CAREER INFORMATION AND RECRUITMENT PORTAL

PROJECT REPORT

submitted by

JINCY P JANARDHANAN (IEAREIT017)

ALEENA SUNNY (IEAREIT006)

ALKA BHAGAVALDAS K (IEAREIT007)

AMEENA SHIRIN (IEAREIT009)

to

the University of Calicut

in partial fulfilment of the requirements for the award of the Degree

of

Bachelor of Technology

in

Information Technology

Department of Information Technology

Institute of Engineering and Technology, University of Calicut, Thenjipalam

Kerala

November 12, 2020

DECLARATION

We undersigned hereby declare that the project report **Career Information and Recruitment Portal**, submitted for partial fulfilment of the requirements for the award of degree of Bachelor of Technology of the University of Calicut, Kerala is a bonafide work done by us under supervision of **Ms. Sruthimol M P**. This submission represents our ideas in our own words and where ideas or words of others have been included, we have adequately and accurately cited and referenced the original sources. We also declare that we have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in our submission. We understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other University.

Place: Thenjipalam

Date: November 12, 2020

Jincy P Janardhanan

Aleena Sunny

Alka Bhagavaldas K

Ameena Shirin

DEPARTMENT OF INFORMATION TECHNOLOGY
INSTITUTE OF ENGINEERING AND TECHNOLOGY
UNIVERSITY OF CALICUT, THENJIPALAM

CERTIFICATE

This is to certify that the report entitled **"CAREER INFORMATION AND RECRUITMENT PORTAL",** submitted by **Jincy P Janardhanan, Aleena Sunny, Alka Bhagavaldas K, Ameena Shirin** to the **UNIVERSITY OF CALICUT** in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Information Technology is a bonafide record of the project presented by them under our guidance and supervision. This report in any form has not been submitted to any other University or Institute for any purpose.

Ms. Anu Manohar	**Ms. Sruthimol M P**
Assistant Professor	Lecturer
Department of IT	Department of IT
(Coordinator)	(Coordinator & Guide)

ACKNOWLEDGEMENT

We would like to express our warmest gratitude to all the people who had been a great help, support and motivation throughout this project. Prima facie, we would like to thank the Almighty for giving us the wisdom and grace for the successful completion of our project. Forever we owe our deepest gratitude to our guide and coordinator, **Ms. Sruthimol M P**, Lecturer, Department of Information Technology, for her expert guidance, co-operation and immense encouragement in pursuing this project. With a profound sense of gratitude, we would like to express our heartfelt thanks to our coordinator, **Ms. Anu Manohar**, Assistant Professor, Department of Information Technology for her earnest co-operation, support and constant inspiration. We extend our sincere gratitude to all the teachers of the Department of IT and to all our friends for their help and support.

Jincy P Janardhanan

Aleena Sunny

Alka Bhagavaldas K

Ameena Shirin

ABSTRACT

This project aims to develop a web application to connect colleges, recruiters, students, and alumni on a single platform. It helps colleges for efficient management of their placement cell. Recruiters can use this application for their HR hiring activities related to campus placement. Students and alumni can further stay informed about various career choices available for them using the information portal. Finding jobs after graduation that best suits their interests and skill set is quite a challenging task for students or alumni. The difficulties in job finding arise from not having proper knowledge of the organization's objective, their work culture, and current job openings. Discerning the best candidate from a list of applicants is a primary responsibility for the recruiters. The web application provides an easy and convenient search feature for students or alumni to find their desired jobs and for recruiters to find the right candidate. Students and alumni can conveniently use this web application for job-seeking. Colleges verify student and alumni information before registering them on the platform. A server admin verifies college and recruiter registration on the platform. Only alumni can update their details on their own, whereas students cannot. The recommendation system available in the web application is very advantageous for students and alumni. The web application automatically generates a CV for students and alumni using their details and information, which can be attached to their job applications. This web application aims at making our society free of jobless students and graduates.

Contents

List of Figures

List of Tables

Abbreviations

HTML - HyperText Markup Language

CSS - Cascading Style Sheets

AJAX - Asynchronous Javascript And XML

XML - eXtensible Markup Language

REST - REpresentational State Transfer

API - Application Programming Interface

STOMP - Simple (or Streaming) Text Oriented Message Protocol

CV - Curriculum Vitae

HR - Human Resource

SRS - Software Requirements Specification

JDK - Java Development Kit

RAM - Random Access Memory

PC - Personal Computer

OS - Operating System

UI - User Interface

DFD - Data Flow Diagram

MVC - Model View Controllers

AI - Artificial Intelligence

Chapter 1
Introduction

Career Information and Recruitment Portal is a web application developed to connect colleges, students, alumni and recruiters on a single platform. It helps colleges for efficient management of their placement cell. Moreover, recruiters can use this application as a job board for their HR hiring activities related to campus recruitment. With this application, students and alumni can find and apply for job opportunities relevant to them and keep track of recruiter updates. Students and alumni can also stay informed about various career choices available for them using the information portal. Besides, the application allows students and alumni to request and receive recommendations from colleges, recruiters and fellow alumni. It helps them to add more value to their resumes.

1.1 Problem Statement

HR recruitment had always been a hard time for recruiters to choose the right talent that best meets their expectations and perfectly matches their job vacancies. It is a lengthy and tiring process in many companies and usually spans over multiple weeks. The process requires a great deal of human effort, planning, strategy and time. Many companies still have not moved on for online recruitment. Also, companies which have opted for e-Recruitment have concerns regarding credibility and trustworthiness of the details submitted by an applicant. Security and ease of use of the application is another concern for a recruiter.

College students and fresher graduates are very resourceful and a worthwhile consideration for most of the job positions in a company. The student and graduate applicant pool are readily available and probably an economical choice for a company. It benefits both the employer and the student. Students might require an industry expertise certification for completion of their course. Additionally, it is a plus to their work profile. With student hiring, recruiters get an excellent opportunity to leverage their profits on a capable and agile worker pool, and also find employees of great potential for their company. Moreover, many colleges are looking out for recruiters who can offer campus placement opportunities for their students. Even with all the rising demand and added benefits to it, there is not yet an efficient platform to connect these stakeholders.

After completing their graduation or under-graduation, most students are in no man's land of thoughts while deciding to pursue their higher studies or get employed soon. Many talented graduates are jobless and do not know various possible career options for them and also does not have access to active job profiles that might fit their qualifications and skills. It is not an easy task for an individual to keep track of job updates and hiring activities carried out by a company. Moreover, having worthful and prised recommendations on their resume can increase the value of any candidate. However, receiving recommendations may not be very easy. There is a high chance that students might miss many potential opportunities to get hired by a company.

1.2 Motivation and Objective

A web application to connect colleges, students, alumni and recruiters on a single platform is obviously, the right choice to solve problems related to campus recruitment. There would be no more hassle for colleges to find companies who can offer placement opportunities for their students and alumni. Managing job listings, tracking received job applications and communicating to applicants from a single platform can considerably reduce the effort of an HR manager at a company.

A web application to find and apply for jobs is an easy-to-use substitute for a manual and paper-based hiring process. It is more efficient to store job information and job applications on an online database which supports fast queries. Using a web application to interact with an

online database saves time for both recruiters as well as students (and alumni). Besides, a paperless economy makes us one step closer to environmental sustainability.

All stakeholders would prefer a secure and reliable web application and would not want any trust issues or data compromises. Thanks to modern technologies that we have all the tools to secure a web application in the best possible way. Spring Security offers a standard for securing Spring-based web applications. The web application can allow colleges to verify student and alumni data before registering them on the platform. Moreover, the web application allows a server admin to verify all college and recruiter registrations on the platform. Thus, we can provide all stakeholders with reliable and trustworthy information.

Requesting and receiving recommendations can be made easy with a web application. Students and alumni can take advantage of a web application with an added feature for the recommendation system. With all the student (or alumni) details available on the platform, it is easy to generate an automatic CV for all the students (and alumni) on the platform. This resume can be easily attached to their job applications. Similar to the job search functionality, recruiters can use a resume search functionality to find relevant candidates for their job vacancies.

An information portal can be attached to this web application which allows students and alumni to discover various career and higher education opportunities available for them. It can help increase awareness and knowledge about career choices for a graduate. Hopefully, it can help us to build a society of less jobless people.

Chapter 2
Literature Review

2.1 Existing Methodologies

After completion of a graduate or undergraduate course, most students apply for a job. Colleges will be looking for recruiters who can provide job opportunities for their students. Besides, recruiters would hire students with the right skills for their job vacancies. Students have to apply for a job and keep track of the recruiter updates. But most of the time, all these works are done manually.

In the existing system, data processing is mostly manual. It has several problems such as efficient storage and retrieval of the information. Keeping track of information becomes a tedious task. By implementing an online database system, we can overcome most of the limitations of the present system. Using a web application, we can considerably reduce manual labour and increase the efficiency and accuracy of the system. Moreover, it reduces the time consumption for various activities to a great extent.

Chapter 3

Proposed System and Feasibility Study

3.1 Proposed Solution

We propose a web application for career information and recruitment to solve the problems related to campus placements. It connects college, students, alumni, and recruiters. It provides an excellent opportunity for students and alumni to find jobs which are suitable for them. A college admin registers all the students of their college on the web platform. Alumni of a college can request registration on the platform to their respective colleges. Both students and alumni can personalize their profile by adding a description, profile photo, projects, skills and achievements. The application uses this information to generate an automatic CV for the student or alumni.

Students and alumni can take advantage of the recommendation system on the platform. Recommendations help students and alumni to add more value to their profile. When students or alumni apply for available jobs from the platform, the recruiters receive their applications sorted according to their profile score. The profile score is calculated based on an applicant's work experiences, educational qualifications, skills and recommendations on their profile. On our platform, it is easy for applicants and recruiters to communicate with each other using the chat feature. The proposed system provides fast operation and low-cost expense than the old system. The application also helps students and alumni to find information about local and international higher studies and job opportunities.

3.2 Technical Feasibility

The technical feasibility assessment focuses on technical resources available to the developers. The assessment helps us to determine whether the technical resources meet the capacity to convert the ideas into a working system. Technical feasibility involves the evaluation of hardware, software, and other technical requirements of the proposed system [1].

The end-user specifications, software and hardware specifications to develop the proposed system are mentioned in sections 4.1 through 4.3. They are easily available or configurable by an end-user and/or developer. Hence the proposed solution is technically feasible.

3.3 Operational Feasibility

Operational feasibility studies how a project satisfies the requirements identified in the requirement analysis phase. It involves undertaking a study to analyze and determine whether-and how-well the requirements are met by completing the project [1].

All the functional and non-functional requirements for the proposed solution are listed out in section 4.4. These functional and non-functional requirements can be easily implemented using the technologies available. Hence the solution is operationally feasible

3.4 Economic Feasibility

Economic feasibility assessment involves a cost/benefits associated with a project, helping us to determine the viability, cost, and benefits associated with a project before financial resources are allocated [1].

The cost-benefit analysis of the system is carried out accordingly. The proposed system can be built and used using basic hardware and software requirements. Moreover, it is easy to maintain the system.

We are using an iterative and incremental model for development of the project which provides more flexibility for development rather than a rigid waterfall model. However, since our learning curve for the various technologies used can affect the development process, the iterative and incremental model is a more economic choice. Hence, our proposed solution is economically feasible with respect to all the technical and development constraints.

3.5 Schedule Feasibility

Schedule feasibility is the most important for project success; after all, a project will fail if not completed on time. In scheduling feasibility, we estimate how much time the project will take to complete [1].

Using an iterative and incremental model helps us for the successful completion of this project. Moreover, the technologies used in developments are time-saving. We are using GitHub for project management. It helps us in schedule maintenance, thus making our project schedule feasible.

Chapter 4
Requirements Gathering and Analysis

A requirement is a necessary attribute in a system, a statement that identifies a capability, characteristic, or quality factor of a system in order for it to have value and utility to a customer or user [2, pp. 1 - 2]. Requirements gathering or requirements elicitation is the process of gaining an understanding of the customers' and users' needs for the planned system and their expectations of it [2, p. 4]. Requirements analysis is a structured (organized) method to understand the attributes that will satisfy a customer need [2, p. 222].

4.1 End User Specification

End-users are people who uses a computer application, as opposed to those who developed or support it [3]. End user specifications are pre-requisite functionalities required by an end-user to use the developed system. End-user specifications for the proposed system are as follows:

- **PC with minimum requirements**
- **Stable network connection**

4.2 Software Specification

Software specifications include the pre-requisite software required to develop the proposed system. Software specifications for the proposed system are as follows:

- **Operating System:**
  - **Windows 7+**
  - **Mac OS X Yosemite 10.10+**
  - **Linux: 64-bit Ubuntu 14.04+, Debian 8+, openSUSE 13.3+ or Fedora Linux 24+**
- **Front end: Thymeleaf**
- **Back end: Spring boot, Spring security and MongoDB**
- **Web server: Embedded Tomcat**
- **Supported browsers: Chrome, Firefox, Internet Explorer 9+**
- **Java JDK 1.8+**

4.3 Hardware Specification

Hardware specifications include the pre-requisite hardware required to develop the proposed system. Hardware specifications for the proposed system are as follows:

- **4 GHz minimum, multi-core processor**
- **RAM: Minimum 2 GB**
- **Hard disk space: Minimum 10 GB**

4.4 SRS Document

The software requirements specification (SRS) specifies the requirements for a computer software configuration item and the methods to be used to ensure each requirement has been met [4]. Benefits of documenting the SRS include [5]:

- It provides a realistic basis for estimating product costs, risks and schedules.
- It provides an informed basis for deploying a product to new users or new operational environments.
- It provides a basis for product enhancement.
- It forces a rigorous assessment of requirements before design can begin and minimizes later redesign.

- It establishes the basis for agreement between the acquirers or suppliers on what the product is to do (in market driven projects, the user input may be provided by marketing).
- Organizations can use the specifications to develop validation and verification plans.

Functional requirements and non-functional requirements for the proposed system according to the SRS document is as follows.

4.4.1 Functional Requirements

A function is a useful capability provided by one or more components of a system. Functional requirements describe what the system or software must do [2, p. 72].

Functional requirements for the proposed system are as follows.

1. Career Information

Description: This section includes webpages for information related to local and international higher studies and job opportunities information for all branches. Users can click on the corresponding menu option to navigate to the respective page.
Input: User clicks on a menu option.
Output: Corresponding webpage is loaded in the web browser.

2. Student, Alumni and Recruiters - Portal

2.1. Colleges Portal

2.1.1. Registration

Description: Allow registration of new colleges. By clicking on register as a college link, the user receives the registration form to fill out college details. On submitting the filled application form, a new college registration request is sent to the server.
Input: User clicks new college registration, completes and submits the registration form.

Output: Server receives registration request on server homepage.

2.1.2. Opt Out

Description: If the college admin wishes to remove the college from this portal, he/she can submit an opt out request by clicking the link for opt out.
Input: User submits opt out request.
Output: Server receives the request.

2.1.3. Login

Description: Users can login using their registered email id and password. Server verifies the login details and grants access to the user session.
Input: User submits login credentials.
Output: Server verifies and grants or declines user session according to successful validation or failure of authentication.

2.1.4. Request for Forgot Password

Description: User can request for resetting password, if he/she forgets the password, by clicking on the forgot password link. On clicking on forgot password link, the user will be prompted to submit the registered email id.
Input: User submits forgot password request for a registered email id.
Output: Server receives the request.

2.1.5. Update Details

Description: Colleges can update their communication details by clicking on the update details link. It redirects to a form for editing the details. After making the necessary changes, the user can click on save button to save the changes.
Input: User makes changes in the college details using the edit form and clicks the save button.
Output: The new details are saved to the database.

