JOB RECOMMENDATION SYSTEM WITH CANDIDATE EVALUATION

Tribhuvan University

Institute of Science and Technology

A Final Year Project Report Submission in

Partial Fulfillment of the Requirement for the Degree of

Bachelor of Science in Computer Science and Information Technology

UNDER THE SUPERVISION OF

Mr. Satya Bahadur Maharjan

Program Coordinator

Trinity International College

SUBMITTED BY

Aakanchhya Sigdel (8097/072)

Subesh Bhandari (8136/072)

Tashi Lama (8141/072)

SUBMITTED TO

Trinity International College

Department of Computer Science and Information Technology

Dillibazar Height, Kathmandu, Nepal

June, 2019

JOB RECOMMENDATION SYSTEM WITH CANDIDATE EVALUATION

Tribhuvan University

Institute of Science and Technology

A Final Year Project Report Submission in

Partial Fulfillment of the Requirement for the Degree of

Bachelor of Science in Computer Science and Information Technology

SUBMITTED BY

Aakanchhya Sigdel **(**8097/072**)**

Subesh Bhandari (8136/072**)**

Tashi Lama (8141/072**)**

SUBMITTED TO

Trinity International College

Department of Computer Science and Information Technology, Dillibazar Height, Kathmandu, Nepal

June 2019

DECLARATION

Project entitled "**JOB RECOMMENDATION SYSTEM WITH CANDIDATE EVALUATION**" is being submitted to the Department of Computer Science and Information Technology, Dillibazar, Kathmandu, Nepal for the fulfillment of the seventh semester under the supervision of **Mr. Satya Bahadur Maharjan**. This project is original and has not been submitted earlier in part or full in this or any other form to any university or institute, here or elsewhere, for the award of any degree.

Aakanchhya Sigdel (8097/072)
Subesh Bhandari (8136/072)
Tashi Lama (8141/072)

RECOMMENDATION

This is to recommend that **Aakanchhya Sigdel**, **Subesh Bhandari** and **Tashi Lama** have carried out research entitled **"JOB RECOMMENDATION SYSTEM WITH CANDIDATE EVALUATION"** for the fulfillment of seventh semester project in B.Sc. in Computer Science and Information Technology under my supervision. To our knowledge, this work has not been submitted for any other degree.

They have fulfilled all the requirements laid down by the Trinity International College Department of Computer Science and Information Technology, Dillibazar, Kathmandu.

Mr. Satya Bahadur Maharjan
Program Coordinator
Department of Computer Science and Information Technology
Trinity International College
Dillibazar, Kathmandu, Nepal

June 2019

TRINITY
INTERNATIONAL
COLLEGE

Dillibazar Height, PO Box: 26111, Kathmandu, Nepal
Tel: +977 1 4445955/4445956, Fax: 4437867
Email: info@trinitycollege.edu.np
www.trinitycollege.edu.np

LETTER OF APPROVAL

Date: 21/09/2019

On the recommendation of **Mr. Satya Bahadur Maharjan**, this Project Report submitted by **Aakanchhya Sigdel**, **Subesh Bhandari** and **Tashi Lama** entitled "**JOB RECOMMENDATION SYSTEM WITH CANDIDATE EVALUATION"** in partial fulfillment of the requirement for the award of the bachelor's degree in Computer Science and Informational Technology is a bonafide record of the work carried out under my/our guidance and supervision at Trinity International College, Kathmandu.

EVALUATION COMMITTEE

Mr. Satya Bahadur Maharjan	Mr. Satya Bahadur Maharjan
Head of Department /	Head of Department /
Program Coordinator,	Project Supervisor,
Trinity International College	Trinity International College
Dillibazar Height,	Dillibazar Height,
Kathmandu, Nepal	Kathmandu, Nepal

External

Date: ______________

ACKNOWLEDGEMENT

We would like to take this opportunity to acknowledge and appreciate all of those who helped us with the project in every possible ways. We would like to express our sincerest gratitude to our final year project supervisor **Mr. Satya Bahadur Maharjan**, Program Coordinator and Supervisor for the tremendous support, critical analysis and crucial suggestions throughout the project development process.

We would also like to thank our Assistance Coordinator Mr. **Abhishek Dewan** and entire member of IT department of Trinity College including **Mr. Shoyam Bhattrai, Mr. Jitesh Tuladhar** who encouraged us to be consistent throughout the project.

Further, we would also like to present our deepest thanks to the staffs of **Trinity International College** who facilitated us with this curriculum to test our potential and break boundaries with the project.

Aakanchhya Sigdel (8097/072)
Subesh Bhandari (8136/072)
Tashi Lama (8141/072)

ABSTRACT

Job Recommendation System with Candidate Evaluation is a web based application that helps job seekers to find jobs of their interest and also helps job providers to find deserving candidates for their organization. The system was developed using agile kanban development strategy. The recommendation engine is built using hybrid filtering technique i.e. both content-based filtering and collaborative filtering. The filtering techniques take job seeker's explicit as well as implicit information in order to recommend jobs with that of similar interest. The system is also implemented with candidate evaluation system that enables job providers to get assisted on filtering candidates by organizing and examining skill tests.

Keywords: *Recommendation System, Candidate Evaluation System, Pearson Correlation, Job Seeker, Job Provider, Skill test*

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

API	Application Programming Interface
CV	Curriculum Vitae
CCRS	Classification Candidate Reciprocal System
SRL	Statistical Relational Learning
REST	Representational State Transfer
JSON	JavaScript Object Notation
OS	Operating System
SQL	Structured Query Language
DB	Database
CRUD	Create Update Delete

LIST OF SYMBOLS

x	Variable input
y	Variable input
$\bar{x}$	Mean of x
$\bar{y}$	Mean of y
Corr(x,y)	Correlation coefficient of x and y
$\sum$	Summation

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Introduction

The Job Recommendation System with Candidate Evaluation is a platform for both job seekers and job providers to connect with one another. The system is not just confined to the job recommendation feature but is also accompanied with candidate evaluation feature which assists job providers in recruiting deserving candidate for the post.

The workflow of the system is quite different compared to the traditional job recommendation systems. The traditional recommendation system takes job applicant's explicit information to build their profile then uses them to generate job recommendations accordingly [5]. And this may produce risks like resulting high number of dissatisfied job seekers and longer duration of unemployment [4]. To prevent such risks, our system not only makes use of job applicant's explicit information but also tracks the implicit information like job applicant's search histories which adds up in generating accurate and effective recommendations.

The system operates on a domain that easily scales up the amount of data in volume [4]. The data from resumes of job seekers and job posts from job providers are produced vigorously. And even more data are generated when there occurs interaction between both job seekers and job providers. Such condition of generating excessive volume of data can be termed information overloading. The system makes proper utilization of information overloading [5]. With the availability of enormous amounts of information, the system is capable of recommending job seekers with the jobs of their interest and field more efficiently. This further enhances the system by establishing a better connection between the job seekers and the job providers.

Basically, the system can be partitioned into two subsystems: job seeker subsystem and job provider subsystem. The job seeker subsystem allows job applicants to enter their information. The information collected is used to create respective job seeker's profile, which is one of the vital factors to consider when it comes to recommending jobs. Whereas, the job provider subsystem enables job providers to enter their information. The information obtained from each of the individual subsystems are stored, observed and compared then, jobs are recommended accordingly.

In the context of Nepal, there are numerous job portals with the recommendation engine implemented on their site. But none of them happened to have implemented a system for evaluating job candidates for the moment in the present scenario. So the system takes job seeker hiring process to a step further by enabling provision for setting up several skill tests. These skill tests can be public and private. The public skill test is the default test carried out by the system whereas the private skill test is the one carried out by the job providers within the system. The candidates are evaluated on the basis of these tests. The points that they obtain through these tests could be a major factor for them to be recruited by the job provider.

