Software Requirements Specification

for

University Placement Portal

Version 1.1 approved

Prepared by

Devanshee Vankani: CE145

Sahil Bhuva: CE017

Dharmsinh Desai University

Software Development Project

December 20, 2020

Table of Contents

Revision History

Name	Date	Reason for Changes	Version
1	26-12-2020	Initial Version	1.0
2	26-12-2020	Functional Requirement Refinement	1.1

1. Introduction

1.1 Purpose

The IT sector is expanding very rapidly and along with that the number of engineers, specifically the developers, graduating every year is exponentially increasing. This means that the graduates not only need to know the latest technology, but also need to be skilled in the domain they want to excel their future in. Now that one has well prepared himself/herself with the technology, it's equally important for him to present his skills in an effective way and get a job, because getting a job and gaining financial stability is the ultimate goal. So, this situation demands an authentic platform and a one stop shop where the job seeker can find apt information right from steps to prepare, practice and crack the toughest interviews. Also, the platform needs to be accessible easily and must be available to everyone so that the reputation of the university not only can be maintained but also improved

1.2 Document Conventions

No Document conventions used.

1.3 Intended Audience and Reading Suggestions

This document is intended for, such as

1. Developers
2. Project managers
3. Admin
4. Naïve users
5. Contributors and testers.

Final approval and acceptance will be done by the organization.

1.4 Product Scope

This project is mainly focused on the content area which is intended for the end users. Its function is to control the interaction between the end user and system. The targeted user groups are the students graduating in near future who will be the main benefit bearer.

1.5 References

- IEEE SRS Document
- Leetcode
- Geeksforgeeks

2. Overall Description

2.1 Product Perspective

The project is aimed at developing the application for the university students in search of jobs. The system is an application that can be accessed and effectively used. This system can be used to practice for interviews. The system also provides features of prediction of recruiters visiting the campus this year for hiring based on previous years. There will also be a feature that will enable naive users to learn the basic tech-stack required by most companies.

2.2 Product Functions

The basic features of the system is: -

- Prediction of Recruiters visiting this year based on past record
- Competitive Coding Practice Support
- Practice Aptitude Quiz Support
- Find Opportunities in MNC's hiring off campus
- Blog (article) support

More details will be discussed in section 4 of the document.

2.3 User Classes and Characteristics

Experts: They are the core users and are able to add articles to the system and permit them to access the expert level features of the system.
Naive (Anonymous): They have access to the facilities provided by the system.
Administrator: They are the root users and are able to manage data directly with access to every feature.

2.4 Operating Environment

Server System: - Any operating system with Angular, Node.js and MongoDB installed, good storage capacity and good internet connection.
Client System: - Any web browser or app and good internet connection.

2.5 Design and Implementation Constraints

Each user must keep their password as confidential. Moreover the user must have an individual ID for creating a login in the platform. Only Administrator can control user addition and deletion in the system. Also, this group has access to all the official activities. When users use a legacy browser then the UI of the website sometimes does not work.

2.6 User Documentation

The website provides online help support. The website will provide a user guide manual which will help users to work on the software.

2.7 Assumptions and Dependencies

- Good internet is a must.
- While practicing competitive coding, files will not be encrypted.

3. External Interface Requirements

3.1 User Interfaces

Login Screen: This is a security feature inbuilt in the software for the verification of authenticated users over intruders.
Expert Account: This enables the user to perform the expert's level activities like giving articles.
Admin Account: This enables the user to manage all of the system features.

3.2 Hardware Interfaces

Experts will be having read-write only access to articles. Anonymous users will be having read-only access to the database with privileges.

3.3 Software Interfaces

System is a multi-user, multi-tasking environment. It enables the users to interact with basic features. It uses Angular as the front-end programming tool and Node.js and MongoDB as the backend application too.

3.4 Communications Interfaces

System will interface mobile data connections or wireless data connection to maintain communication between devices. We will be using forms to collect the data from the users.

4. System Features

The system mainly focuses on making all the information (about hiring and placement) available at one place. There are intelligent technologies and models used to automate the entire task and make the administration work easy and simple. The system supports the following features and may also be extended further by adding various other modules.

R.1 MANAGE PLACEMENT / HIRING

R.1.1 VIEW PAST RECORD OF PLACEMENT (on-campus)

Input: User action.
Output: Table displaying the records.
Description: All the users can view the record of placement of previous years.
Process: Fetch from database and process it to show to end users.

R.1.2 VIEW PREDICTION OF COMPANIES VISITING THE CAMPUS

Input: User action.
Output: Table displaying information like no. of companies, vacancies or requirements, etc.
Description: All the users can see the prediction of the companies visiting the campus current year based on the machine learning model.

R.1.3 VIEW HIRING OPTION FROM COMPANIES HIRING OFF-CAMPUS

Input: User action.
Output: Table displaying information like no. of companies, vacancies or requirements, etc.
Description: All the users can view the off-campus hiring opportunities from various companies.
Process: Based on user input process data and display it to end users.

R.1.4 ADD PLACEMENT RECORDS

Input: User action.
Output: Form in which expert will enter placement detail.
Description: Experts can add the placement records which can be viewed by all other system users.
Process: The data will be validated and added to the database if valid data is valid and expert is authenticated.
Precondition: The form for adding record is shown if the user is signed in as an expert.
Postcondition: Final table displaying all the records along with a new record is displayed.

R.2 USER MANAGEMENT

R.2.1 MANAGE EXPERT

R.2.1.1 REGISTER EXPERT DETAILS

Input: User action.
Output: Registration form to be filled.
Description: New experts need to be registered. The registration form will be filled by the admin.
Process: User data is validated and added to the database if data is valid.
Precondition: Registration form is displayed to admin.
Postcondition: Success/Failure Registration message displayed.