2.1.6. Student and Alumni Management

2.1.6.1. Add New Student Enrolments of the College to the Portal

Description: Colleges can add multiple new students by uploading an excel file containing all the details of new student enrolments (name, contact number, email, communication address, educational qualifications) by clicking add new students and uploading the file.
Input: User uploads an excel spreadsheet of student details.

Output: All new student entries are added to the student database and students get login credentials via email.

2.1.6.2. Add Alumni Students

Description: Colleges can receive alumni registration requests on homepage. The college admin verifies the alumni details and approves or cancels the registration by clicking verify or cancel registration link. Applicants get notified by email regarding approval or cancellation of registration request.
Input: Alumni requests for registration.
Output: College admin verifies or cancels the registration by clicking verify or cancel registration button.

2.1.6.3. Remove Alumni Students

Description: Colleges can receive alumni opt out requests on homepage. On clicking on the request, he/she is directed to a list of all related records of the alumni out of which information to be retained can be selected. The admin can click on the delete alumni button to retain all selected records and remove other records.
Input: Alumni requests for opt out.
Output: College admin approves the request by selecting the records to keep and clicking delete alumni button.

2.1.6.4. Update Student Details

Description: When some student's details are to be updated or semester marks to be uploaded the admin can either upload a new excel spreadsheet file of the details by clicking the edit student details in the students' tab, or selecting individual student with email search and clicking on edit details. All records of the students with the login details specified in the spreadsheet will be updated.
Input: Student details are submitted via spreadsheet or edit details form.
Output: Updated student details are stored in the database.

2.1.6.5. Recommend Student or Alumni

Description: Colleges receive recommendation request from student or alumni on dashboard. The college admin can submit recommendation letter(s) from faculties by clicking on the request and selecting upload files.
Input: Student or alumni requests for recommendation.
Output: College admin uploads recommendation letter(s) from faculty.

2.2. Student Portal

2.2.1. Login

Description: Users can login using their registered email id and password. Server verifies the login details and grants access to the user session.
Input: User submits login credentials.
Output: Server verifies and grants or declines user session according to successful validation or failure of authentication.

2.2.2. Request for Forgot Password

Description: User can request for resetting password, if he/she forgets the password, by clicking on the forgot password link. On clicking on forgot password link, the user will be prompted to submit the registered email id.
Input: User submits forgot password request for a registered email id.
Output: Server receives the request.

2.2.3. Personalization

Description: Students can add skills, interests, experience, project links, awards and honours, organizations, profile pic and a description to their profile by adding or editing details from the personalization tab. An automatic CV is generated with these details if the student does not upload a CV.
Input: Student edits personalization details from the personalization tab.
Output: Updated details are stored in the student's database. Automatic CV is generated with these details if no uploaded CVs of the student are available.

2.2.4. Follow

Description: Students can follow companies and topics of interests, and set notification preferences for each company or topic from the follow tab and selecting a company or topic and preference from the options indicated against it.
Input: Student selects a company or topic to follow and indicates notification preference for the same.
Output: The follow details are stored or updated in the student's database.

2.2.5. Job Feed

Description: Students receive job feed in their homepage according to the follow details in his/her student record. Students can click on each job listing and view its details.
Input: Student follow details from database.
Output: Job listings are shown in the student's homepage according to the follow details.

2.2.6. Notifications

Description: Students receive notifications in the notifications tab about new job listings from a company or a following topic according to his/her notification preferences for that company or topic.
Input: Student follow details from database.

Output: Job listing notifications are shown in the student's notifications tab according to notification preferences in follow details.

2.2.7. Apply for Jobs

Description: On clicking apply button on a job details page, students will be prompted to enter description for applying. This can be submitted by clicking the submit button. The recruiter gets the applicant's CV and description for applying in his/her applications tab.
Input: Students submit a job application.
Output: Recruiter receives the applicant's CV and description for applying in his/her applications tab.

2.2.8. Request Recommendations

Description: Students can request for recommendations from his or her college or from an alumnus by searching for the name of the alumnus and submitting a request for recommendation by the request recommendation button. The college / alumnus receives the request on dashboard.
Input: Student requests recommendation from college or alumnus.
Output: College or alumni receives the request for recommendation dashboard.

2.2.9. Reply to Recruiters

Description: Students can receive messages from recruiters and reply them in the chat feature.
Input: Student receives message from recruiters in the chat feature.
Output: Student replies to the message in the chat feature.

2.3. Alumni Portal

2.3.1. Registration

Description: Allow registration of new alumni. By clicking on register as an alumnus link, the user receives the registration form to fill out personal details and college details. On submitting the filled application form, a new alumni registration request is sent to the respective college admin.
Input: User clicks new alumnus registration, completes and submits the registration form.
Output: College admin receives registration request on college homepage.

2.3.2. Opt Out

Description: If an alumnus wishes to remove his company from this portal, he/she can submit an opt out request by clicking the link for opt out. College admin receives the request from alumni on dashboard.
Input: Alumni submits opt out request.
Output: College admin receives the request on his dashboard.

2.3.3. Login

Description: Users can login using their registered email id and password. Server verifies the login details and grants access to the user session.
Input: User submits login credentials.
Output: Server verifies and grants or declines user session according to successful validation or failure of authentication.

2.3.4. Request for Forgot Password

Description: User can request for resetting password, if he/she forgets the password, by clicking on the forgot password link. On clicking on forgot password link, the user will be prompted to submit the registered email id.
Input: User submits forgot password request for a registered email id.
Output: Server receives the request.

2.3.5. Personalization

Description: Alumni can add or update personal details, higher education details, skills, interests, experience, project links, awards and honours, organizations, profile pic and a description to their profile by adding or editing details from the personalization tab. An automatic CV is generated with these details if the alumnus does not upload a CV.

Input: Alumnus edits personalization details from the personalization tab.

Output: Updated details are stored in the alumni database. Automatic CV is generated with these details if no uploaded CVs of the alumnus are available.

2.3.6. Follow

Description: Alumni can follow companies and topics of interests, and set notification preferences for each company or topic from the follow tab and selecting a company or topic and preference from the options indicated against it.

Input: Alumnus selects a company or topic to follow and indicates notification preference for the same.

Output: The follow details are stored or updated in the alumni database.

2.3.7. Job Feed

Description: Alumni receive job feed in their homepage according to the follow details in his/her record. He/she can click on each job listing and view its details.

Input: Student follow details from database.

Output: Job listings are shown in the alumnus's homepage according to the follow details.

2.3.8. Notifications

Description: Alumni receive notifications in the notifications tab about new job listings from a company or a following topic according to his/her notification preferences for that company or topic.

Input: Alumni follow details from database.

Output: Job listing notifications are shown in the alumni's notifications tab according to notification preferences in follow details.

2.3.9. Apply for Jobs

Description: On clicking apply button on a job details page, alumni will be prompted to enter description for applying. This can be submitted by clicking the submit button. The recruiter gets the applicant's CV and description for applying in his/her applications tab.
Input: Alumni submit a job application.
Output: Recruiter receives the applicant's CV and description for applying in his/her applications tab.

2.3.10. Request Recommendations

Description: Alumni can request for recommendations from his or her college or from a fellow alumnus by searching for the name of the alumnus and submitting a request for recommendation by the request recommendation button. The college / alumnus receives the request on dashboard.
Input: Alumni requests recommendation from college or alumnus.
Output: College or alumni receives the request for recommendation dashboard.

2.3.11. Recommend Students or Fellow Alumni

Description: Alumni receive recommendation request from students or fellow alumni on dashboard. He/she can submit a recommendation letter by clicking on the request and selecting upload files.
Input: Student or alumni requests for recommendation.
Output: The alumni uploads a recommendation letter.

2.3.12. Reply to Recruiters

Description: Alumni can receive messages from recruiters and reply them in the chat feature.
Input: Alumni receives message from recruiters in the chat feature.
Output: Alumni replies to the message in the chat feature.

2.4. Recruiter Portal

2.4.1. Registration

Description: Allow registration of new recruiters. By clicking on register as a recruiter link, the user receives the registration form to fill out company details. On submitting the filled application form, a new company registration request is sent to the server.

Input: User clicks new recruiter registration, completes and submits the registration form.

Output: Server receives registration request on server homepage.

2.4.2. Login

Description: Users can login using their registered email id and password. Server verifies the login details and grants access to the user session.

Input: User submits login credentials.

Output: Server verifies and grants or declines user session according to successful validation or failure of authentication.

2.4.3. Request for Forgot Password

Description: User can request for resetting password, if he/she forgets the password, by clicking on the forgot password link. On clicking on forgot password link, the user will be prompted to submit the registered email id.

Input: User submits forgot password request for a registered email id.

Output: Server receives the request.

2.4.4. Opt Out

Description: If a recruiter wishes to remove his company from this portal, he/she can submit an opt out request by clicking the link for opt out.

Input: User submits opt out request.

Output: Server receives the request.

2.4.5. Update Details

Description: Colleges can update their communication details by clicking on the update details link. It redirects to a form for editing the details. After making the necessary changes, the user can click on save button to save the changes.
Input: User makes changes in the college details using the edit form and clicks the save button.
Output: The new details are saved to the database.

2.4.6. Create Job Listings

Description: Recruiters can create job listings by clicking listings tab and create new button. He/she will be prompted to enter a job title and description, salary offered and add all related tags (topics) for the job. On clicking submit button, a new job listing will be created.
Input: Recruiter submits job details for new job listing.
Output: A new job listing is created.

2.4.7. Edit / Delete Job Listing

Description: Recruiters can delete job listings by clicking listings tab and selecting an already created job listing. He/she will be directed to an edit details form. By clicking update button or delete button, the job listing will be updated or deleted accordingly.
Input: Recruiter clicks update or delete button on a job listing.
Output: The job listing is updated or deleted according to the button clicked.

2.4.8. Review CV

Description: Recruiters can view CV of all students and alumni on the portal following tags associated with job listings of the company from the Potential tab. The student profiles (name, profile picture, and description) will be listed in a sorted order according to a score calculated based on work experience, educational qualifications, skills and recommendations. On clicking a student's or alumnus's name, the recruiter will be directed the student's or alumni's CV.

From this page, the recruiter can click on contact button to message the student or alumnus regarding recruitment. The webpage directs to the chat tab with the student or alumni and recruiter can type his message and click on send button.
Input: Recruiter selects a CV and sends a message to the student or alumni.
Output: The student or alumni receives the messages in the chat tab.

2.4.9. Review Applications

Description: Recruiters can view all received application for a job listing from the applications page of the listing. The description for application along with student profile (name, profile picture) will be listed in a sorted order according to a score calculated based on work experience, educational qualifications, skills and recommendations. On clicking a student's or alumnus's name, the recruiter will be directed the student's or alumni's job application. He/she can click on view CV button to view the CV of the candidate. The recruiter can click on contact button to message the student or alumnus regarding recruitment. The webpage directs to the chat tab with the student or alumni and recruiter can type his message and click on send button.
Input: Recruiter selects a job application and sends a message to the student or alumni.
Output: The student or alumni receives the messages in the chat tab.

2.4.10. Recommend Student or Alumni Employees

Description: Recruiters can recommend current or past employees of the company registered as student or alumni in the portal by searching for the name of the employee and selecting recommend button. The recruiter can upload a recommendation letter by clicking on upload files on the redirected page.
Input: Recruiter selects a current or previous employee of the company and uploads a recommendation letter.
Output: The student or alumni receives the recommendation in his profile.

2.5. Server Admin

2.5.1. Verify / Cancel College and Recruiter Registration

Description: The server admin personally verifies details of registration of college users and recruiter users before allowing them login access. All new college and recruiter registrations pending verification will be displayed on the server homepage. The server admin can click on each registration to view the details submitted. If the admin finds the registration details are correct (credible), he/she can click verify registration button to verify the college or recruiter registration; otherwise, the admin can click cancel button to cancel the registration. On approval or cancellation of registration, email messages are automatically sent to the respective users. This is used for ensuring credibility of registered colleges and recruiters.

Input: Colleges and recruiters submit registration.

Output: Server admin clicks on new registrations of colleges and recruiters, verifies the details and clicks verify or cancel registration button. Autogenerated emails are sent to the respective users regarding confirmation or cancellation of registration.

2.5.2. Handle Opt Out Requests of Colleges and Recruiters

Description: When colleges or recruiters request for opting out from the application, the server admin gets the request on the server homepage. On clicking on the request, he/she is directed to a list of all related records of the user out of which information to be retained can be selected. The admin can click on the delete user button to retain all selected records and remove other records.

Input: Colleges and recruiters request for opt out.

Output: Opt out request is displayed on server homepage. Admin clicks on the request to view all related records. He/she selects records to be retained and deletes all other records by clicking on delete user button.

2.5.3. Close Active Login Sessions with No Activity for a Long Duration

Description: When user session is inactive for over 15 minutes, the session is automatically closed by the server.

Input: User session remains inactive for 15 minutes.
Output: Server closes the request.

2.5.4. Handle Forgot Password Requests

Description: When a user submits forgot password request, the server automatically sends a password reset link to the registered email of the user.
Input: User submits forgot password request.
Output: Server automatically sends a password reset link to the registered email of the user.

4.4.2 Non-Functional Requirements

Users have implicit expectations about how well the software will work. These characteristics include how easy the software is to use, how quickly it executes, how reliable it is, and how well it behaves when unexpected conditions arise. The non-functional requirements define these aspects about the system [6].

The non-functional requirements for the proposed system are as follows.

1. **Portability:**

   Since this is a web application it can be used from all supported browsers irrespective of the underlying operating system.

2. **Security:**

   The web application is developed using Spring boot framework for Java, Spring security core, and MongoDB RESTful authentication is used in the back end. The APIs are guarded by spring security REST. This is a great combination of a secure modern web application. The database used is highly secured and protected from unauthorized access.

3. **Usability:**

The web application will be available all the time since it is hosted on a real time server. It has a simple and user-friendly interface. Both skilled and unskilled users can use the application efficiently to meet their requirements. The user interface is designed to make the user to interaction simple and easy as possible.

4. **Reliability:**

Performance and reliability are two key components of the system. The application offers high performance and reliability of data and information as well as transactions of the database.

5. **Maintainability:**

The database used provides high performance. It can be used to store files of any size easily without any complications. In the case of a failure, it is easy to be administered. The system can be easily maintained by the admin users.

6. **General Constraints on Design and Development:**

The software and technologies used for development are Java, Spring boot framework for Java, Thymeleaf, and MongoDB database.

Chapter 5

System Design

Systems design is the process of defining the architecture, modules, interfaces, and data for a system to satisfy specified requirements [7]. In this chapter we try to briefly describe our system design.

5.1 System Architecture

A system architecture is the conceptual model that defines the structure, behaviour, and more views of a system [8]. The architecture diagram for the proposed system is as follows.