1.2 Problem Definition

Both of the users, job seekers and job providers have to face corresponding problems. Job seekers have to visit several job portals and browse through thousands of jobs in order to find the relevant ones of their interest [6]. This results in fetching and rendering of huge numbers of job posts with matching search keywords. So job seekers have to scroll and view job details of each and every job post until they finally find a job post with matching criteria. Then they apply for it with no confirmation of their placements from the recruiters. And despite following all the above actions, it is also not expected for the job seekers to get response from the recruiting company very often. And this puts the job seekers in a dilemma whether to apply for other companies as well or to just wait for the response.

Whereas, job providers have to surf through resume of hundreds and thousands of job applicants. They have to select those candidates with the exact matching criteria that the company is looking for the time being. They enlist the selected candidates for further examinations and contact each one of them individually to conduct necessary tests. After carrying out tests, they have to enlist and filter the passed candidates, then contact those candidates to further conduct countless hours of interviews.

The process is tedious, time consuming and inefficient. And it also requires a lot of manpower to carry out the process successfully.

1.3 Objectives

The objectives of the system are mentioned below:

- To recommend jobs to job seekers based on their skills, location, search histories and interests.
- To facilitate job seeker with user interfaces to search for jobs with ease.
- To enable job seekers to take skill tests set up by the system itself and/or job providers.
- To filter out candidates best fit for the job providers.
- To notify when new jobs fit for the users are available.

1.4 Scope of the project

The system will allow Job seekers to easily search for jobs they are looking for. They will be recommended jobs based on their history, preferences and similarity between other users who have applied for other jobs. They will see the overview of their activity in their dashboards. They will be able to acquire skills based on the tests that they take. With skills Job Providers can easily target Job Seekers and easily recruit them. They will be notified for each action they perform and the status of their application. For Job Providers, they will be able to post jobs, create tests and customize their job requirements. To ease out the process of searching through hundreds of CV's they will be able to evaluate Job Seekers based on their skills and their qualifications automatically. They can also set up tests, so that they can target users with the talent they require. The tests can be manual and automated according to the Job Providers preferences. They will also be able to see their reports on how the posted job is doing so far. It aims to ease out both the process of searching for jobs and hiring candidates.

1.5 Limitation of the project

Although, the system aims to address the modern issue for job recruitment and recommendation, it is only the minimal version of what can be done. It fails to track all the information required for Job seekers and Job Providers. When the job seekers not logged in to the system it fails to recommend jobs. The tests in the system are not modifiable and can only show multiple choice questions.

CHAPTER 2

RESEARCH METHODOLOGY

2.1 Literature Review

With the explosive growth in the digital information and the use of internet, the systems built are concerned to make our life more easier for day-to-day activities. Job recommendation has gained its popularity shifting from the traditional way of recruiting and applying for jobs to e-recruitment process.

Bilateral recommendation in any recommendation system means matching the two objects or entities with one another. Reciprocal recommender system recommends people-to-people according to the preference of both sides. Classification Candidate Reciprocal System (CCRS) is used to recommend job in [1].

Job recommendation can be done using 4 different algorithms, that is, content-based filtering, collaborative filtering, information-based filtering and hybrid algorithm. Statistical Relational Learning (SRL) is used to construct hybrid job recommendation system, that combines representational abilities of first order logic with the ability of probability theory to model uncertainty [3].

Job recommendation system only based on explicit information can create high risk, such as, higher number of unsatisfied job and longer unemployment duration as well. Thus, recommendation should also include the users activity and the activity or similar users to further increment the users recommendation [4].

As user of job recommendation system is frequently changing object, it faces a lot of cold-start problem. Along with changing users, the interest of the users are also changing matter, for which employ the dynamic recommendation in a job recommender system, where the user profile is dynamically updated [5].

There are various challenges that should be faced while developing a system that recommends job, such as, scalability, job sparsity and cold-start problem. To overcome the challenges, a novel item-based job recommendation system is implemented. Rather than using job profiles made on the recruitment system, the system can also make use of social network on which the user has registered himself [6].

2.2 Related Study

There are several different websites that is similar to this system and two of them are primary means of research for this system.

2.2.1 Upwork

Upwork is a freelancing company that gathers around freelancers and job providers. It has a skill earning and building scheme which are then used by job providers to hire workers. This website is used as a reference for our candidate evaluation system.

2.2.2 Mero Job

Mero job is one of the biggest websites that serves as an online Job Recruitment in Nepal. It has a wide audience and a big reach in the market. This website is used to build up structure for our system interface and backend design. It has a lot of features but it lacks some major features that this system is proposing to solve i.e. candidate evaluation system.

2.3 Flow chart of working of the model

The workflow of the system can be seen in Figure 1 below. The system has two main users i.e. Job Seekers and Job Providers. The system behaves differently according to the user currently in the system. A Job Seeker can acquire skills, search and apply to jobs. Whereas a Job Provider can create a vacancy, set up tests to evaluate the candidates and finally hire them. They can also view their reports and analytics in their dashboards. When there are no other activities in the system the users will logout and terminate the application. And start the process again until Job seekers find satisfactory jobs and Job Providers satisfactory candidates.

