional testing of all system features specified in the SRS.
- Non-functional testing focusing on performance, basic security measures, and user usability.
- End-to-end workflow testing covering the complete job application lifecycle, from job seeker application submission to employer status updates and applicant review.

Out of Scope:

- Advanced scalability and optimization mechanisms, such as caching and distributed search engines.
- Extended third-party services beyond the minimum viable product, including automated resume parsing and real-time notification systems.

4. Test Items

The components subject to testing include:

- The frontend user interface, including registration, login, job listings, and application forms.
- Backend application programming interfaces (APIs) responsible for authentication, job management, and application processing.
- The database layer, specifically tables related to users, job postings, and applications.
- The deployment environment used for staging and demonstration purposes.

5. Testing Methodology and Strategy

A multi-level testing approach is adopted to ensure comprehensive coverage:

- **Unit Testing:** Individual functions and modules are tested by developers to verify correct internal logic.
- **Integration Testing:** Interactions between APIs, backend services, and the database are tested to ensure data consistency and proper communication.
- **System Testing:** Complete system workflows are tested to validate end-to-end functionality.
- **User Acceptance Testing (UAT):** Representative user scenarios are simulated to assess whether the system meets user expectations and project requirements.

6. Test Environment

Testing is conducted in a controlled staging environment that closely resembles the final deployment setup:

- Frontend hosted on a cloud-based deployment platform.
- Backend services implemented using serverless functions.
- A PostgreSQL-based database for persistent data storage.
- Testing tools include API testing utilities, unit testing frameworks, and standard web browser developer tools.

7. Roles and Responsibilities

- **Quality Assurance Lead:** Responsible for planning testing activities, designing test cases, executing tests, and documenting defects.
- **Development Team:** Responsible for implementing features, performing unit testing, and resolving identified issues.
- **Project Manager:** Oversees project timelines, coordination, and submission of deliverables.
- **Documentation Lead:** Maintains testing documentation and related artifacts in the project repository.

8. Entry and Exit Criteria

Entry Criteria:

- Core system features have been implemented and deployed to the staging environment.

Exit Criteria:

- All critical and high-priority test cases have passed.
- Major defects have been resolved or appropriately documented.
- The system is considered stable for academic review and demonstration.

9. Test Deliverables

The following artifacts are produced as part of the testing process:

- The Test Plan document.
- A detailed Test Case specification covering all major features.
- A Defect or Bug Report log documenting identified issues and resolutions.
- Supporting evidence of testing activities, such as screenshots, logs, and execution records.