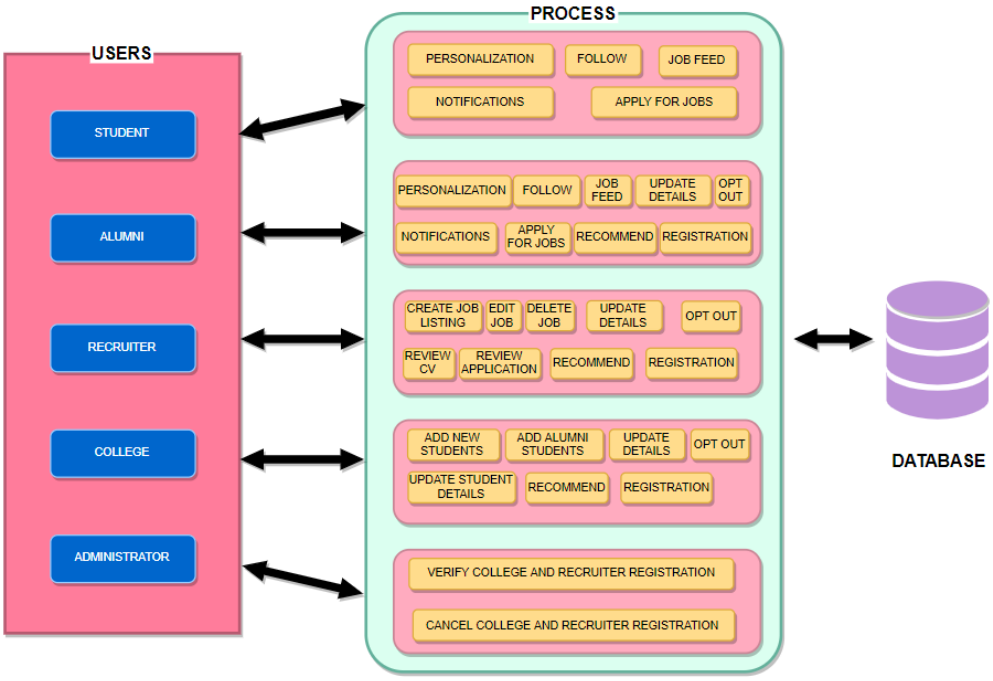

Figure 5.1. System Architecture

5.2 Interface Design

User interface design (UI) or user interface engineering is the design of user interfaces for machines and software, such as computers, home appliances, mobile devices, and other electronic devices, with the focus on maximizing usability and the user experience. The goal of user interface design is to make the user's interaction as simple and efficient as possible, in terms of accomplishing user goals (user-centered design) [9].

Various user interfaces used in our proposed system are as follows.

Table 5-1 - Interface Design

INTERFACE NAME	DESCRIPTIONS
Home	Default home page for the web application
Register	UI to redirect users to their respective registration pages
Terms and Conditions	UI to display terms and conditions for user registration
Register as College	UI for registration of colleges
Register as Recruiter	UI for registration of recruiters
Register as Alumnus	UI for registration of alumni
User Login	UI for all users to login
Forgot Password	UI for users to request for resetting forgot password.
Update Password	UI for users to update password.
Admin - Admin Panel	UI for admin users to approve/reject college/recruiter registrations
Profile - College	UI for colleges to view and update their profile
College - Admin Panel	UI for colleges to approve/reject new alumni, add and update students
Profile - Recruiter	UI for recruiters to view and update their profile
Manage Jobs	UI for recruiters to view a brief of job listings, delete jobs, redirect to update job, redirect to view all job applications/job applications for a specific job
Create or Update Job	UI for Recruiters to create or update a new job listing.
Recruiter - View Job	UI for Students to view created job listing on college dashboard.
Job Applications	UI for Recruiters to review received job applications, hire/reject the applicant, redirect to chat with the applicant

Profile - Student	UI for students/alumni to view and update their profile
Update Personalization	UI for students/alumni to personalize their profile by adding skills, experiences, etc.
Job Suggestions	UI for students/alumni to get job suggestions based on their skills.
Chat	UI to allow recruiters and job applicants to communicate with each other
Search Results	UI to show search results
Public Profile - College	UI to display the public profile info of a college
Public Profile - Recruiter	UI to display the public profile info of a recruiter
Public Profile - Student	UI to display the public profile info of a student/alumnus
Public Profile - Job	UI to display the public profile info of a job

5.3 Data Flow Diagrams

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It can be manual, automated, or a combination of both [10].

DFDs for the proposed system are represented in the following pages.

LEVEL 0 - CONTEXT DIAGRAM

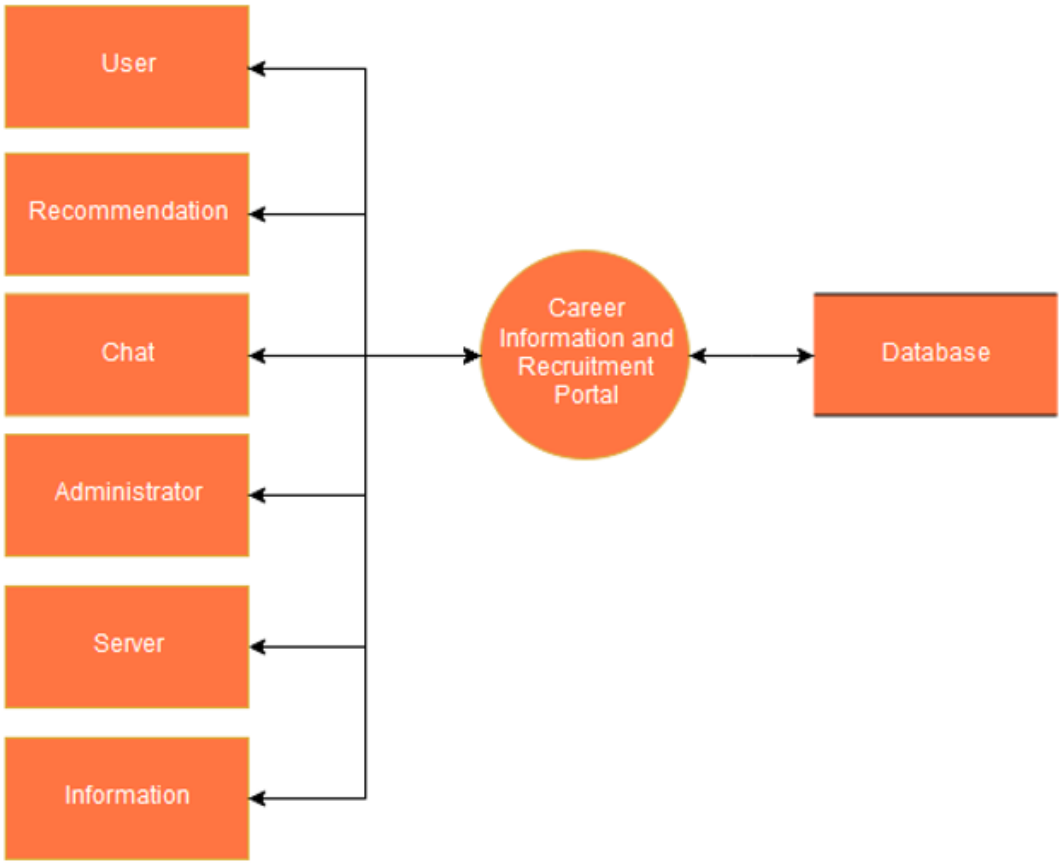

Figure 5.2 - Level 0 - Context diagram for Career Information and Recruitment Portal

5.3 Data Flow Diagrams

LEVEL 1 DIAGRAMS:

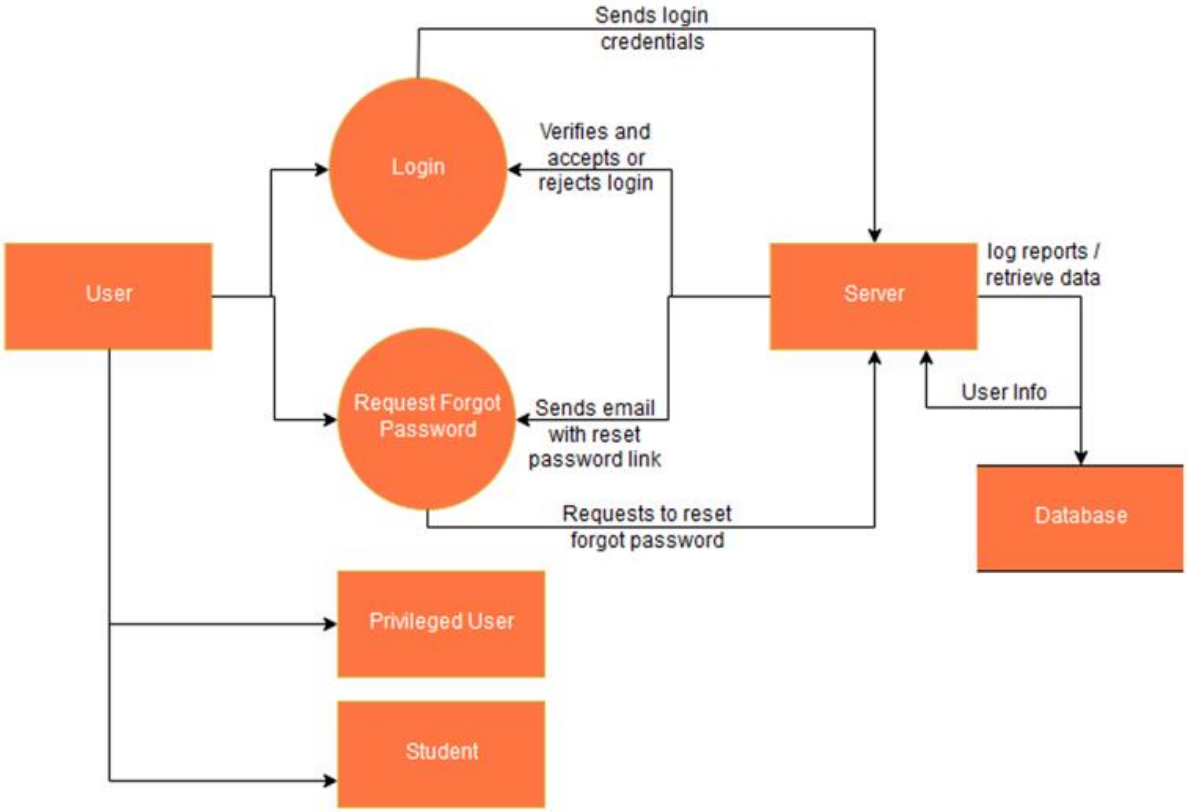

Figure 5.3 - Level 1.1 - User Module

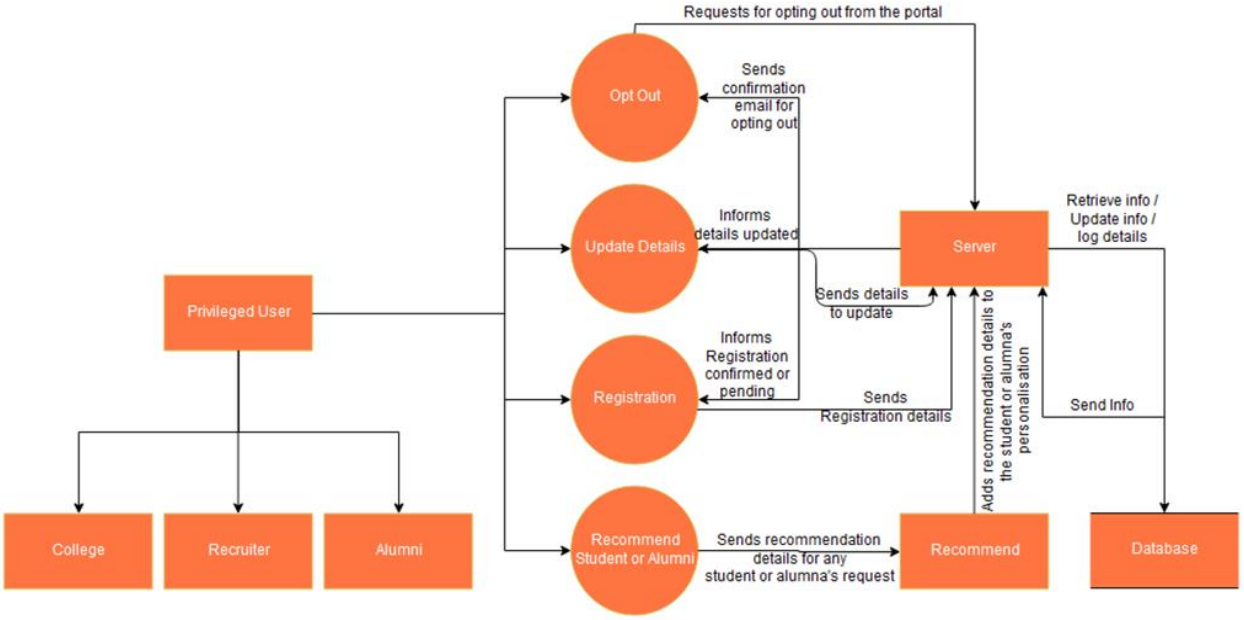

Figure 5.4 - Level 1.1.1 - Privileged User Module

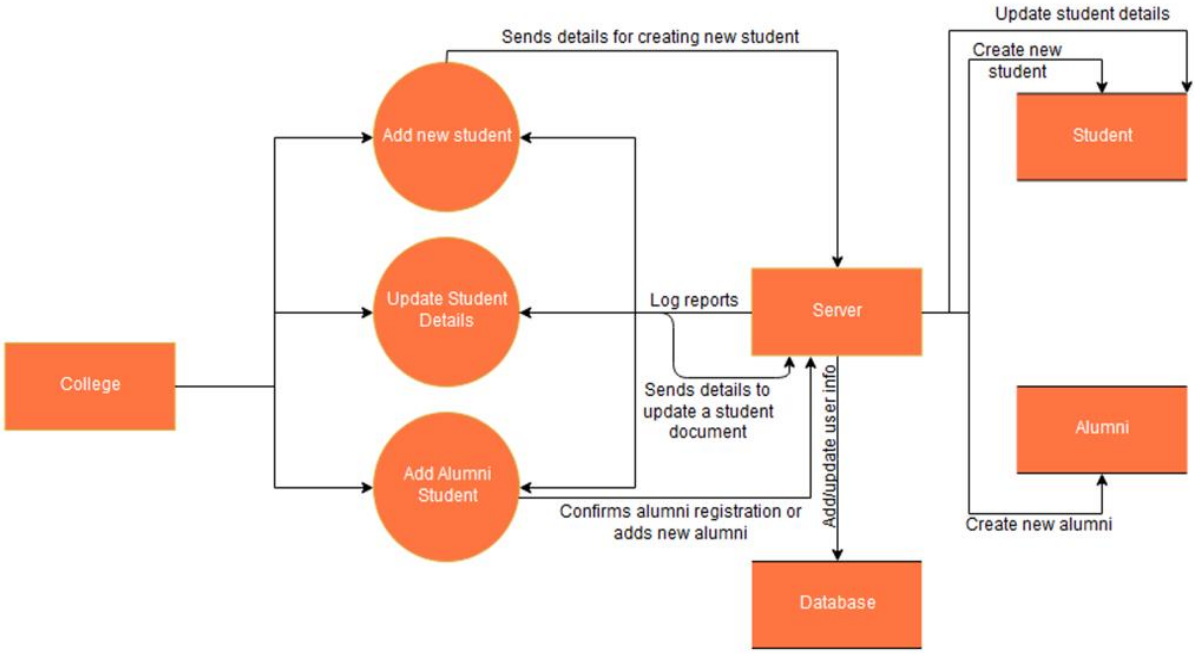

Figure 5.5 - Level 1.1.1.1 - College Module

5.3 Data Flow Diagrams

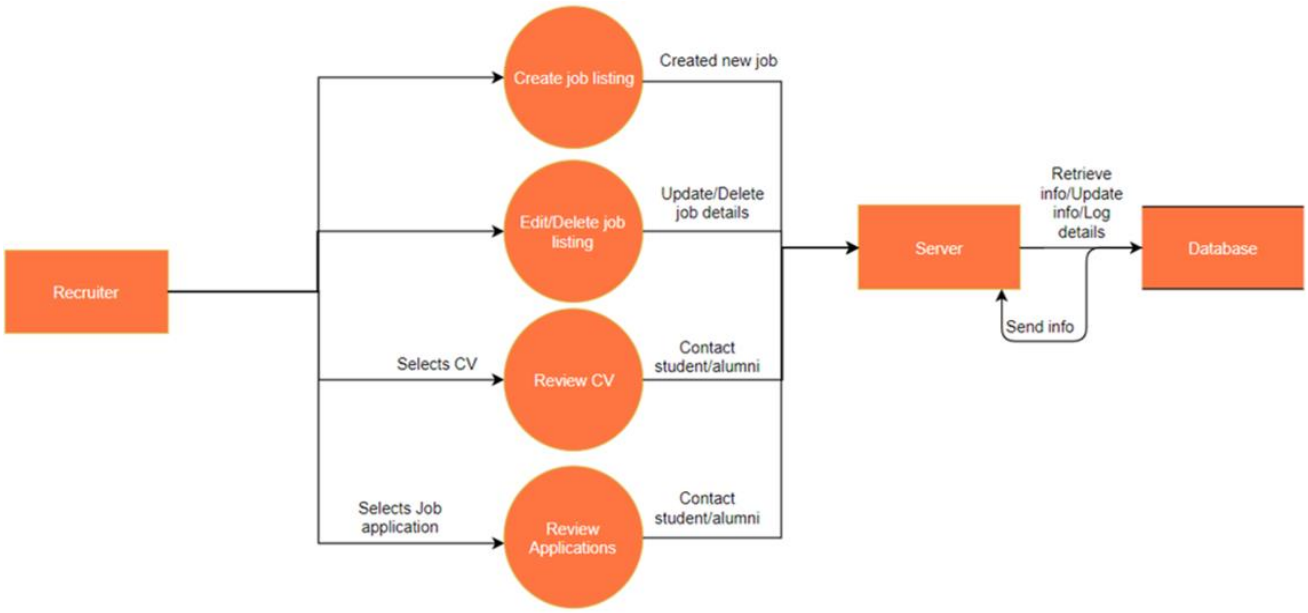

Figure 5.6 - Level 1.1.1.2 - Recruiter Module

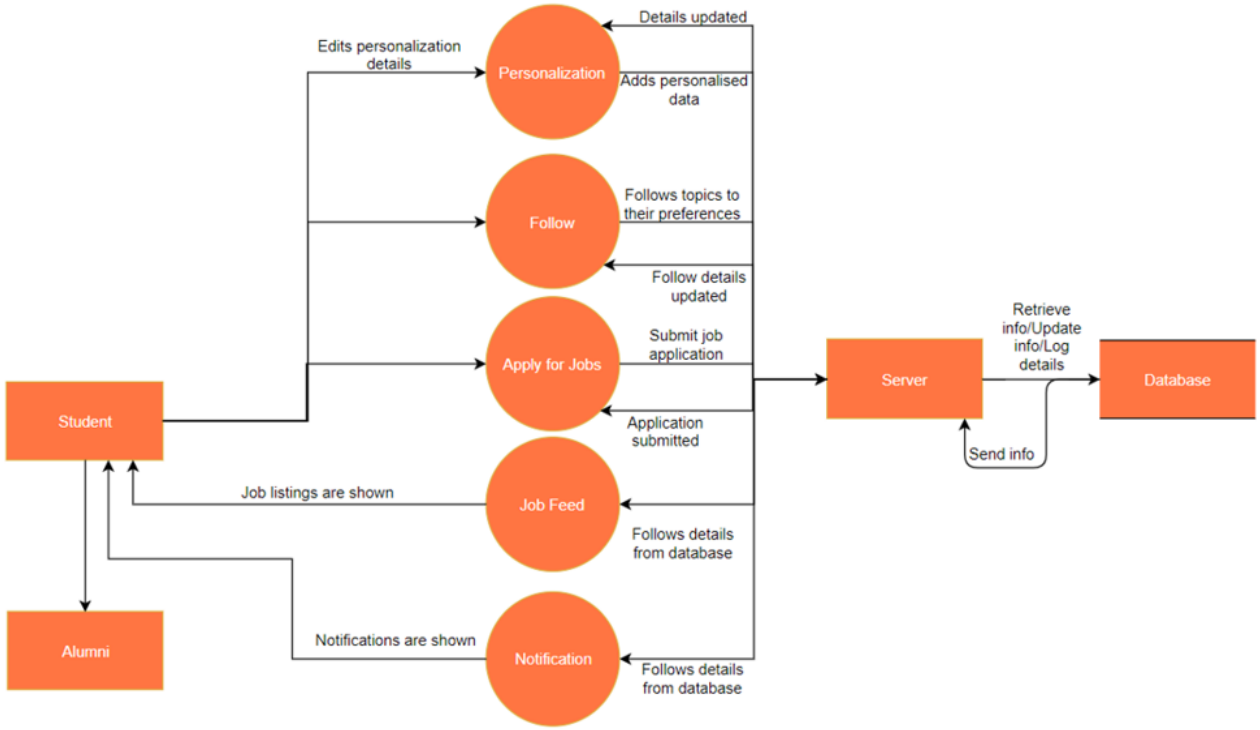

Figure 5.7 - Level 1.1.2 - Student Module

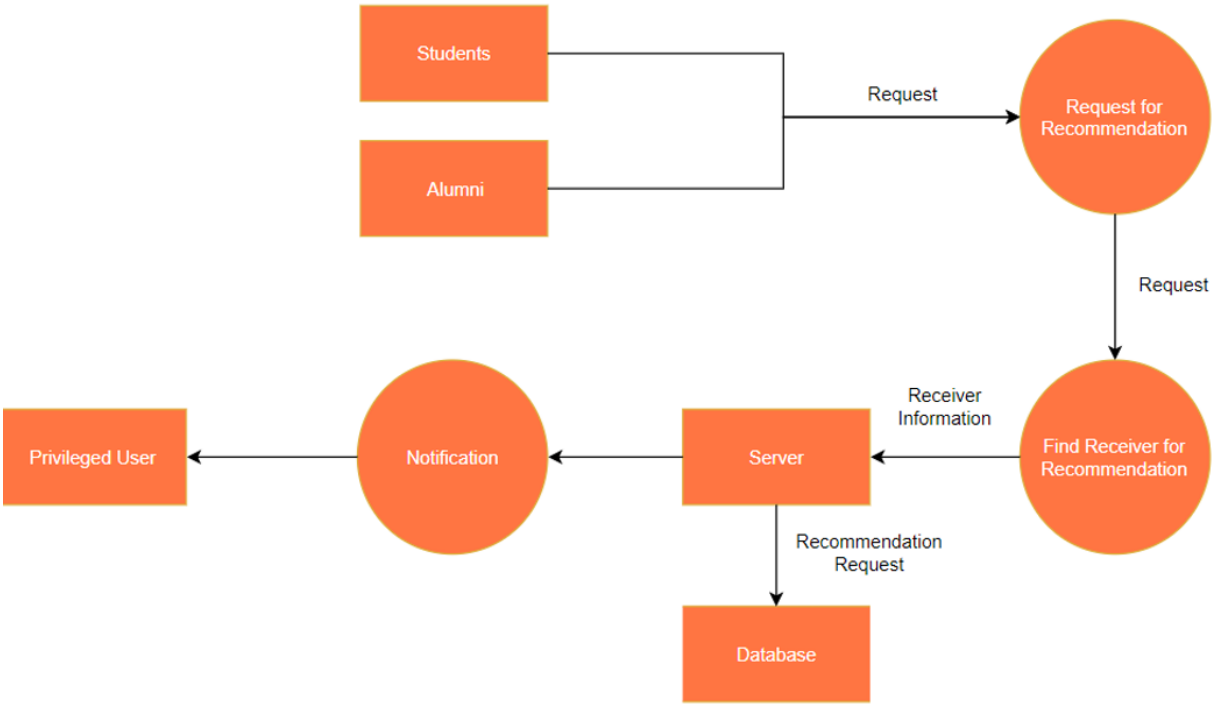

Figure 5.8 - Level 1.2.1 - Request Recommendation Module

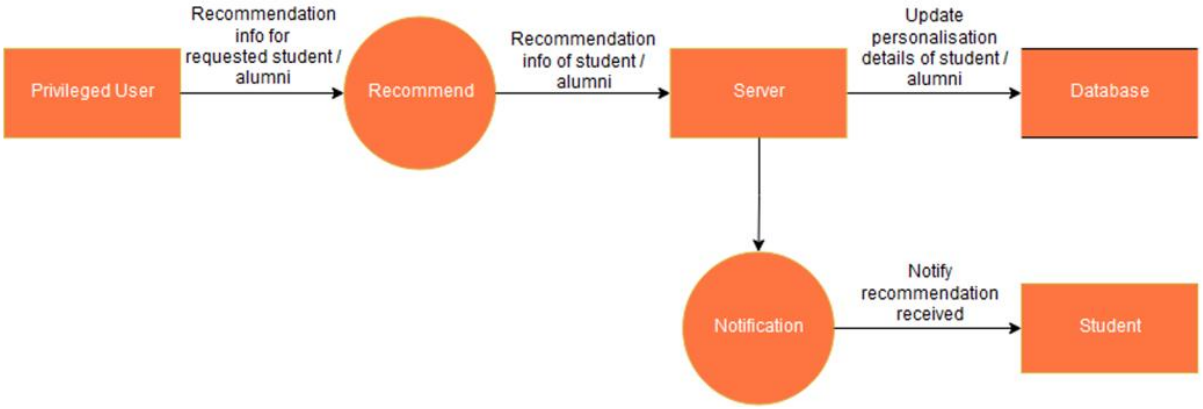

Figure 5.9 - Level 1.2.2 - Recommend Module

5.3 Data Flow Diagrams

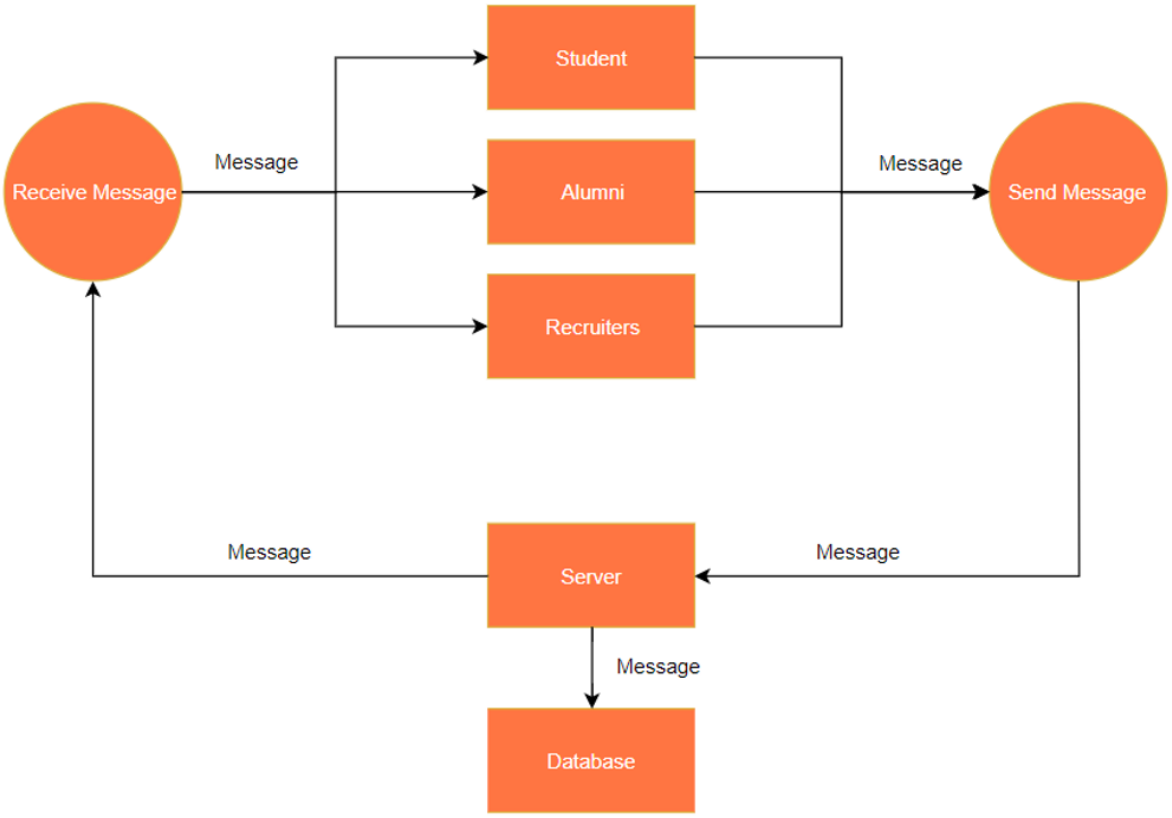

Figure 5.10 - Level 1.3 - Chat Module

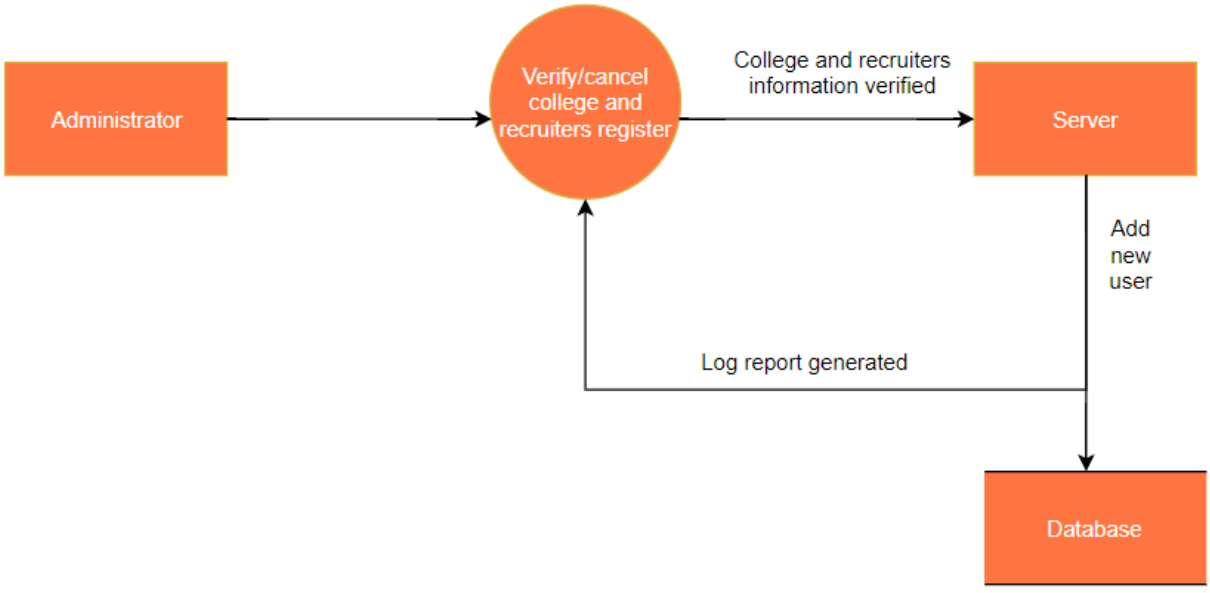

Figure 5.11 - Level 1.4 - Administrator Module

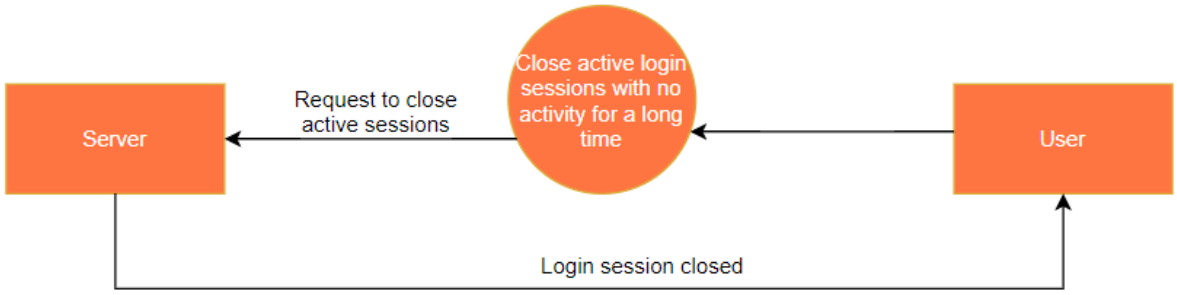

Figure 5.12 - Level 1.5 - Server Module

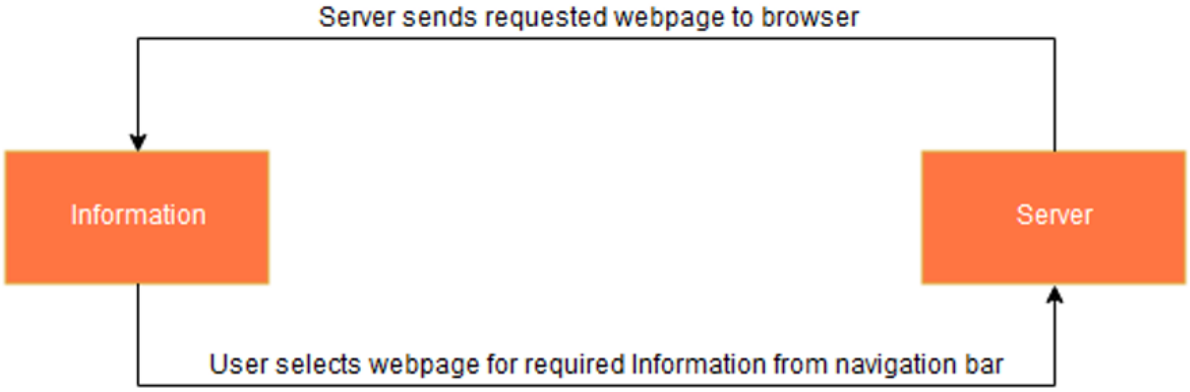

Figure 5.13 - Level 1.6 - Information Module

5.3 Data Flow Diagrams

LEVEL 2 DIAGRAMS:

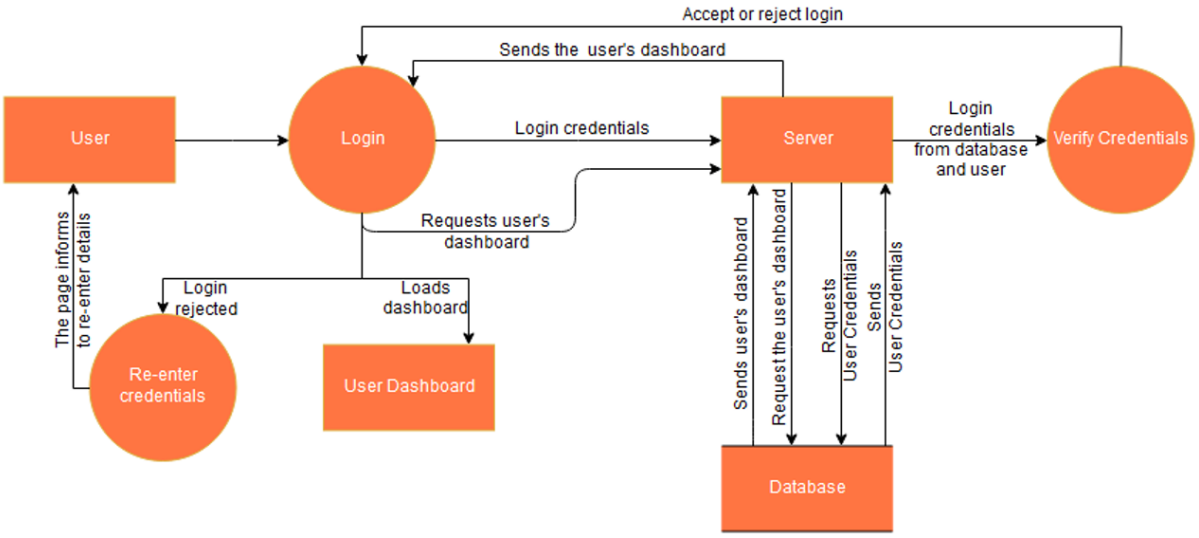

Figure 5.14 - Level 2.1 - User Login

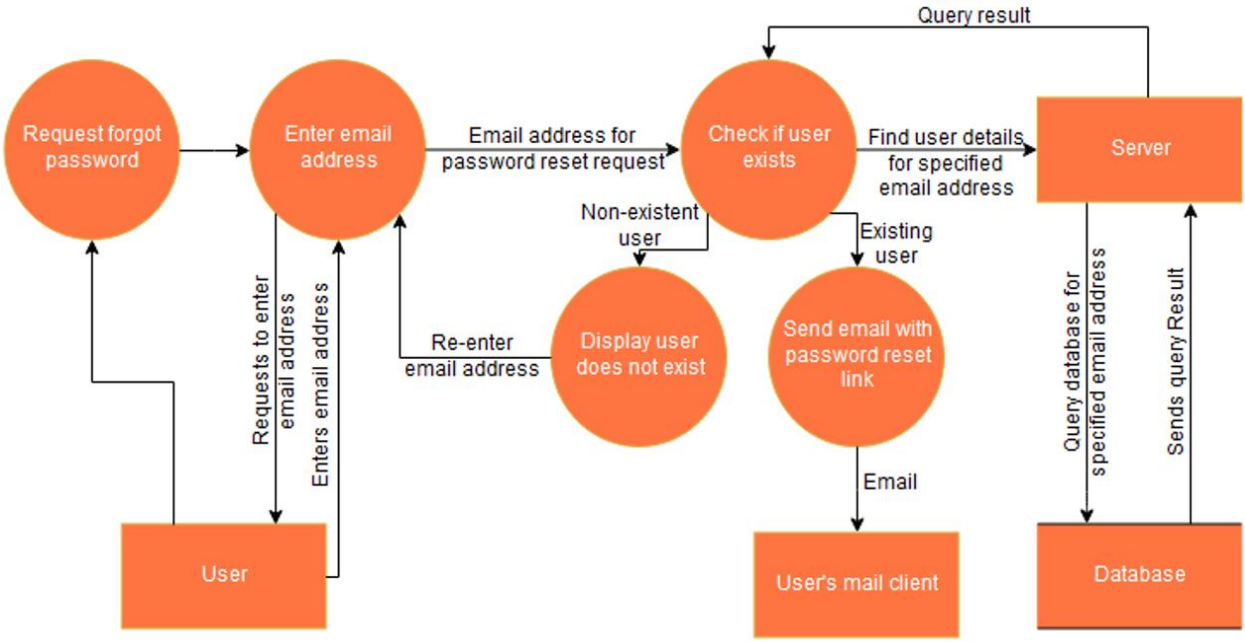

Figure 5.15 - Level 2.2 - Request for Forgot Password

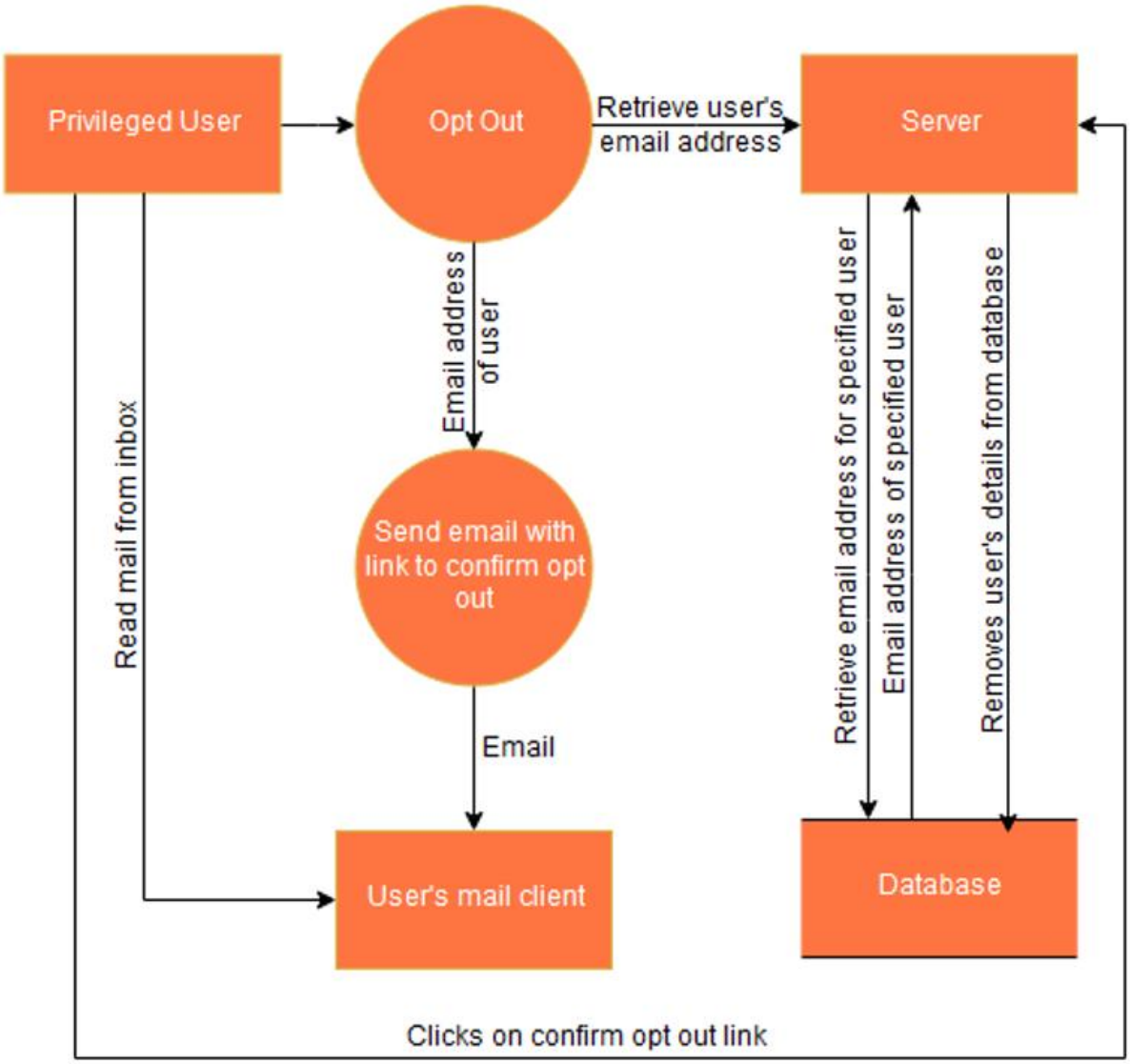

Figure 5.16 - Level 2.3 - Opt out Request by Privileged User

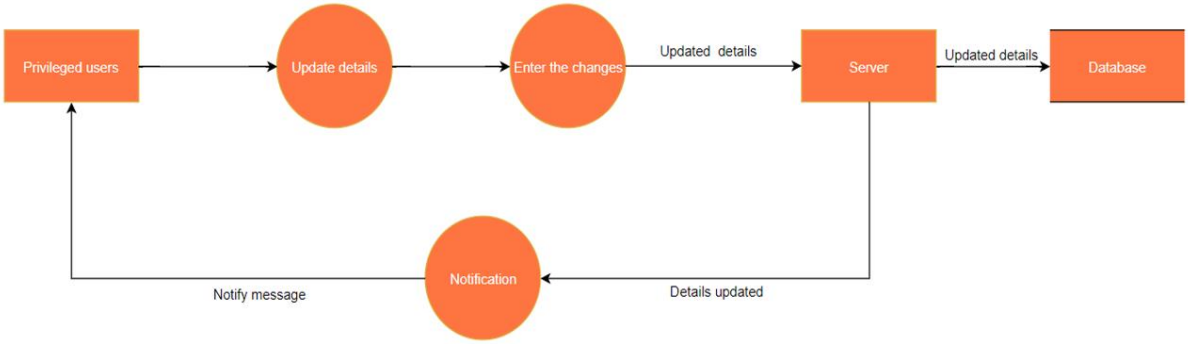

Figure 5.17 - Level 2.4 - Update Details Of Privileged User

5.3 Data Flow Diagrams

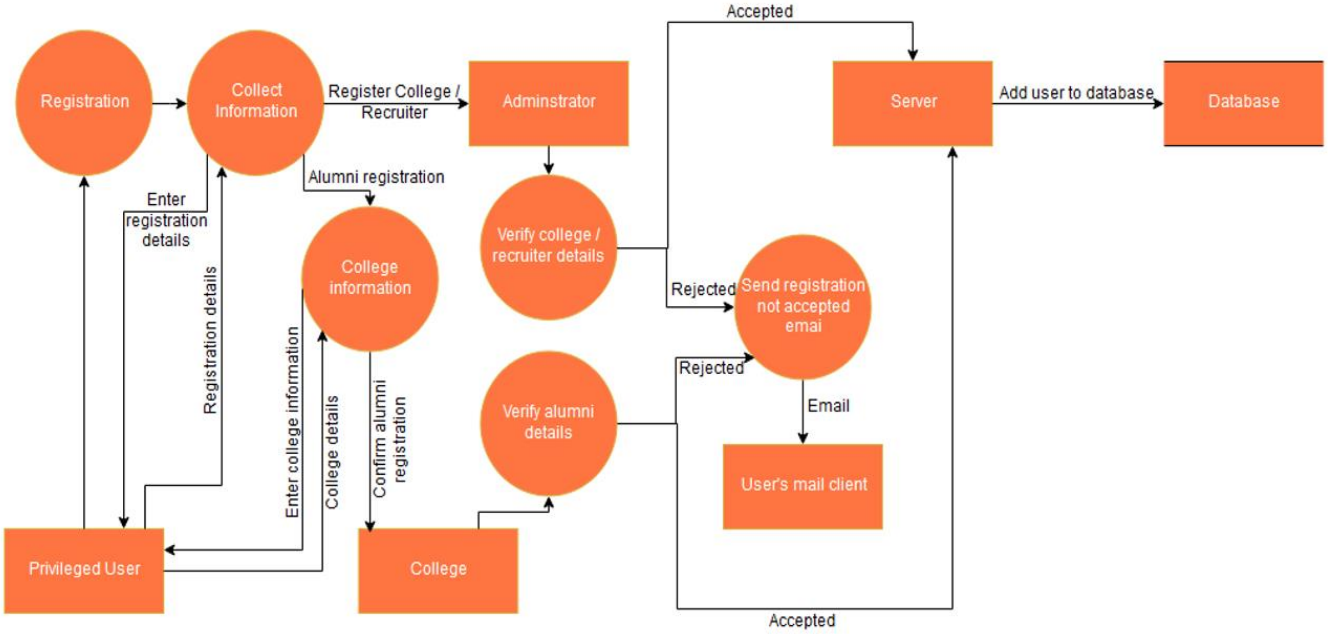

Figure 5.18 - Level 2.5 - Registration of Privileged User

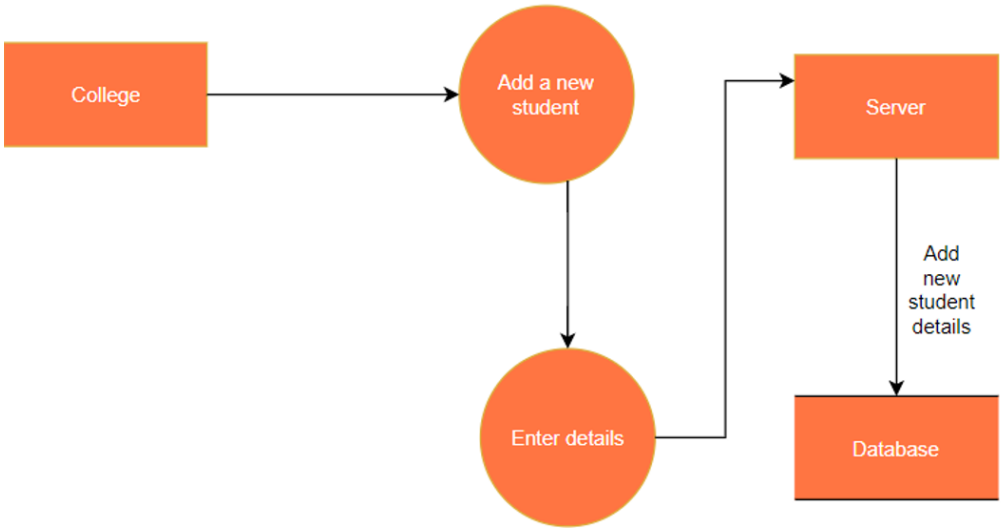

Figure 5.19 - Level 2.6 - Add New Student by College

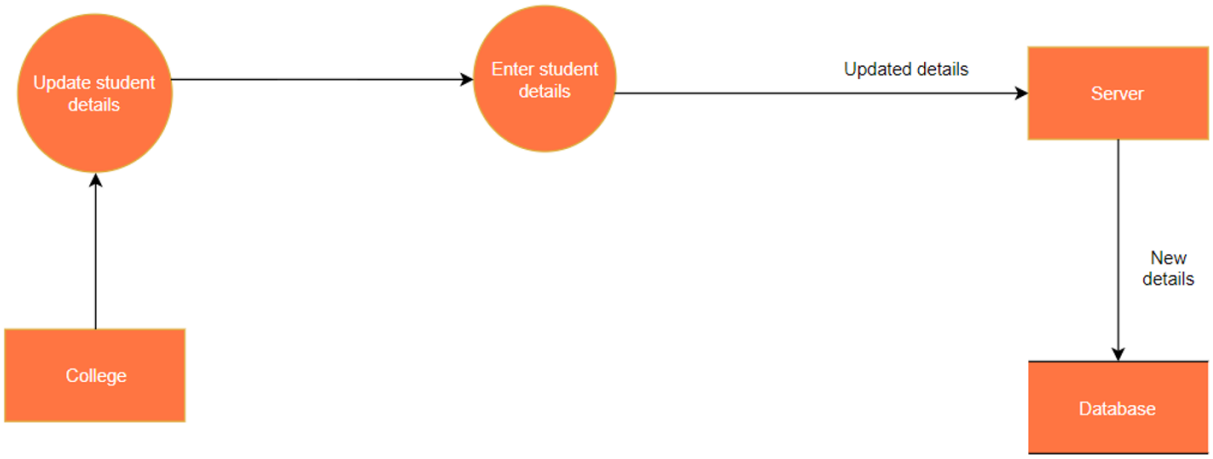

Figure 5.20 - Level 2.7 - Update Student Details

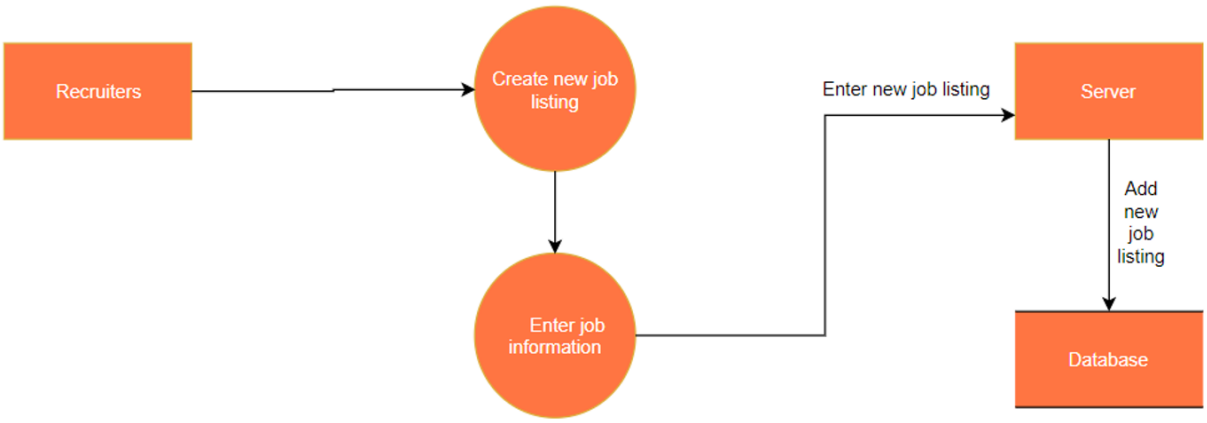

Figure 5.21 - Level 2.8 - Create New Job Listing by Recruiters

5.3 Data Flow Diagrams

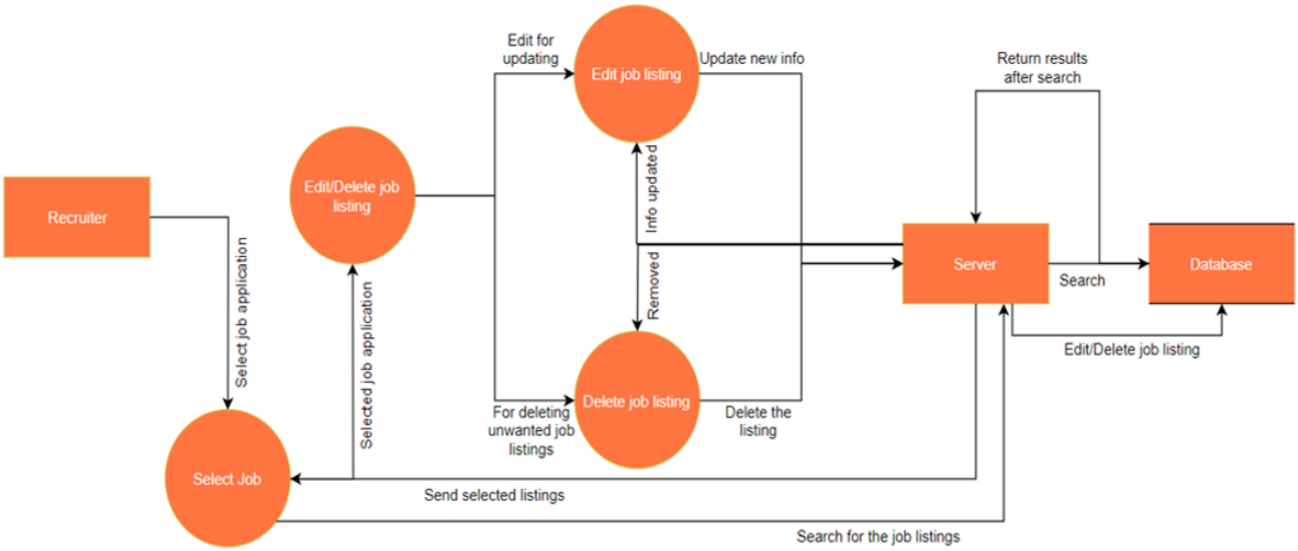

Figure 5.22 - Level 2.9 - Edit or Delete Job listing by Recruiters

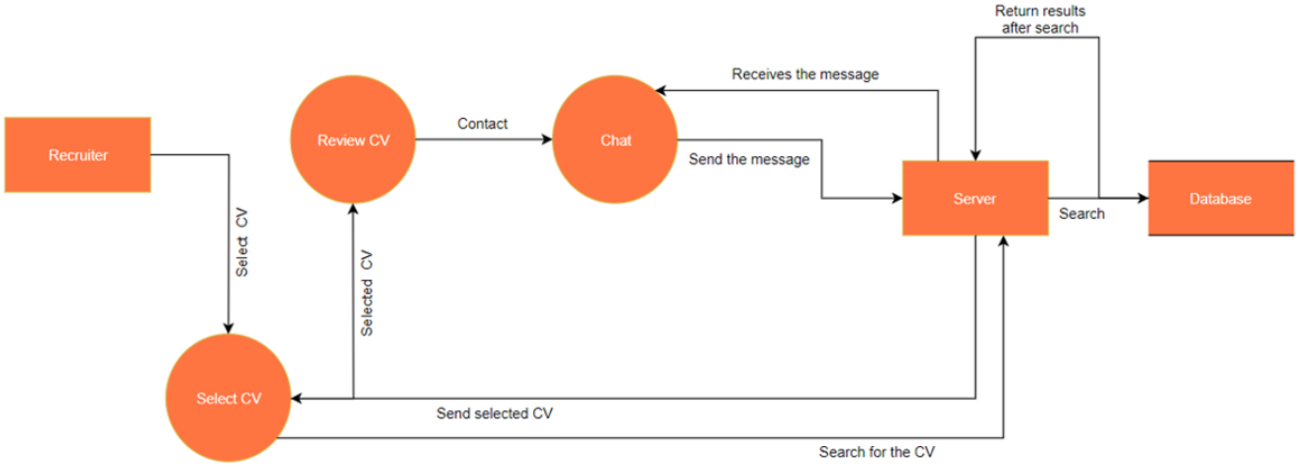

Figure 5.23 - Level 2.10 - Review CV by Recruiter

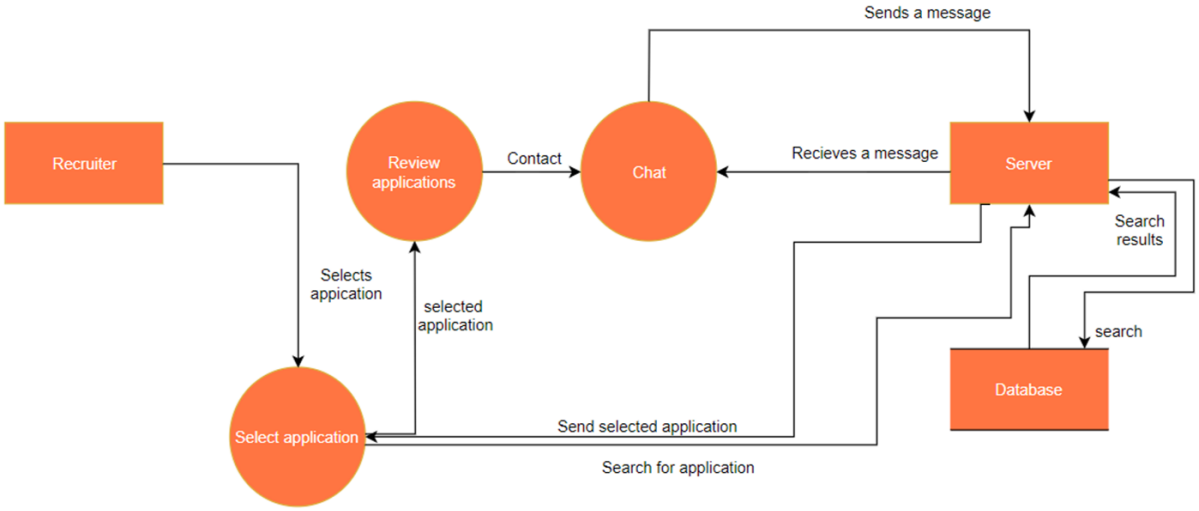

Figure 5.24 - Level 2.11 - Review Application by Recruiter

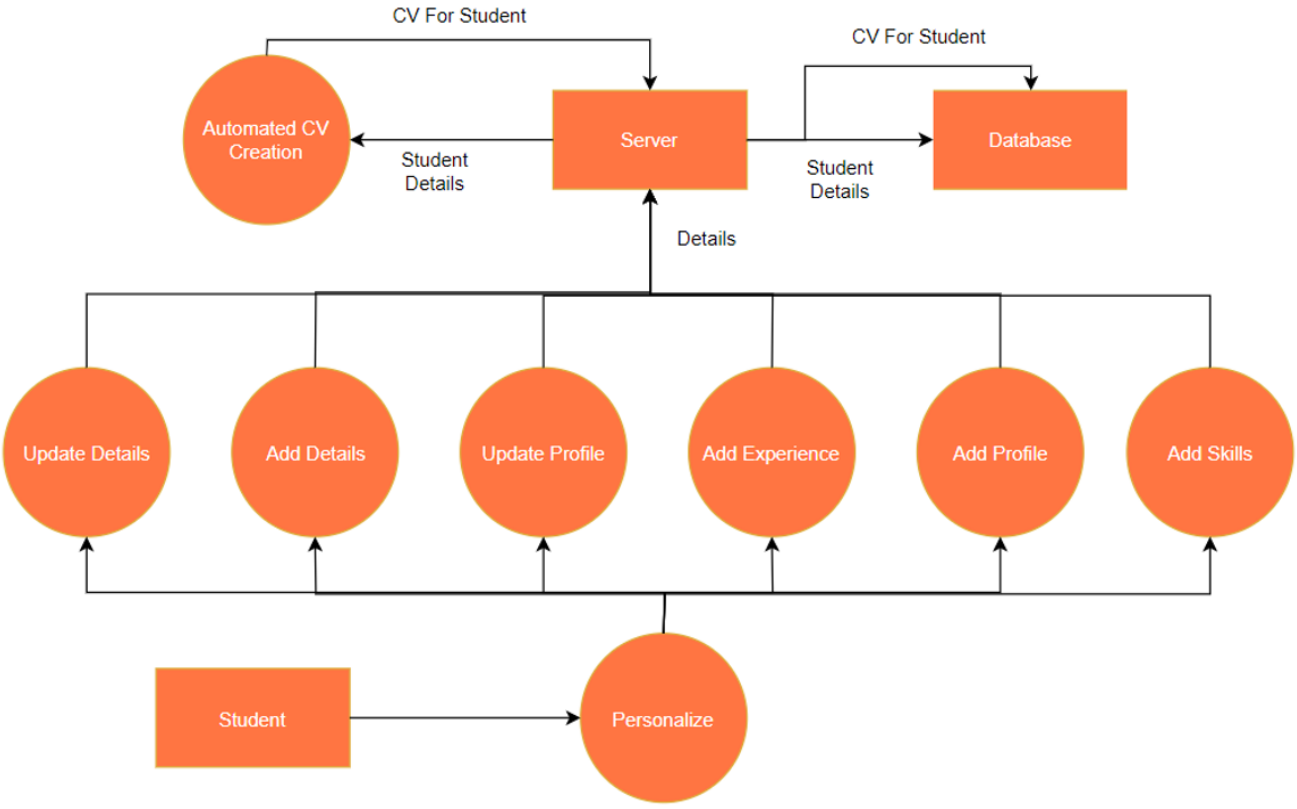

Figure 5.25 - Level 2.12 - Entering Personalisation Details by Student

5.3 Data Flow Diagrams

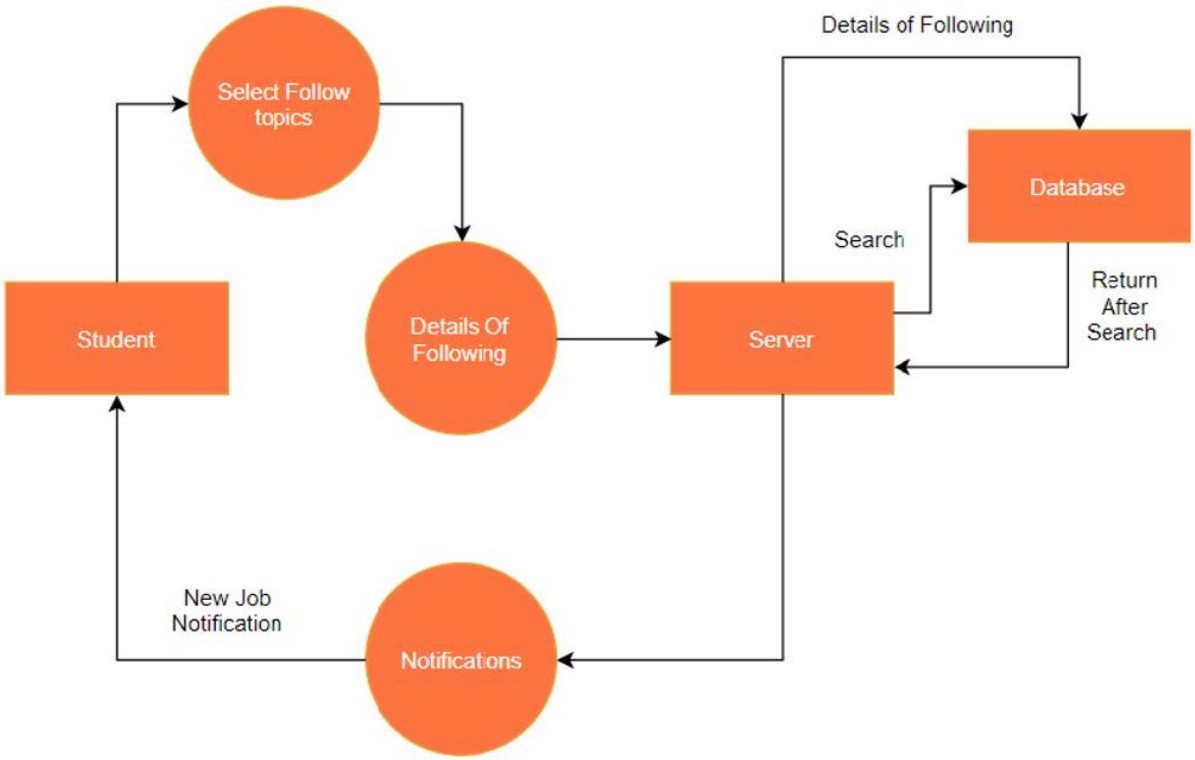

Figure 5.26 - Level 2.13 - Following Topics or Recruiters by Student

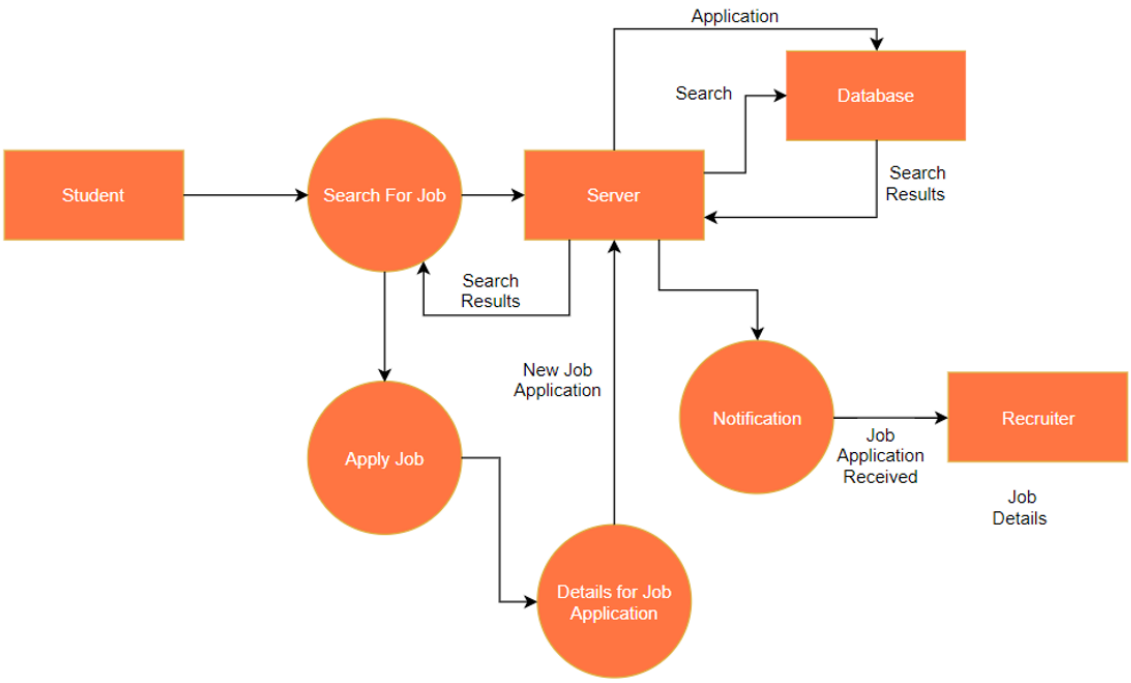

Figure 5.27 - Level 2.14 - Apply for Job by Student

5.4 Data Dictionary

A Data Dictionary is a collection of names, definitions, and attributes about data elements that are being used or captured in a database, information system, or part of a research project. It describes the meanings and purposes of data elements within the context of a project, and provides guidance on interpretation, accepted meanings and representation [11].

Data dictionary for the proposed system are represented below as tables.

5.4.1 Collections

Table 5-2 - List of Collections

COLLECTION NAME	DESCRIPTION
college	To store details about colleges
recruiter	To store details about recruiters
admin	To manage user requests

5.4 Data Dictionary

student	To store student details and manage it
alumnus	To store details of alumni of colleges
job	To store job listings and job applications
chatChannel	Channels for chat between two users
message	To store chat messages

Table 5-3 - College Collection

VARIABLE NAME	DATA TYPE	DESCRIPTION
username	String	Username of the user
password	String	Password for login
name	String	Name of the college
mobile	String	Mobile number of the user
email	String	Login email for the user
status	String	Status of approval
role	String	Role of the user
profile_pic	String	Filename for profile picture of the user
desc	String	About of the user
bg_img	String	File name for background image of the user
approval_count	int	Approval count for the user
affil_univ	String	Affiliated university of the college
contact	ContactInfo	College contact info
students	List<String>	List of usernames of usernames of all students
alumni_pending	List<String>	List of usernames of all alumni requests pending

alumni_rejected	List<String>	List of usernames of all alumni requests rejected
alumni	List<String>	List of usernames of all alumni approved
recc_req_recvd	List<Recommendation>	List of recommendation requests received
recommended	List<Recommendation>	List of recommendation requests approved
recc_rejected	List<Recommendation>	List of recommendation requests rejected

Table 5-4 - Recruiter Collection

VARIABLE NAME	DATA TYPE	DESCRIPTION
username	String	Username of the user
password	String	Password for login
name	String	Name of the recruiter
mobile	String	Mobile number of the user
email	String	Login email for the user
status	String	Status of approval
role	String	Role of the user
profile_pic	String	Filename for profile picture of the user
desc	String	About of the user
bg_img	String	File name for background image of the user
approval_count	int	Approval count for the user
license_no	String	License number of the recruiter
contact	ContactInfo	Recruiter contact info

5.4 Data Dictionary

jobs	List<String>	List of job ids of jobs created by the recruiter
recc_req_recvd	List<Recommendation>	List of recommendation requests received
recommended	List<Recommendation>	List of recommendation requests approved
recc_rejected	List<Recommendation>	List of recommendation requests rejected

Table 5-5 - Admin Collection

VARIABLE NAME	DATA TYPE	DESCRIPTION
username	String	Username of the user
password	String	Password for login