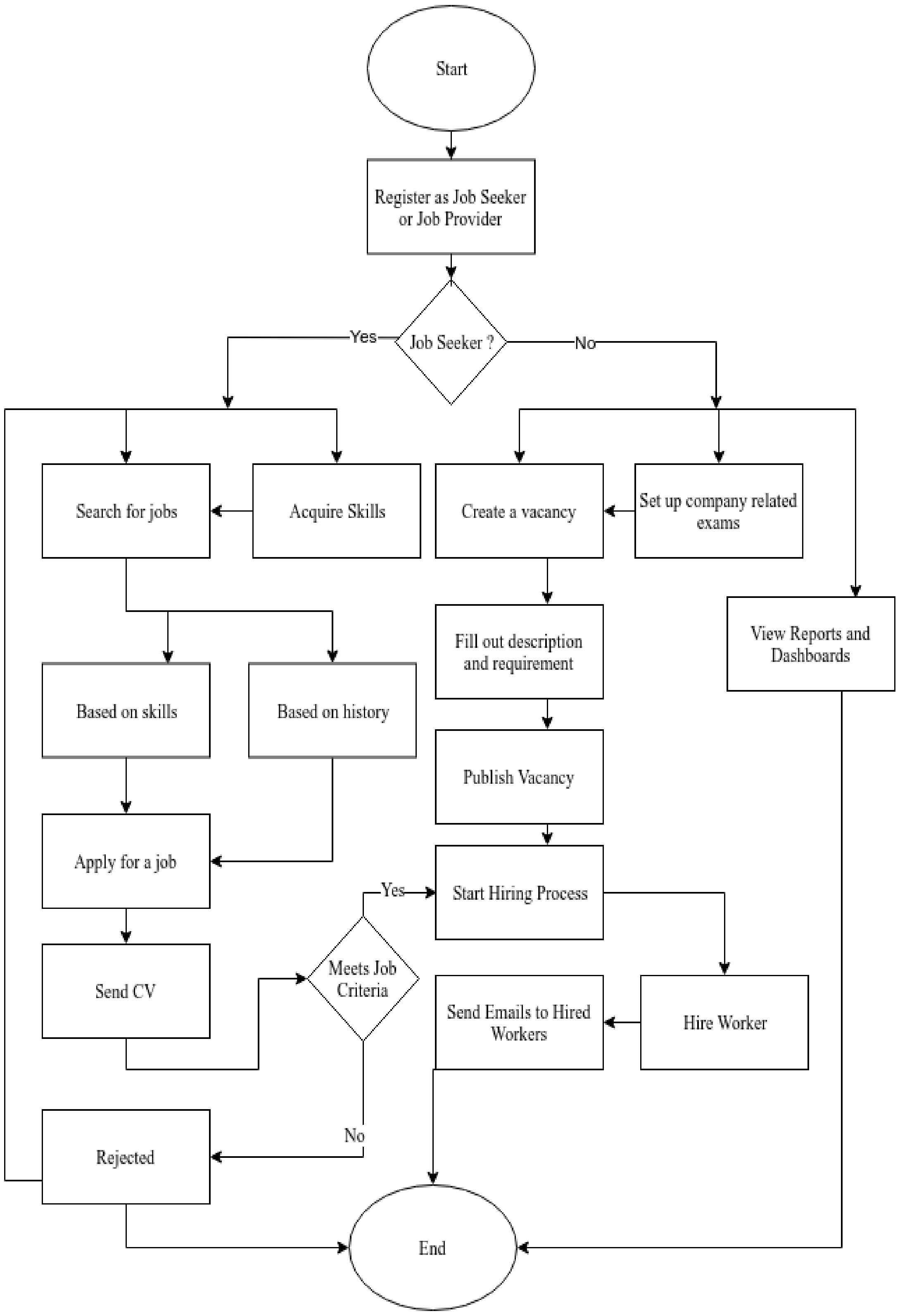


Figure 1: System workflow

2.4 Methodology

2.4.1 Recommendation System

From, the information researched and gathered collaborative filtering seems to be the appropriate technique for our use case. Recommendation based on explicit information can create high risk, such as, higher number of unsatisfied job and longer unemployment duration as well [6]. So, it is decided to use collaborative filtering to solve this issue and serve better recommendation.

2.4.1.a User-Item Collaborative Filtering

In User-Item Collaborative filtering similar users are found, using a similarity function [5]. Then recommendation is made to one another with the jobs they haven't applied or viewed. Users with similar views are grouped and based on their applications to jobs their recommendation is calculated.

2.4.1.b Item-Item Collaborative Filtering

In item based collaborative filtering, the similarity of an item is calculated with the existing item being consumed by the existing users. Then on the basis of amount of similarity, similar item is recommended to the user. In our case suppose a user A applies to job E and job F, another user B applies to job E, then job F is recommended to user A as they are highly likely to be similar.

2.4.1.c Hybrid User-Item Based Collaborative Filtering

In this approach, both the User-Item and Item-Item collaborative filtering are used to recommend items to the user. In the Job page the system will be showing similar jobs and in Job Seekers home page it will be showing jobs based on similar users.

2.4.1.d Pearson correlation

Pearson similarity is the covariance of the two n-dimensional vectors divided by the product of their standard deviations [5]. The system will be using it to calculate similarity.

$$\text{Corr}(x,y) = \sum_i (x_i-\bar{x})(y_i-\bar{y}) / \sqrt{\sum(x_i-\bar{x})^2 \sum(y_i-\bar{y})^2}$$

The pseudo code can be seen as below:

```
function recommend(user)
    for each other_user in users do
        score = pearson_similarity(user, other_user)
        if score <= 0:
            continue
        for each job in jobs(other_user)
            if job not in jobs(user)
                add_recommendation(user)
        end
    end
    return recommendations
end
```

2.4.2 Candidate Evaluation System

It is quite troublesome for Job Provider systems to filter out list of all applied candidates. The system will be using an evaluation system to filter out the candidates according to the Job Seeker's requirement. The evaluation system will consist of rules defined by Job Providers to only select proper candidates. It will also be able to conduct exams which provide skills that can be used to filter out candidates based upon their skills. The pseudo code can be seen as below:

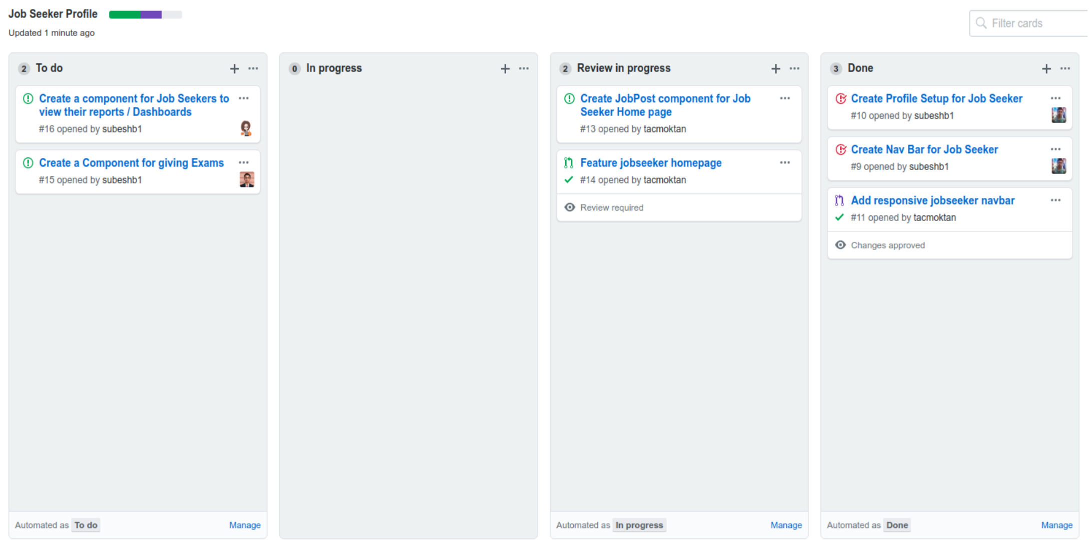
```
function evaluate(user, job)

    user_stats = fetch_user_stats(user)
    job_requirements = fetch_job_requirement(job)
    if  eligibility_check(users_stats, job_requirements)
        allow_apply(user)
    end
    else
        reject_apply(user)
    end
end
```

CHAPTER 3

DEVELOPMENT AND DESIGN

3.1 System Development Process

The software development process has been divided into several phases which will be described in the upcoming sections. In order for the system to accommodate both the requirements of Job Seekers and Job Providers, The system had to be agile and keep up with change of plans and requirements as they come and go. To address this issue the development strategy of Kanban is followed. Kanban is an agile strategy for software development, where developers list out the things they want to do in a story board. It helps with the visualization of what they need to, what they are doing and what they have completed. This strategy is used to increase collaboration, productivity and sustain the development of the system. For this the tools provided by GitHub is used to keep track of system development as shown in Figure 2.



Job Seeker Profile

Updated 1 minute ago

2 To do
- Create a component for Job Seekers to view their reports / Dashboards — #16 opened by subeshb1
- Create a Component for giving Exams — #15 opened by subeshb1

Automated as To do

0 In progress

2 Review in progress
- Create JobPost component for Job Seeker Home page — #13 opened by tacmoktan
- Feature jobseeker homepage — ✓ #14 opened by tacmoktan — Review required

Automated as In progress

3 Done
- Create Profile Setup for Job Seeker — #10 opened by subeshb1
- Create Nav Bar for Job Seeker — #9 opened by subeshb1
- Add responsive jobseeker navbar — ✓ #11 opened by tacmoktan — Changes approved

Automated as Done

Figure 2: Agile Kanban

The figure above shows a storyboard holding all the issues which is created in order to divide the tasks into detailed sub tasks. For each sub task assigned to an assignee he/she is responsible for completing the task. Once a task is completed it is reviewed and finally is added to the main document or repository.

3.2 System Requirement Specification

For this system, the requirements can be categorized into two categories, namely Job Seeker and Job Provider. We'll be discussing about them accordingly.

3.2.1 Functional Requirement

3.2.1.a Job Seekers

- Job seekers should be able to apply to jobs where their criteria meet.
- Able to acquire skills by giving exams and evaluation process.
- Able to search and query jobs accordingly.
- The system recommends jobs based on their history, preferences and similar user activity.
- View their activity and reports in a dashboard.

3.2.1.b Job Providers

- Job Providers should be able to post jobs.
- Ability to filter out candidates based on filters specified.
- Create exams and hand out skills to workers.
- Define a reusable hiring process to hire candidates which is transparent to the applicants.
- View reports of their job postings and get meaningful reports.

