Job-Client Webapp

Digital Heritage

A Report Submitted
in Partial Fulfillment of the Requirements for the Course
B.Tech. Project

by

Manish Kumar (B17CS032)
Manisha (B17CS033)

under the guidance of

Dr. Suman Kundu

to the

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY JODHPUR

Contents

1. **Introduction**

2. **Architecture**
2.1 Overall Architecture
2.2 Job_type Architecture
2.3 Job Architecture

3 **Technology Used**
3.1 MERN Stack
3.2 HTML CANVAS 5

4 **Features**
4.1 User
4.2 Job_types

5 **Future Works**

6 **SCREENSHOTS**

References

INTRODUCTION

The objective of this project was to make a web application where a job provider can provide jobs from the set of available job types and the job holders can complete the jobs. Any user can register himself as a job holder (to do the job) or a provider to give jobs. After registering, the users have to log in and then they go to their respective pages according to the user type they have chosen during the registration.

There are three types of jobs available currently which are “Adding Description”, “Adding Annotation”, “Adding Images”. More job types can be added by the admin from the admin page by just adding the title and the route of the job type code. It will be automatically reflected on the provider page.

We have used MERN stack development technologies to build the project. Apart from that, we have used CSS for styling and HTML5 Canvas for making the editor for annotation job, and Javascript for scripting. The project uses the previous CROWDANN editor for the annotation job which was made in the previous BTech project by us.

ARCHITECTURE

The whole project can be broken down into three parts that are
i.) Frontend
ii.) Rest APIs
iii.) DataBase

The project runs on two ports one is for the frontend and the other is for the rest APIs. Whenever a user logs in the server sends a JWT token which is saved in the session storage and whenever a request is made to the server this token is sent along with the data for authentication. After authentication, the server fetches the data from the database and sends it to the frontend.