name	String	Name of the recruiter
mobile	String	Mobile number of the user
email	String	Login email for the user
status	String	Status of approval
role	String	Role of the user
address	Address	Address of the user
college_request_pending	List<String>	Pending college registration requests
college_approved	List<String>	Approved college registrations
college_denied	List<String>	Denied college registrations
recruiter_request_pending	List<String>	Pending recruiter registration requests
recruiter_approved	List<String>	Approved recruiter registrations
recruiter_denied	List<String>	Denied recruiter registration requests

Table 5-6 - Student Collection

VARIABLE NAME	DATA TYPE	DESCRIPTION
username	String	Username of the user
password	String	Password for login
name	String	Name of the student
mobile	String	Mobile number of the user
email	String	Login email for the user
status	String	Status of approval
role	String	Role of the user
profile_pic	String	Filename for profile picture of the user
desc	String	About of the user
bg_img	String	File name for background image of the user
approval_count	int	Approval count for the user
address	Address	Address of the user
reg_no	String	Register number of the user
course	String	Name of the course
branch	String	Name of the course branch
st_date	LocalDate	Starting date for the course
end_date	LocalDate	Ending date for the course
sgpa	float	SGPA of the student
cgpa	float	CGPA of the student
college_id	String	Username of the college of the student
personalisation	Personalisation	Personalisation details
recommend_req	Recommendation	Recommendation requests made by the student

5.4 Data Dictionary

recommendations	Recommendation	Recommendations received
applied_jobs	List<String>	List of job ids to which the student applied
hired_jobs	List<String>	List of job ids to which the student got hired
rejected_jobs	List<String>	List of job ids to which the application of the student was rejected

Table 5-7 - Alumnus Collection

VARIABLE NAME	DATA TYPE	DESCRIPTION
username	String	Username of the user
password	String	Password for login
name	String	Name of the alumnus
mobile	String	Mobile number of the user
email	String	Login email for the user
status	String	Status of approval
role	String	Role of the user
profile_pic	String	Filename for profile picture of the user
desc	String	About of the user
bg_img	String	File name for background image of the user
approval_count	int	Approval count for the user
address	Address	Address of the user
reg_no	String	Register number of the user
course	String	Name of the course
branch	String	Name of the course branch
st_date	LocalDate	Starting date for the course

end_date	LocalDate	Ending date for the course
sgpa	float	SGPA of the alumnus
cgpa	float	CGPA of the alumnus
college_id	String	Username of the college of the student
personalisation	Personalisation	Personalisation details
recommend_req	Recommendation	Recommendation requests made by the alumnus
recommendations	Recommendation	Recommendations received
applied_jobs	List<String>	List of job ids to which the alumnus applied
hired_jobs	List<String>	List of job ids to which the alumnus got hired
rejected_jobs	List<String>	List of job ids to which the application of the alumnus was rejected
recc_req_recvd	List<Recommendation>	List of recommendation requests received
recommended	List<Recommendation>	List of recommendation requests approved
recc_rejected	List<Recommendation>	List of recommendation requests rejected

Table 5-8 - Job Collection

VARIABLE NAME	DATA TYPE	DESCRIPTION
_id	String	Job ID
recruiter_id	String	Username of the recruiter associated with this job
profile_pic	String	Display picture for the job
name	String	Job title

5.4 Data Dictionary

desc	String	Job description
location	String	Location of work
duration	String	Duration of job
stipend	float	Stipend/salary for the job
last_date	Date	Last date to apply
questions	List<String>	Questions to the applicant
applicants	List<Application>	Applications received
hired	List<Application>	Hired applications
rejected	List<Application>	Rejected applications
skills	List<String>	Skills required for the job

Table 5-9 ChatChannel Collection

VARIABLE NAME	DATA TYPE	DESCRIPTION
id	String	Chat channel id
user1	String	Username of first participant
nameUser1	String	Name of first participant
user2	String	Username of second participant
nameUser2	String	Name of second participant

Table 5-10 - Message Collection

VARIABLE NAME	DATA TYPE	DESCRIPTION
id	String	Chat channel id
message	String	Chat message