3.2.1.c Use Case Diagram

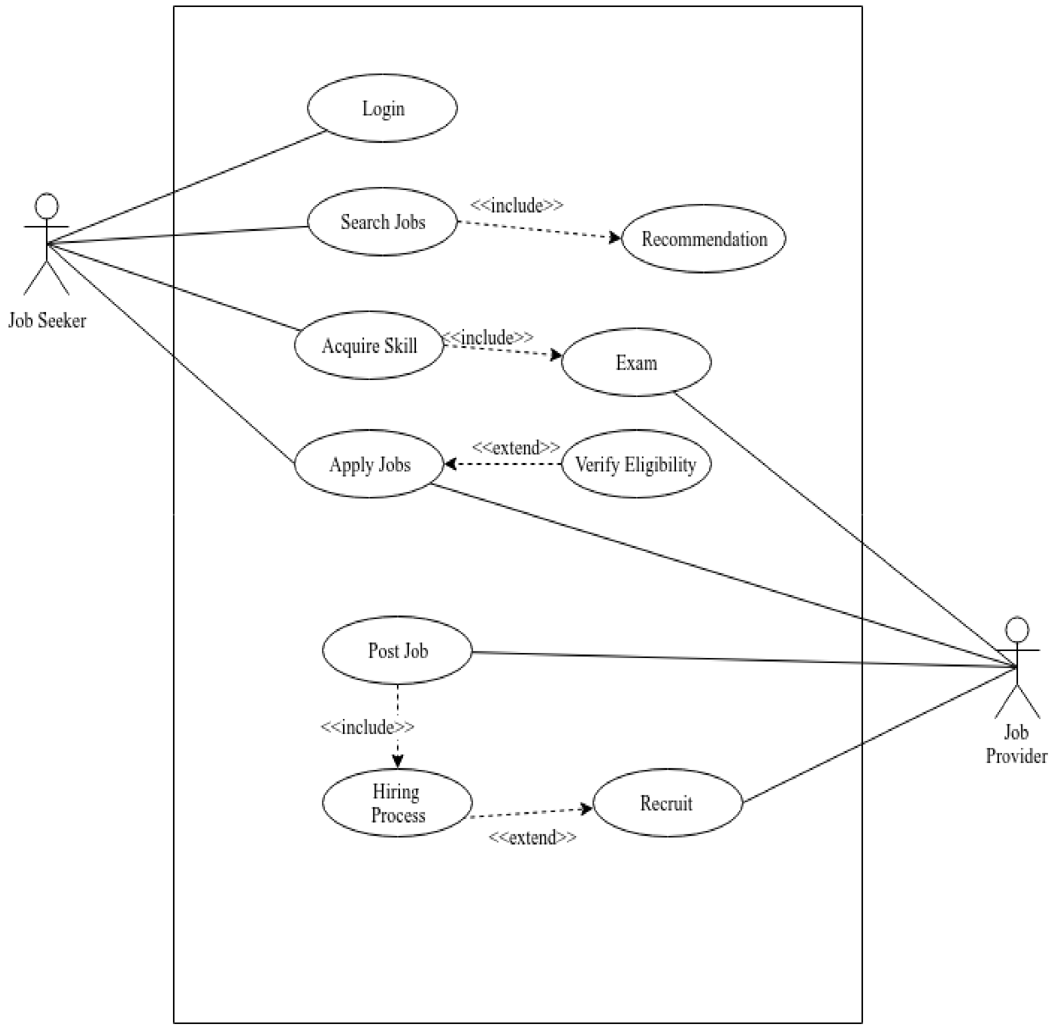


Figure 3: Use case diagram

The figure above shows the Use case diagram of the system. There are two primary actors i.e. Job Seekers and Job Providers. Job Seekers have 4 main use cases i.e. logging in, searching for jobs, acquiring skills and applying to jobs. Whereas, the Job Providers use cases are creating exams posting jobs and recruiting Job Seekers. They act as reactors whenever a Job Seeker applies to jobs. The opposite can be said for the Job seekers i.e. whenever a Job Providers posts a job.

3.2.2 Non Functional Requirement

- The system must be user friendly, responsive, fast.
- The system should provide a secure medium for users to access their system.
- It must be cross platform and have support for all the major browsers.

3.3 Feasibility Study

There are various feasibility studies that can be performed on a project based on what output is required. The proposed feasibility study for this project is:

3.3.1 Schedule feasibility

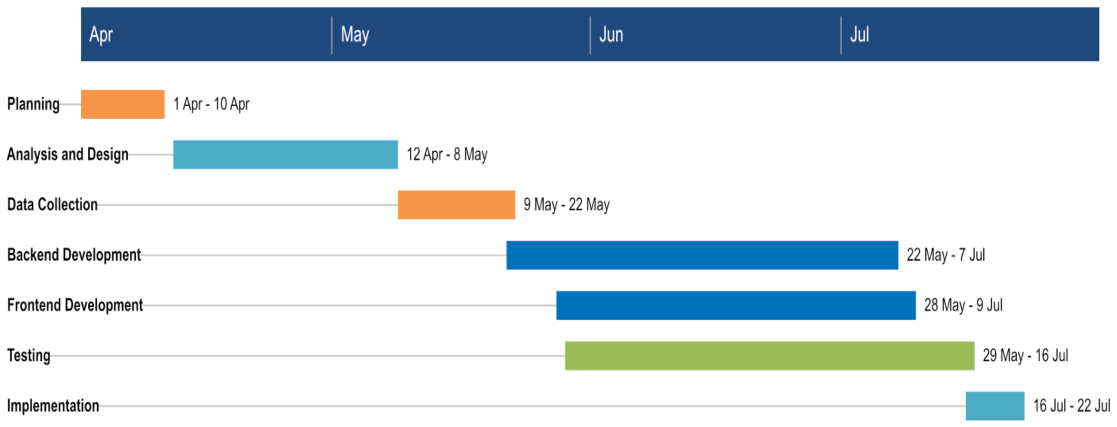


Figure 4: Gantt Chart

3.3.2 Technical feasibility

The project is the application made using various web development tools. The user interface is made using React JavaScript, whereas the application is built using Ruby on Rails. PostgreSQL is used as database. For our deployment and testing netlify, heroku and github was used to ease out development process. As all the tools and framework is open source, the proposed project is technically feasible.

3.3.3 Operational feasibility

The system is proposed to be user friendly with different interfaces to communicate with users. The interface is easy to use as abstraction is maintained. Any user with a computing device and an internet facility can easily access and use the applications, hence making it operationally feasible.

3.3.4 Economic feasibility

All the functionality in the system is developed using an open source platform, making the system a lot cheaper. A user with any computing device and internet facility can use the application without extra cost. Thus, the project is economically feasible as well.

3.4 System Architecture

The system follows N-Tier architecture, where presentation, logic and data tiers are separated. The system architecture can be seen as follows:

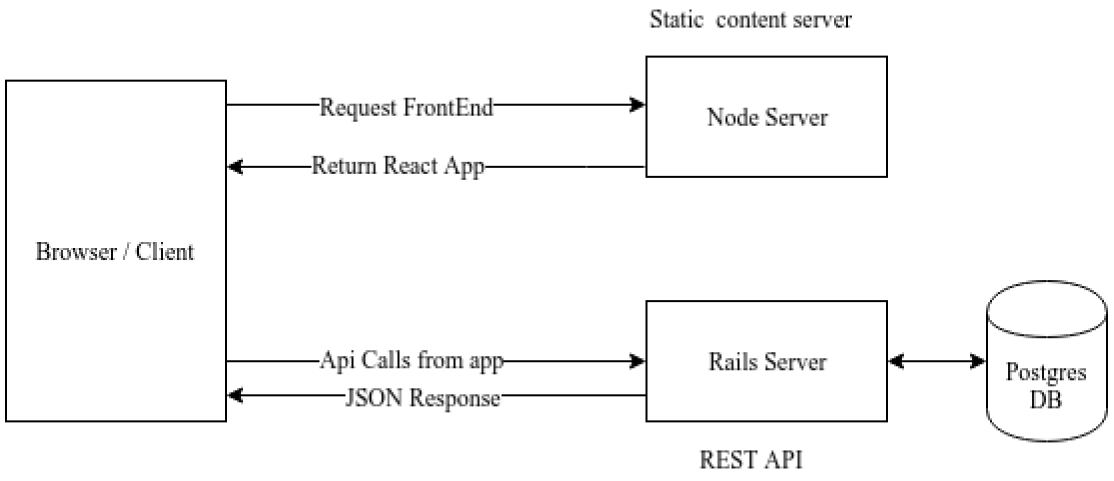


Figure 5: System Architecture

There are two servers, one for serving the static front end and second for handling requests made by the client. The server responsible for handling all the requests is linked with a Postgres DB, and is responsible for all the logic required for the system to work including authentication. The backend is a REST API that returns json response for every request. With this architecture the developer is able to use any front end in the client side to access the backend making it flexible.