R.2.1.2 UPDATE EXPERT DETAILS

Input: ExpertId.
Output: Update form.
Description: Update existing expert details by admin.
Process: A Form containing current details of the expert is displayed to the admin.

R.2.1.3 DELETE EXPERT DETAILS

Input: ExpertId.

Output: Delete confirmation message.
Description: Delete the details of the expert leaving the institute by admin.
Process: The expert record will be removed from the database.

R.2.2 MANAGE ADMIN

R.2.2.1 REGISTER ADMIN DETAILS

Input: User action.
Output: Registration form.
Description: Registration details of new admin in the institute.
Process: Admin details added to the database.
Postcondition: Redirected to dashboard.

R.2.2.2 UPDATE ADMIN DETAILS

Input: AdminId.
Output: Update Form.
Description: Update the details of the existing admin.
Process: Verify data in the database if found valid update data.

R.2.2.3 DELETE ADMIN DETAILS

Input: AdminId.
Output: Delete Acknowledgement message.
Description: Delete the details of the admin leaving the institute by admin.
Process: Remove the corresponding record from the database

R.3 PRACTICE COMPETITIVE CODING

R.3.1 ADD PROBLEM STATEMENT

Input: User action.
Output: Form to upload problem statement.
Description: Experts or admins can add the problem statement for competitive coding.
Process: Valid data stored in the database.
Precondition: Users must be logged in and have the role of expert or admin.

R 3.2 UPDATE PROBLEM STATEMENT

Input: User action, ProblemStatementId.
Output: Form to update problem statement.
Description: Allows experts or admins to update the existing problem statement.
Process: Validate data if validation passes save changes to database.
Precondition: Users must be logged in and have the role of expert or admin.

R 3.3 DELETE PROBLEM STATEMENT

Input: User action, ProblemStatementId.
Output: Delete Confirmation Message.
Description: Allows experts or admins to delete the problem statement.
Process: Remove the data from database
Precondition: Users must be logged in and have the role of expert or admin.

R 3.4 VIEW PROBLEM STATEMENT

Input: User action.
Output: Display problem statement.
Description: Allows all the system users to view the problem statements.
Process: Fetch problem statement from database.

R 3.5 SOLVE A PROBLEM

Input: User action, file.
Output: Form to upload file.
Description: Allows all the users to solve the problem by letting them upload the file.
Process: Display a form which takes a file from the user
Postcondition: Go to the judgment video.

R 3.6 GET JUDGEMENTS

Input: User action.
Output: Result of the code execution (no of test cases passed/failed, execution time, CPU cycles used, etc.).
Description: Allows students to get the judgements of the problem he is solving.
Process: Run and Analyse the code file uploaded by the user.
Precondition: File must be uploaded.
Postcondition: Show the code details (like execution time, number of machine cycles etc)

R 3.7 VIEW SOLUTION

Input: User action.
Output: Solution code along with explanation and details like execution time, CPU cycles consumed, time complexity, etc.
Description: All the system users to view the solution of the problem.
Process: Display the existing solution uploaded by the expert to the user.

R.4 PRACTICE QUIZ

R.4.1 ADD QUIZ

Input: User action.
Output: Form to upload quiz.
Description: Allows experts or admins to add the quiz of different topics on the system.
Process: Validate data if found valid add to database.
Precondition: Users must be logged in and have the role of expert or admin.

R 4.2 UPDATE QUIZ

Input: User action, QId.
Output: Form to update quiz.
Description: Allows experts or admins to update the existing quiz.
Process: Validate data if found valid update database.
Precondition: Users must be logged in and have the role of expert or admin.

R 4.3 DELETE QUIZ

Input: User action, QId.
Output: Delete confirmation message.
Description: Allows experts or admins to delete the quiz.
Process: Validate data if found valid delete from database.
Precondition: Users must be logged in and have the role of expert or admin.

R 4.4 PRACTICE QUIZ

Input: User action.
Output: Questions in the form of Form.
Description: Allows all the users to take a quiz.
Process: Fetch quiz from database and process it to display.
Postcondition: After successful submission score will be displayed.

R.5. BLOG MANAGEMENT

Precondition: Users must be logged in and must have the role of expert.

R.5.1 ADD BLOG POST

Input: User input.
Output: Upload blog post Form.
Description: Upload a new blog post by an expert.
Process: Validate data if found valid save changes to the database.

R.5.2 UPDATE BLOG POST

Input: User input, BlogPostId.
Output: Update blog post Form.
Description: Update blog post details by expert.
Process: If validation passes then save changes to the database.

R.5.3 DELETE BLOG POST

Input: User input, BlogPostId.
Output: Delete Confirmation Message.
Description: Delete post by expert.
Process: If validation passes then save changes to the database.

R.5.4 VIEW BLOG POST

Input: User input.
Output: List of latest blog post articles.
Description: To view the post by all system users.
Process: If validation passes then save changes to the database.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

Performance should not be an issue because all the data queries involve small pieces of data. Changing the screen will require very little computation and thus will occur very quickly. Server updates could only take a few seconds as long as the device can maintain steady signal. System will also be up for 24x7 except the maintenance. There will be slight degradation in performance based on internet connectivity.

5.2 Safety Requirements

Only the project admin will be having access to the database at the back-end. So, the admin will have rights for modifications as well as direct updates to the database

5.3 Security Requirements

The server allows only authorized users the direct access to the internal structures. So, no naïve users or unauthorized people can harm the project. The unauthorized access to the server must be prevented.