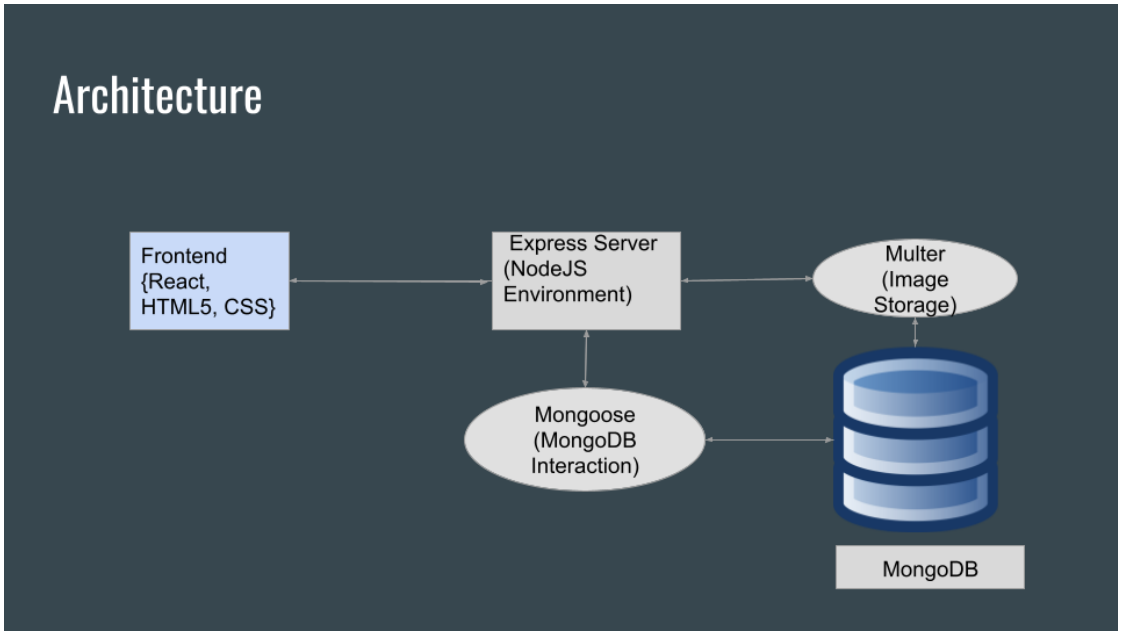
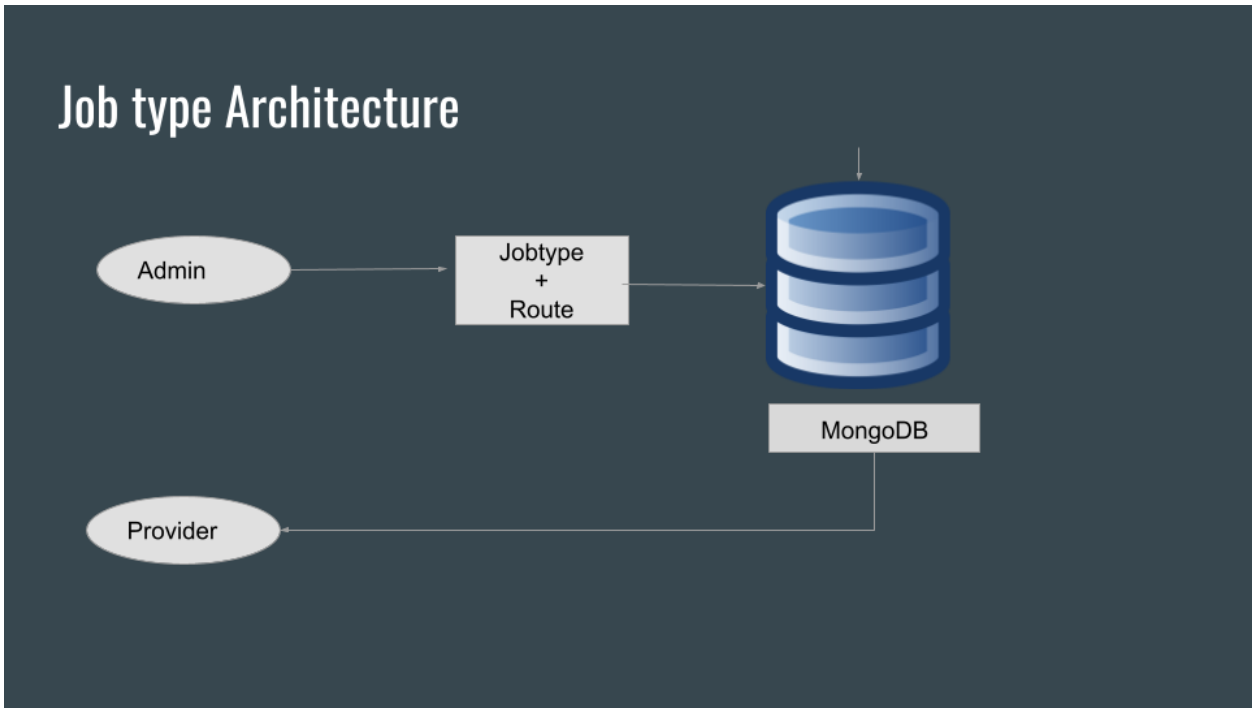


Fig: Architecture

We have used Multer library for storing Images. Multer stores the Images in a given local that can be the server or any cloud storage and returns the link to the path which we have saved in the database. Mongoose library is used for query processing and interacting with the MongoDB.
The Admin has the access of adding a Job type. Currently we have three Jobtype

1. Writing description
2. Adding Images
3. Annotate Images

The Admin can add another Job Types by just adding the title and route of the Job code and the type will be added in the database and gets reflected in the provider page. The provider can assign Job on Job Type and the Job holder can see the Job.



The Job posted are saved in the database with a specific Job_id and then the client can add a solution to the Job. The solutions are reflected to the provider. The provider can download the solutions like in the annotation Job coordinates are saved in the database and the provider can download those in JSON Format.