The node server sends front end of the system, which then makes requests to the rails server. The front end consists of both the job seeker and job providers interfaces. Once the user logs in they are shown with interface according to their role. The user interface makes request to the rails server and it handles the request and processes the request and sends out a request. The rails server handles all the CRUD actions and updates the database accordingly.

3.5 Sequence Diagram

3.5.1 Job seeker sequence diagram

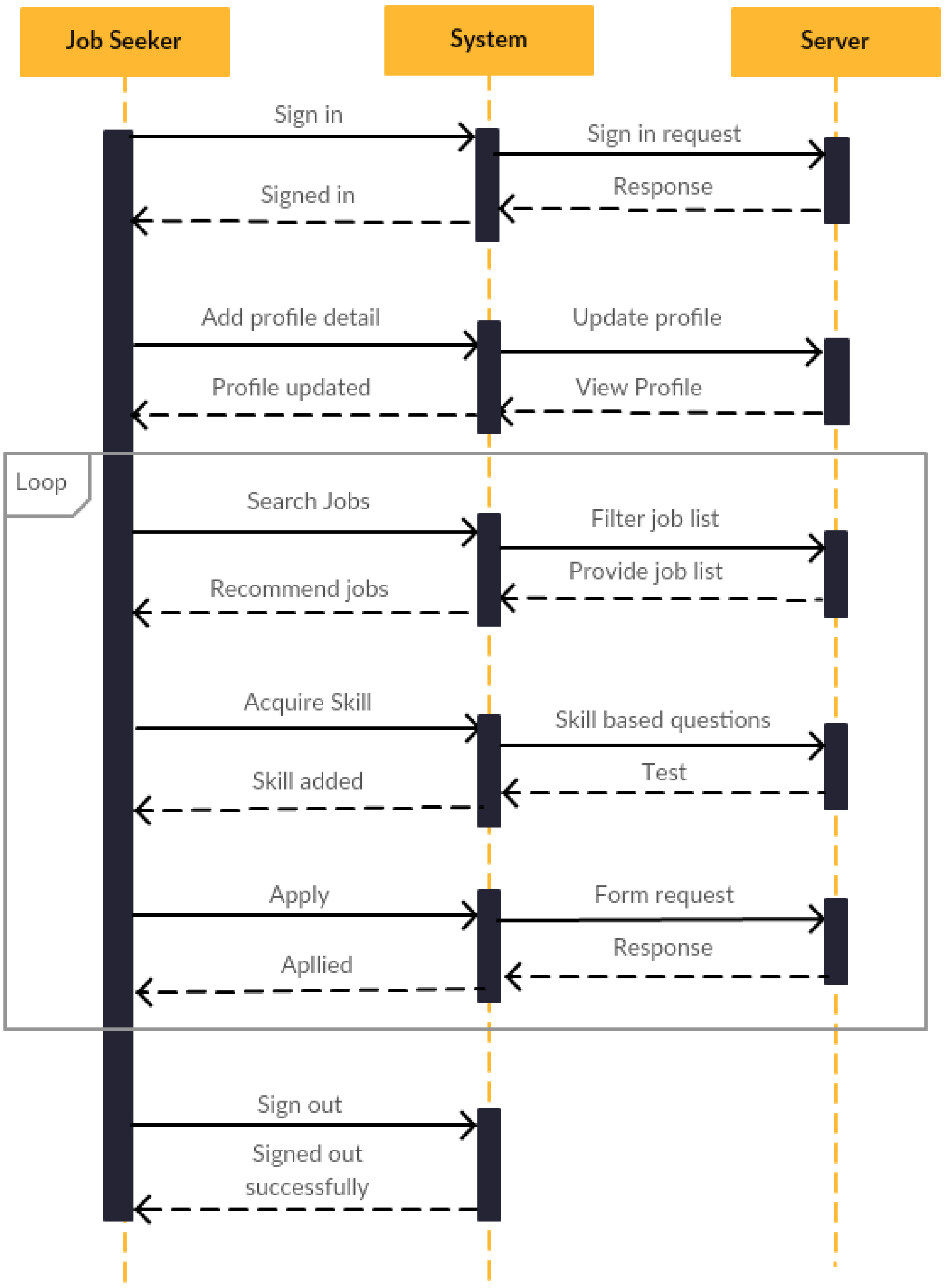


Figure 6: Job seeker sequence diagram

3.5.2 Job provider sequence diagram

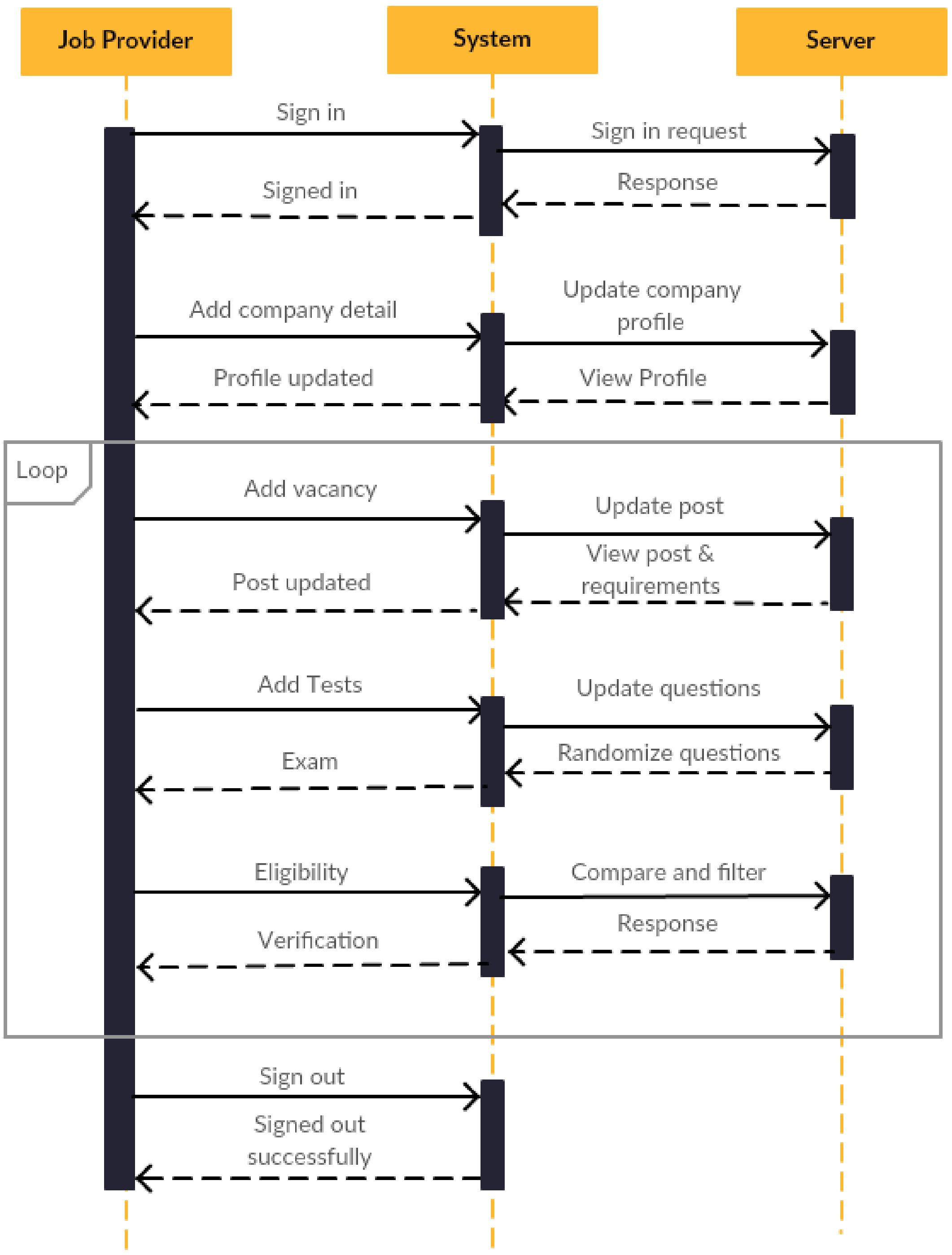


Figure 7: Job Provider sequence diagram

In Figure 6, it shows the activity of Job Seekers and server. First they sign in to the system, then update their profile. Then they search for jobs and the server responds with filtered job list along with the recommended jobs. When they find a job suitable for them, they will apply to the job. If they don't have skills specified in the job then they will earn skill and then apply for the job. This process is continued until they find a job or wish to log out.