sender	String	Username of sender
receiver	String	Username of receiver
channelid	String	Channel id for chat

timestamp	String	Timestamp of message sending

5.4.2 Embedded Documents

Table 5-11 - List of Embedded Documents

COLLECTION NAME	DESCRIPTION
Address	To store address information
Application	To store job applications
Awards	For awards received by a student or alumnus
Communities	For communities associated with a student or alumnus
ContactInfo	Contact info wrapper for college and recruiter
Education	For education details of student or alumnus
Personalisation	For personalising a student or alumnus account
Project	For projects undertaken by a student or alumnus
Recommendation	To store and manage recommendations
WorkExperience	For work experience of a student or alumnus

Table 5-12 - Address Document

VARIABLE NAME	DATA TYPE	DESCRIPTION
address_line1	String	Address line 1
address_line2	String	Address line 2
city_or_town	String	City or town
district	String	District
state	String	State
country	String	Country
pincode	Long	Pin code

5.4 Data Dictionary

Table 5-13 - Application Document

VARIABLE NAME	DATA TYPE	DESCRIPTION
applicant_id	String	Username of applicant
timestamp	Date	Timestamp of job applied
answers	List<String>	Answers to questions in the job application

Table 5-14 Awards Document

VARIABLE NAME	DATA TYPE	DESCRIPTION
name	String	Name of the award
year	int	Year of receiving the award
website_of_org	String	Website of award organizers

Table 5-15 - Communities Document

VARIABLE NAME	DATA TYPE	DESCRIPTION
name	String	Name of the community
website	String	Website of the community

Table 5-16 - ContactInfo Document

VARIABLE NAME	DATA TYPE	DESCRIPTION
website	String	Website
landph	String	Landphone
public_email	String	Public email address
address_line1	String	Address line 1

address_line2	String	Address line 2
city_or_town	String	City or town
district	String	District
state	String	State
country	String	Country
pincode	Long	Pincode

Table 5-17 - Education Document

VARIABLE NAME	DATA TYPE	DESCRIPTION
course	String	Name of course
institute	String	Name of institute
board	String	Board of education
marks	String	Marks received
year	int	Year of completion

Table 5-18 - Personalisation Document

VARIABLE NAME	DATA TYPE	DESCRIPTION
education	List<Education>	List of educational qualifications
work	List<WorkExperience>	List of work experience
skills	List<String>	List of skills
project	List<Project>	List of projects
communities	List<Communities>	List of communities
awards	List<Awards>	List of awards

5.4 Data Dictionary

Table 5-19 - Project Document

VARIABLE NAME	DATA TYPE	DESCRIPTION
name	String	Name of project
desc	String	Description of project
tech	List<String>	List of technologies used

Table 5-20 - Recommendation Document

VARIABLE NAME	DATA TYPE	DESCRIPTION
requester_id	String	Username of requester
requester_name	String	Name of requester
recommender_id	String	Username of recommender
recommender_name	String	Name of recommender
status	int	Status of recommendation
recc_msg	String	Recommendation message

Table 5-21 - WorkExperience Document

VARIABLE NAME	DATA TYPE	DESCRIPTION
job	String	Name of job
company	String	Name of company
skills	List<String>	Skills required for the job
desc	String	Description of job
st_date	Date	Starting date of job
end_date	Date	Ending date (Optional)
status	String	Status of job (current or former)

Chapter 6
Implementation

In this chapter we provide a brief explanation of the package structure used to develop the web application.

6.1 Module Description

The entire application can be packed into a single Java module and no additional configurations are made so as to specify the module hierarchy. This approach is suitable since there would be no standalone parts to this web application. However, appropriate package hierarchy is chosen within this application as module component, to distinguish various logical and functional elements of the application. The package hierarchy chosen is as follows:

- **JAVA PACKAGES**
  - **com.cirp.app**
    - **config**
      - **security**
    - **controllers**
    - **model**
    - **repository**
    - **service**
      - **security**

6.1 Module Description

- **RESOURCE PACKAGES**
  - **static**
    - **css**
      - **assets**
    - **js**
  - **templates**
    - **admin**
    - **college**
    - **common**
    - **recruiter**
    - **student**
    - **view**

6.1.1 Java Packages

com.cirp.app is the base package for this web application. A brief description of all its sub-packages are given below.