In Figure 7, Job Providers sign in to the system and the server sends in a signed in response. They will fill out their profile and server will update the profile. After the profile completion, the job providers will be able to create jobs, add tests and recruit workers. They will be able to perform this task until they find suitable candidates and finally sign out after completion.

3.6 Class Diagram

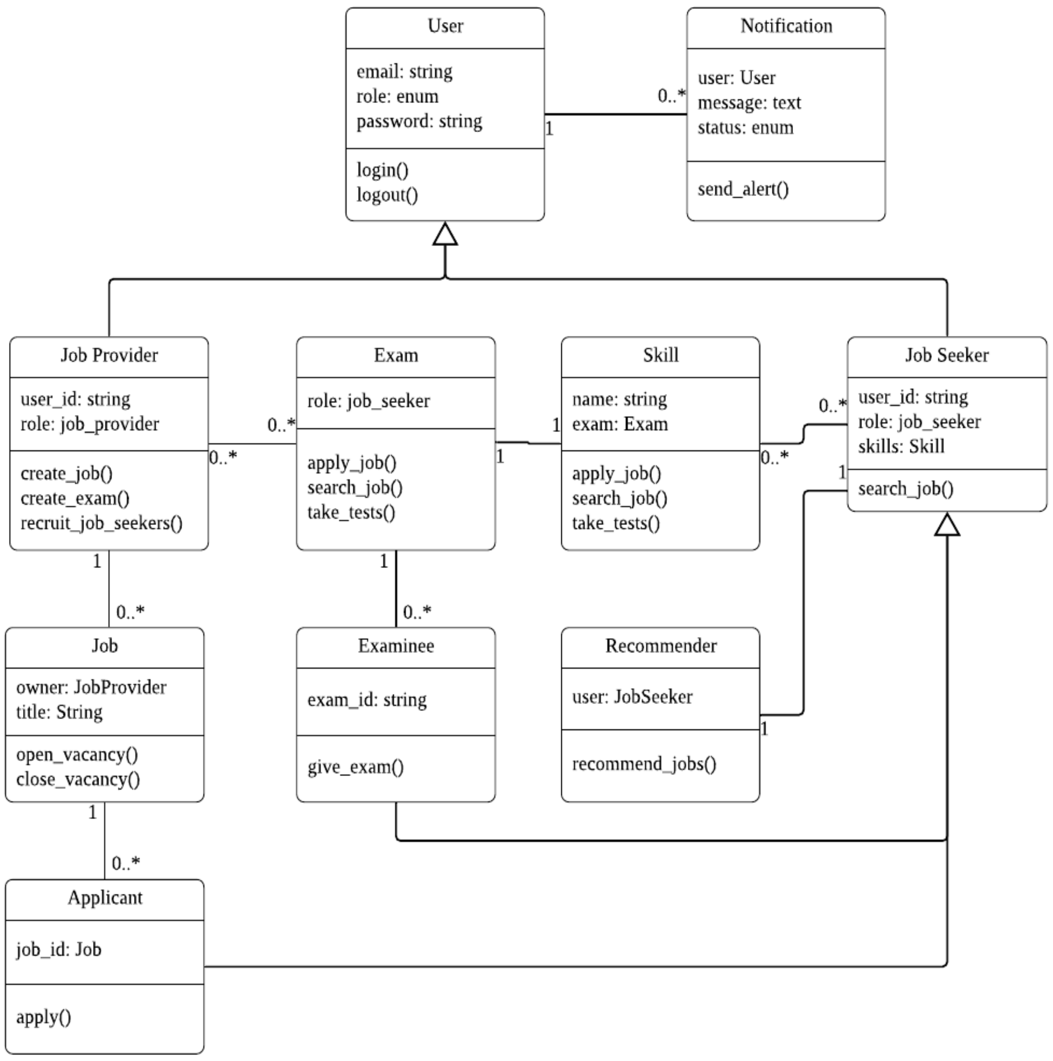


Figure 8: Class diagram

In the above class diagram, there are several classes like User, Notification, Job Provider, Job Seeker, etc. These classes consist of two fields: data and methods. The data represents type of the data that the class stores. And the methods represent the operations that can be carried out within the class. The class diagram also depicts relationship between classes. Each class can have several instances.

The user class can have two types of instances on the basis of their roles i.e. job seeker and job provider. The job seeker is able to search for jobs as well as view recommended jobs. The properties of job seeker instance can be further inherited by classes like applicant and examinee which allows them to apply for jobs and give exams respectively. The job provider is able to create jobs, create skill testing tasks, post vacancies and recruit job seekers.

3.7 Activity Diagram

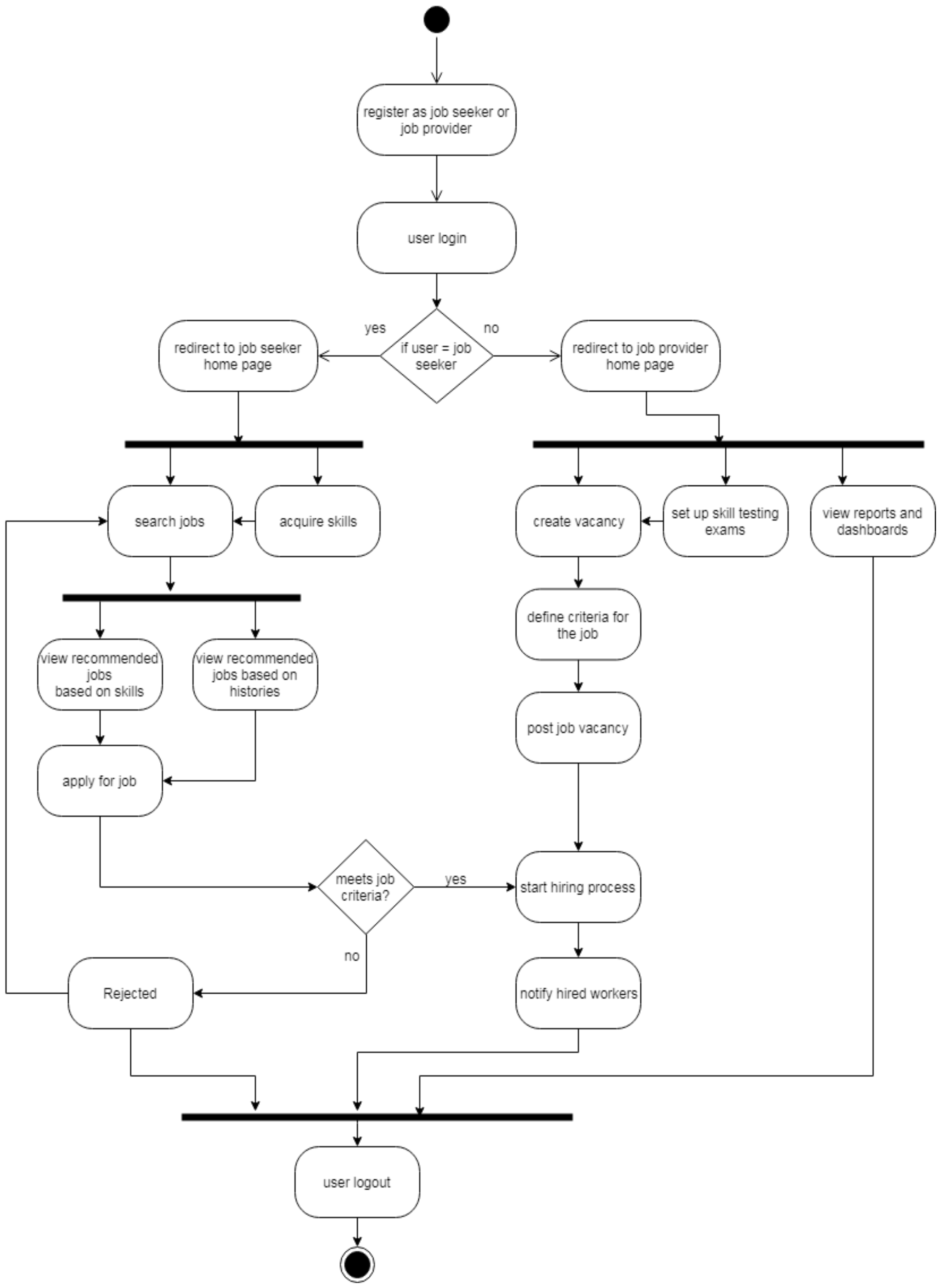

register as job seeker or job provider
user login
yes
if user = job seeker
no
redirect to job seeker home page
redirect to job provider home page
search jobs
acquire skills
create vacancy
set up skill testing exams
view reports and dashboards
define criteria for the job
view recommended jobs based on skills
view recommended jobs based on histories
post job vacancy
apply for job
meets job criteria?
yes
start hiring process
no
Rejected
notify hired workers
user logout

Figure 9: Activity Diagram

In the above activity diagram, the control flow from one activity to another is modeled. The circular black dot represents start point and the circular black dot outlined by another circle represents end point. The activity diagram is similar to flow chart with more focus on the activities performed in the system. The thick horizontal line symbolizes fork as well as join nodes in this diagram. The fork nodes represent the flow from one activity to multiple activities whereas the join nodes represent the flow from multiple activities to a single activity.

The user first registers either as job seeker or job provider depending on their respective roles. The user logs into the system by then the system recognizes their roles and according to that constraint, the user is redirected to their respective home page.

If the user is a job seeker, the fork node symbolizes direction to multiple activities that they’re able to perform i.e. search for jobs and acquire skills. The job searching activity further leads to activities like viewing recommended jobs based on their skills or their histories. Then job seeker applies for job. If the job criteria are matching, job provider starts hiring process, else job seeker is rejected.

If the user is a job provider the fork node symbolizes direction to activities like creating vacancies, setting up exams and view reports. The job provider simply defines job criteria and starts hiring process based on those criteria then notifies the hired candidates. After all the work is done, both of the users logout which is represented by the join node in the figure above and finally the system terminates.

CHAPTER 4

IMPLEMENTATION AND TESTING

4.1 Implementation

The tools used in the implementation of the project are described below:

- Languages

  React, a JavaScript library is used for front-end development. And for backend development Ruby on Rails was used.

- Operating System

  The system development is worked upon both windows OS as well as Ubuntu Linux.

- Development Tools

  The following are the development tools used for this project:

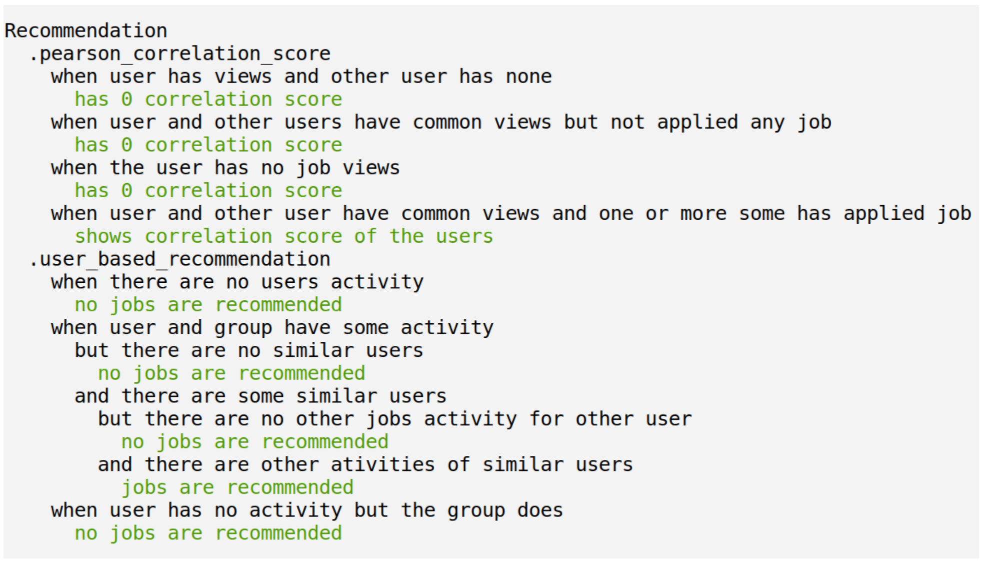
  1. Visual Studio Code
  2. Postman
  3. Git & Github

4.2 Testing

Testing is a process that validates a system by examining it on different conditions. So we carried out the following types of testing on this project:

4.2.1 Unit Testing

Smallest block of codes in the program can be referred to unit. This test helped us examine each unit individually and makes sure that the unit is working exactly how it is intended to. This testing came in handy whenever it is thought of adding new features to the system. Since this testing detects small changes in the program so it made sure that each unit is validated before moving on to a new unit to work. Automated unit tests are performed in major units in our backend side. Since, 100% code coverage is not possible in a short period of time it is only tested major components in our system.



```
Recommendation
  .pearson_correlation_score
    when user has views and other user has none
      has 0 correlation score
    when user and other users have common views but not applied any job
      has 0 correlation score
    when the user has no job views
      has 0 correlation score
    when user and other user have common views and one or more some has applied job
      shows correlation score of the users
  .user_based_recommendation
    when there are no users activity
      no jobs are recommended
    when user and group have some activity
      but there are no similar users
        no jobs are recommended
      and there are some similar users
        but there are no other jobs activity for other user
          no jobs are recommended
        and there are other ativities of similar users
          jobs are recommended
    when user has no activity but the group does
      no jobs are recommended
```

Figure 10: Automated unit test for recommendation unit

The figure above shows the automated test carried out for our recommendation component. It performs unit tests on each code block ensuring that the new changes added in the future won't affect the current existing feature.

4.2.2 Integrated Testing

In order to conduct this testing, the individually validated units in the program are combined together to form a component and the component is put through several tests. The main purpose of this testing is to check how the integrated units interact with each other within the component and whether or not the component as a whole is generating satisfying results. The test is executed by creating various test cases and iterating the component over those test cases one by one.

S.N.	Test cases	Remarks
1.	No internet connection	Unable to connect to the server
2.	Empty input fields	Display error message
3.	Invalid credentials	Display error message
4.	Valid credentials	Display success message
5.	Logged in as job seeker	Redirect to job seekers home page
6.	Logged in as job provider	Redirect to job providers home page

Table 1: Login Component Testing

The table above shows the integrated testing of the frontend and backend. The login form consists of frontend validation and onsubmit triggers backend validation. The above table shows the tests that are performed on the login form. Similarly, tests are performed in other components. Showing all the test cases in the document is not feasible so only the gist of the tests performed is listed out which have been carried out it other components as well.