1. **config**

   The config package is used to define various configuration classes for this application. It includes classes for web MVC configuration, web socket configuration and a sub-package called **security** which defines classes for implementing web security and custom authentication success handling.
2. **controllers**

   The controllers package defines classes for the various REST API controllers used by the web application.
3. **model**

   The model package defines classes for the various data models used by the application.
4. **repository**

   The repository package defines classes and interfaces to implement data repositories which perform operations for various data manipulation on the defined data models.
5. **service**

The service package defines classes to implement various business/logical operations necessary for the controllers or/and repositories.

6.1.2 Resource Package

Resource package includes the HTML pages, CSS files, assets and Javascript files used to render an interactive user interface to the user. It also includes the application.properties file in its root directory which holds the environment variables and properties for the application. Inside the root folder, there are two sub folders as follows:

1. **static**

   The static folder holds the static components like CSS and Javascript files placed under the css and js subdirectories respectively. For convenience of access, static assets (images) are placed under the css directory in a separate subdirectory called assets.

2. **templates**

   Templates folder under resources is the default directory used by Thymeleaf to lookup HTML pages. While views with public access are placed in the root directory, access-restricted views are placed accordingly under sub-directories. The templates folder has the following sub-directories:

   1. **admin**

      Contains views (HTML pages) for users with role as ADMIN.

   2. **college**

      Contains views for users with role as COLLEGE.

   3. **recruiter**

      Contains views for users with role as RECRUITER.

   4. **student**

      Contains views for users with role as STUDENT or ALUMNUS.

   5. **common**

      Contains few views which are common to one or more types of users.

   6. **view**

      Contains public profile views of all users and jobs, accessible to all users.

6.2 Sample Codes

- **Database connection:**

```
spring.data.mongodb.uri =
mongodb+srv://<username>:<password>@cluster0.yf8sm.mongod
b.net/cirp?retryWrites=true&w=majority
```

- **Session control:**

```
server.servlet.session.timeout = 20m
server.servlet.session.cookie.http-only = true
server.servlet.session.cookie.secure = true
server.error.whitelabel.enabled = false
```

- **View resolution:**

```
public void addViewControllers(ViewControllerRegistry
registry) {
      registry.addViewController("/home").setViewName("ind
      ex");
      registry.addViewController("/").setViewName("index")
      ;
      registry.addViewController("/index").setViewName("in
      dex");
      registry.addViewController("/register").setViewName(
      "register");
      registry.addViewController("/registercollege")
      .setViewName("register_college");
      registry.addViewController("/registerrecruiter")
      .setViewName("register_recruiter");
      registry.addViewController("/registeralumnus")
      .setViewName("register_alumnus");
      registry.addViewController("/login").setViewName("lo
      gin");
      registry.addViewController("/reset-passwordrequest")
      .setViewName("reset_password_request");
      registry.addViewController("/terms-and-conditions")
      .setViewName("terms_and_conditions");
      registry.addViewController("/error").setViewName("er
      ror");
      registry.addViewController("/pending-approval")
      .setViewName("common/registration_pending");
      registry.addViewController("/update-password")
      .setViewName("common/reset_password");
      registry.addViewController("/info/csit")
      .setViewName("cs_it_info");
      registry.addViewController("/info/ec").setViewName("
      ec");
```

```
        registry.addViewController("/info/eee").setViewName(
        "eee");
    }
```

- **Email:**

```
public class SendEmail {
    @Autowired
    JavaMailSender send_email;
    public void sendEmail(String to, String sub, String
    content) {
        try {
            MimeMessage msg =
            send_email.createMimeMessage();
            msg.setRecipient(Message.RecipientType.TO,
            new InternetAddress(to));
            msg.setSubject(sub);
            msg.setContent(content, "text/html");
            send_email.send(msg);
        }
        catch (MessagingException e) {
            e.printStackTrace();
        }
    }
}
```

- **Security:**

```
protected void configure(HttpSecurity http) throws
Exception {
    http.authorizeRequests().antMatchers("/", "/index",
    "/home",        "/register", "/register-recruiter",
    "/register-college", "/register- alumnus","/terms-
    and-conditions","/error", "/login" ,"/reset-
    password-request", "/update-password",
    "/info/**").permitAll()
    .antMatchers("/admin/**").hasRole("ADMIN")
    .antMatchers("/college/**").hasRole("COLLEGE")
    .antMatchers("/recruiter/**").hasRole("RECRUITER")
    .antMatchers("/student/**").hasAnyRole("STUDENT",
    "ALUMNUS")
    .antMatchers("/alumnus/**").hasRole("ALUMNUS")
    .antMatchers("/pending-approval").hasRole("PENDING")
    .antMatchers("/chat/**").hasAnyRole("RECRUITER","STU
    DENT","ALUMNUS")
    .antMatchers("/common/**").hasAnyRole("ADMIN",
    "COLLEGE", "STUDENT", "RECRUITER", "ALUMNUS")
    .anyRequest().authenticated().and().formLogin()
    .successHandler(customizeAuthenticationSuccessHandle
    r)
```

```
            .loginPage("/login").failureUrl("/login?error=true")
            .and()
            .httpBasic();
      }
```

- **Service:**

```
      public void resetPassword(String username, String
      new_password) {
            Class<?> user_class = find.findClass(username);
            repo.updatePassword(username, passwordEncoder
                        .encode(new_password), user_class);
            repo.setToken("", username, user_class);
      }


      public void deleteRejectedRegistrations() {
            Calendar calender = Calendar.getInstance();
            calender.setTime(new Date());
            calender.add(Calendar.DATE, -14);
            mongoTemplate.findAllAndRemove(
                  new Query(where("status_changed")
                        .is(calender.getTime())), User.class);
      }
```

- **Repository:**

```
      public  void  updateContact(ContactInfo  contact,  String
      username, Class<?> user_class) {
            Query query = new Query();
            query.addCriteria(where("username").is(username));
            mongoTemplate.updateFirst(query,
                  new Update().set("contact", contact),
                  user_class);
      }
```

- **Controller:**

```
      @PostMapping(value="/accept-user")
      public String acceptRegistration(
        @RequestParam String username,
        Authentication authentication,
        HttpServletRequest request,
        RedirectAttributes redirectAttributes) {
            acceptReject.acceptRejectRegistration(username,
            "confirm", authentication.getName());
            if(request.isUserInRole("ROLE_ADMIN"))
                  return "redirect:/admin/admin-panel";
            else if(request.isUserInRole("ROLE_COLLEGE"))
                  return "redirect:/college/admin-panel";
```

```
            else
                  return "/error";
      }
```

Chapter 7
Testing

Software testing is a process, to evaluate the functionality of a software application with an intent to find whether the developed software met the specified requirements or not and to identify the defects to ensure that the product is defect-free in order to produce the quality product.

7.1 Testing Explained

Junit tests were conducted for each of modules and packages in the application. Unit test are used to test individual code components and ensure that code works the way it was intended to. Junit test is a framework which uses annotation to identify methods that specify a test. By default, Junit tests generate simple report XML files for its test execution. These XML files can then be used to generate any custom reports as per the testing requirement.

Junit tests ensure that every single part or functioning in your software is tested even before module or system level testing is performed. Every time we make a small or big modification in the code, we can make sure that the function is performing well and has not broken any older functionality by executing all Junit test in one go written for that function. Junit testing is one of the best testing frameworks that can be selected for an efficient testing process.

7.2 Sample Test Cases

Table 7-1 - Sample Test Case 1

Sample Test Case 1	
Test case id: Register student	Test Designed By: Jincy P Janardhanan
Test priority (Low/Medium/High): High	Test Designed Date:10/10/2020
Module Name: College	Test Executed By: Alka Bhagavaldas K
Test title: New student registration	Test Execution Date:10/10/2020
Description:	Register a new student

Pre-Conditions: College registered and approved
Dependencies: none

Step	Test Steps	Test Data	Expected Result	Actual Result	Status	Notes
1.	Input data	username: raghav name:Raghav email: rg@gmail.com password: Rg1234 mobile:967836778	Student successfully registered	Student successfully registered	Pass	

Post conditions: Student have successfully registered.

Table 7-2 - Sample Test Case 2

Sample Test Case 2	
Test case id : Apply job	Test Designed By: Alka Bhagavaldas
Test priority (Low/Medium/High):High	Test Designed Date:12/10/2020
Module Name: Student	Test Executed By: Jincy P Janardhanan
Test title: Student applies for job	Test Execution Date:12/10/2020
Description:	Apply for a job

Pre-Conditions: Student registered
Dependencies: none

Step	Test Steps	Test Data	Expected Result	Actual Result	Status	Notes

1.	Input data	username: raghav job_id: 48258312545 recruiter_id: recruiter	Student applied for job	Student applied for job	Pass

Post conditions: Student applied for job.

Table 7-3 - Sample Test Case 3

Sample Test Case 3	
Test case id: Login user	Test Designed By: Aleena Sunny
Test priority (Low/Medium/High): High	Test Designed Date:10/10/2020
Module Name: User	Test Executed By: Ameena Shirin
Test title: User login	Test Execution Date:10/10/2020
Description:	Test login functionality

Pre-Conditions: User registered

Dependencies: none

Step	Test Steps	Test Data	Expected Result	Actual Result	Status	Notes
1.	Login with username	username: raghav password: Rg1234	User login successful	User login successful	Pass	
2.	Login with email	email: rg@gmail.com password: Rg1234	User login successful	User login successful	Pass	

Post conditions: User login successful.

Table 7-4 - Sample Test Case 4

Sample Test Case 4	
Test case id: Create job	Test Designed By: Ameena Shirin
Test priority (Low/Medium/High): High	Test Designed Date:12/10/2020
Module Name: Recruiter	Test Executed By: Aleena Sunny
Test title: Add new job listings.	Test Execution Date:12/10/2020
Description:	Create a new job

Pre-Conditions: Recruiter registered and approved

Dependencies: none

Step	Test Steps	Test Data	Expected Result	Actual Result	Status	Notes
1.	Input data	username: recruiter job_title: Manager location: Mumbai duration:1 year stipend: 40000	Job successfully created	Job successfully created	Pass	

Post conditions: Job successfully created.

Chapter 8
Maintenance

Software Maintenance is the process of modifying a software product after it has been delivered to the customer. The main purpose of software maintenance is to modify and update software application after delivery to correct faults and to improve performance. A common perception of maintenance is that it merely involves fixing defects. However, one study indicated that over 80% of maintenance effort is used for non-corrective actions. This perception is perpetuated by users submitting problem reports that in reality are functionality enhancements to the system. Software Maintenance must be performed in order to:

- Correct faults.
- Improve the design.
- Implement enhancements.
- Interface with other systems.
- Accommodate programs so that different hardware, software, system features, and telecommunications facilities can be used.
- Migrate legacy software.
- Retire software.

8.1 Types of Maintenance

We follow different types of maintenance of the web application with the help of GitHub issue tracking of our repository:

1. **Bug fixing:**

   Corrective maintenance of a software product may be essential either to rectify some bugs observed while the system is in use, or to enhance the performance of the system. A bug fix is also known as a program temporary fix.

   If a user suspects a potential bug in the system, it may be reported as an issue with the bug tag associated with it.

2. **Security enhancement:**

   When a user suspects that the system has security loopholes, he/she may report it as an issue with security tag and optionally assign a priority tag like high, low, or medium.

3. **Feature enhancement:**

   In case of outdated features, we can update the features of the application according to user preferences. This may require restructuring of the software code and the database used in the software and can take time to develop and deploy.

   Such issues may be reported with enhancement and feature request tags.

4. **User Interface enhancement:**

   In-house inspection of the application and user reviews for beta/regular version releases can help us in deciding and implementing UI enhancements. When necessary, the users may also file a GitHub issue with enhancement and UI tags.

5. **Improving the software to support user requirements:**

   Any user requirements that do not fit any of the above categories may be reported with enhancement tag and any other appropriate tag that best suits the context and content of the issue. Generally, requests for enhancement of the functionality of the software, performance improvements, or customizing data processing functions as desired by the user can be considered.

Chapter 9
Conclusion

This project provided us practical knowledge about Java enterprise web application development using spring boot and spring security frameworks, and efficiently using MongoDB - a document-oriented database - for backend development. It was a great learning experience to explore the basic and advanced features of the MongoDB platform, which is different from the traditional SQL databases.

We have exploited various functionalities offered by Thymeleaf to render the webpages with ease. We learnt and made use of web sockets over STOMP protocol to implement the chat feature of our application. To develop the user interfaces, we have made use of HTML5, Javascript and CSS. Bootstrap was used occasionally in a few pages when required. Also, at one point we have made use of JQuery AJAX post methods for convenience of using the REST methods when Thymeleaf techniques where incompatible/complex to suit the application needs.

The developed application offers simple and interactive forms. Authentication and role-based authorization is used to secure the REST endpoints. Job aspirants, recruiters and colleges can take advantage of this web application as an excellent job board. The application is commercially deployable.

The application source is available at https://github.com/jincy-p-janardhanan/Career-Information-and-Recruitment-Portal and is deployed at https://career-info-n-recruitment.herokuapp.com/ .

Chapter 10
Future Perspectives

This project fulfils the primary requirements of the job-seeking collegiate students, alumni and employers. It can be extended in several ways – We can provide recommendations and email updates for new job postings based on the job seeker's search history. Since, the job seekers might be interested in building a strong resume, we can provide tips and information for the same. We can also provide templates for building a resume which might interest most applicants. The web application is developed to most suit the needs of a job seeker, it can be extended to support more desirable functionalities for an employer as well.

Through career information and recruitment portal finding a job becomes much easier. We can also provide more information about higher studies options available for students. Future development scopes for the application include job/candidate suggestions using AI, advanced application tracking etc.

Chapter 11
References

[1] "What is Feasibility Study and Its Importance in Project Management," Simplilearn, 17 July 2020. [Online]. Available: https://www.simplilearn.com/feasibility-study-article. [Accessed 23 July 2020].

[2] R. R. Young, The Requirements Engineering Handbook, Norwood, MA, USA: Artech House, 2003.

[3] D. Howe, "FOLDOC entry for "end-user"," 29 03 1997. [Online]. Available: https://foldoc.org/end-user. [Accessed 22 07 2020].

[4] "DI-IPSC-81433A, DATA ITEM DESCRIPTION: SOFTWARE REQUIREMENTS SPECIFICATION (SRS)," EverySpec, 15 December 1999. [Online]. Available: http://everyspec.com/DATA-ITEM-DESC-DIDs/DI-IPSC/DI-IPSC-81433A_3709/. [Accessed 07 22 2020].

[5] D. Garbar, "SRS document Helps to Protect IT Projects From Failure," BELITSOFT, 8 August 2016. [Online]. Available: https://belitsoft.com/php-development-services/software-requirements-specification-helps-protect-it-projects-failure. [Accessed 22 07 2020].

[6] A. Stellman and J. Greene, "Glossary," in *Applied Software Project Management*, O'Reilly Media, 2014.

[7] "Systems Design," Wikipedia, 25 March 2020. [Online]. Available: https://en.wikipedia.org/wiki/Systems_design. [Accessed 22 July 2020].

[8] "Systems Architecture," Wikipedia, 8 May 2020. [Online]. Available: https://en.m.wikipedia.org/wiki/Systems_architecture. [Accessed 23 July 2020].

[9] "User Interface Design," Wikipedia, 14 July 2020. [Online]. Available: https://en.m.wikipedia.org/wiki/User_interface_design. [Accessed 23 July 2020].

[10] "Data Flow Diagrams," javaTpoint, [Online]. Available: https://www.javatpoint.com/software-engineering-data-flow-diagrams. [Accessed 23 July 2020].

[11] "What Is a Data Dictionary?," UCMERCED Library, [Online]. Available: http://library.ucmerced.edu/data-dictionaries. [Accessed 23 July 2020].