S.N.	**Test Case**	**Expected**	**Result**

1.	Fetch Profile:
	
	GET /api/v1/profile
		Returns profile in json format with status 200.	{
			"basic_info":{},
			"educations": {},
			…..
			}
			Status: 200

2.	Update Basic Info (valid params)
	
	PUT /api/v1/profile/basic_info
		Updates basic information and returns the updated object in json format with status 201.	{
			"name": "ABC Tech",
			
			"address": "CCC",
			…..
			}
			Status: 201

3.	Update Basic Info (invalid params) PUT /api/v1/profile/basic_info	Returns validation error with status 422	{ "message":"Validation Failed", "errors": [{ "field" : "name","message":"can’t be empty" }]} Status: 422
4.	Unknown api request GET /api/invalid/path	Return not found with status 404	Status: 404 Not Found

5.	When unauthorized path is called.
	Example (job seeker create job)
		Return Forbidden with status 403	{
			"message": "Forbidden"
			}
			Status: 403

Table 2: API Testing

The table above shows the API tests that is carried out. There are well over 25+ routes in the backend. Thus, showing all the test cases is not feasible. In our test case for each request it is tested by two cases i.e. sending valid parameters and sending invalid parameters. For each successful request the api returns response with status 200, 201, 2XX. For invalid request the api responds with 401, 403, 404, 422, 4XX status codes. If there are faults in the server then 500 status code is sent out.

4.2.3 System Testing

This testing is carried out after all the components in the system are complete and fully integrated. The testing was performed in order to verify whether or not the system as a whole meets its requirements. This testing is concerned on finding defects and bugs on the whole system behavior, system design and expectations of the end-user. The testing mainly focused on evaluating functional and end-user requirements. Since the system is platform independent so the responsive view of the system on different platforms are tested:

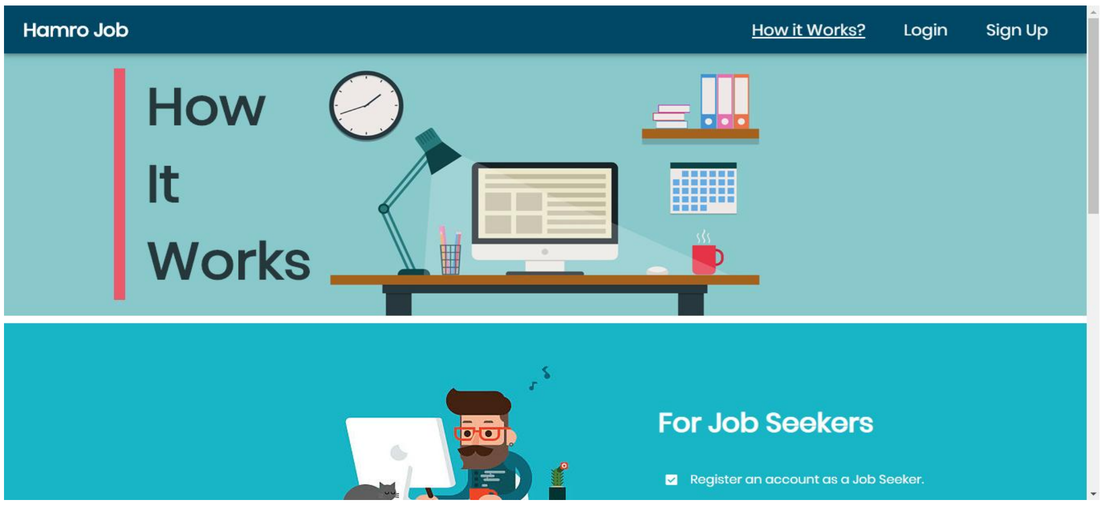
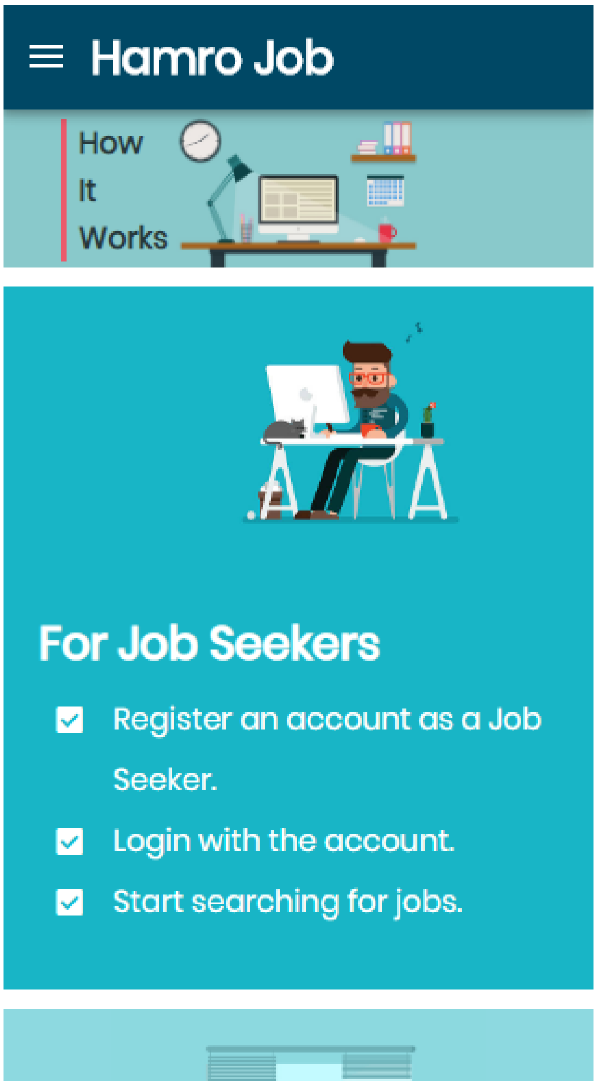


Figure 11: Desktop view



Hamro Job

How It Works

For Job Seekers

- ✓ Register an account as a Job Seeker.
- ✓ Login with the account.
- ✓ Start searching for jobs.

Figure 12: Mobile view

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

Overall, the project is successful. The system is able to meet its requirement and fulfill the objectives discussed. The project is able to stand out from all the generic job portals in Nepal by successfully implementing candidate evaluation system. And the recommendation engine worked successfully by recommending relevant jobs to the job seekers. And the candidate evaluation system showed no issues with its performance as well.

5.2 Recommendations

The system is not limited when it comes to adding further more enhancements to it. It has various possibilities with so much more to be explored and expanded.

The features that could be further integrated to the system are mentioned below:

- Real-time online interview session for job seekers could be conducted by the job providers.
- The profile of job seekers that has not been updated for several days could be dynamically updated by tracing their activities on the site while they are online.
- Learning materials for the enthusiastic job seekers could also be added to sharpen their skills.

REFERENCES

[1] G. Ozcan and S.G. Oguducu, *Applying Different Classification Techniques in Reciprocal Job Recommender System for Considering Job Candidate Preference*, (2016)

[2] M. Diaby and E. Viennet, *Taxonomy-based Job Recommender Systems On Facebook and LinkedIn Profiles*, (2014)

[3] S. Yang, M. Korayem, K. AlJadda, T. Grainger, S. Natarajan, *Combining content-based and collaborative filtering for job recommendation system: A cost-sensitive Statistical Relational Learning approach*, (2017)

[4] W. Chen, Pan Zhou, S. Dong, S. Gong, M. Hu, K. Wang and D. Wu, *Tree-based Contextual Learning for Online Job or Candidate Recommendation with Big Data Support in Professional Social Networks*, (2016)

[5] W. Hong, S. Zheng, H. Wang, *Dynamic User Profile-Based Job Recommender System*, (2013)

[6] W. Shalaby, B. AlAila, M. Korayem, L. Pournajaf, K. AlJadda, S.Quinn, and W. Zadrozny, *Help Me Find a Job: A Graph-based Approach for Job Recommendation at Scale*, (2017)