TECHNOLOGY USED

Here we have used the MERN stack technology to build the project. MERN stack is a combination of four technologies MongoDb for database, Express for APIs, React for Frontend, and Node for the setup environment. Apart from this Cascading Style Sheet CSS was used to style the editor. HTML5 Canvas was used as the image loading and drawing platform.

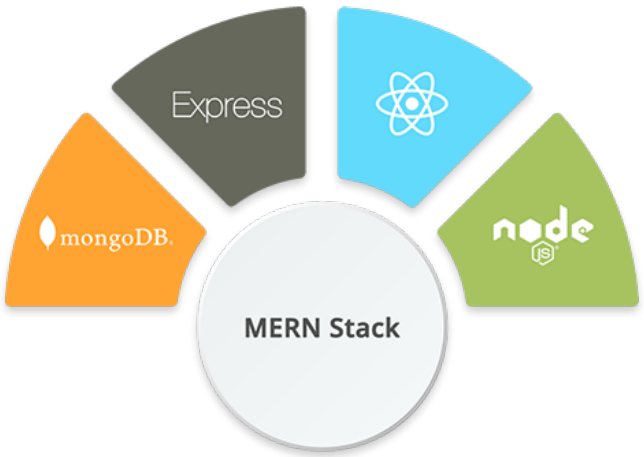


Fig: MERN Stack

i) MongoDB

We have used MongoDB as our database. It is a NoSQL document-oriented database. It stores data in JSON format. We have used MongoDB(local) and MongoDb Altas (cloud) for our project. Mongoose library is used to connect with the database by the specified URL. Since the structure of ur database is like a Graph and some of the fields are of type array so we have used MongoDb rather than SQL database because in SQL database multiple storage in same field is not preferred.

ii) Express Js

It is a backend web application framework for NodeJs. It is used for building APIs. It is very easy to use and supports lot of other libraries that can be used to build efficient APIs. Some of the libraries are Multer (used for saving images,videos), Mongoose used to connect with MongoDB, etc. It uses JavaScript as a coding language so it becomes easy while using react.

iii) ReactJs

It is an open-source library that is maintained and developed by Facebook. It is a Javascript library used for making User Interfaces. It is used to develop single-page applications. However, it only renders data to the DOM (Document Object Model) and so it needs additional libraries such as Routers or ReactDux for making react applications. The advantage of using react is the concept of components and state after changing a single data of the state the whole page re-renders itself.

iv) HTML5 Canvas

We have used Canvas in the annotation Job. The Canvas Element provides a means for drawing graphics via JavaScript and the HTML Canvas element. Among other things, it can be used for animation, game graphics, data visualization, photo manipulation, and real-time video processing.

The Canvas API largely focuses on 2D graphics. The Canvas provides different types of function like making rectangle, drawing free hand, displaying image and getting coordinates using the canvas object.

FEATURES

The Webapp has different types of features based on the type of User. We have three types of users here which are Admin, Provider and Client.

Admin: Admin has the access of all the databases. He can add or delete Job types which gets reflected in the provider page. He also has access to all the User and Job list. He can see all the submissions Jobs in the Joblist tab. He can also see the submission details of a particular user in a Bar Graph.

Provider: At present the provider can add Jobs by choosing from the Job Type dropdown and can post Job. he can see the number of solutions for a particular job and also can download the

solutions for the annotation Job. The solutions are stored in a .txt file in JSON format. He can also see the details of the user who has submitted the solution and can contact them.

Client: The client can complete the Job assigned to them. They can write description about the image, add images and annotate Images. In the annotation part they have got an editor where they can annotate images and the coordinates of the pixels of the Image is stored in the database.The clientpage is separated in two parts with a toggle one is new job section and the other is completed job section.

The user can register himself/herself as a provider or client during time of registration. Based on the type the user will be able to access the clientpage or provider page.

JOB TYPES

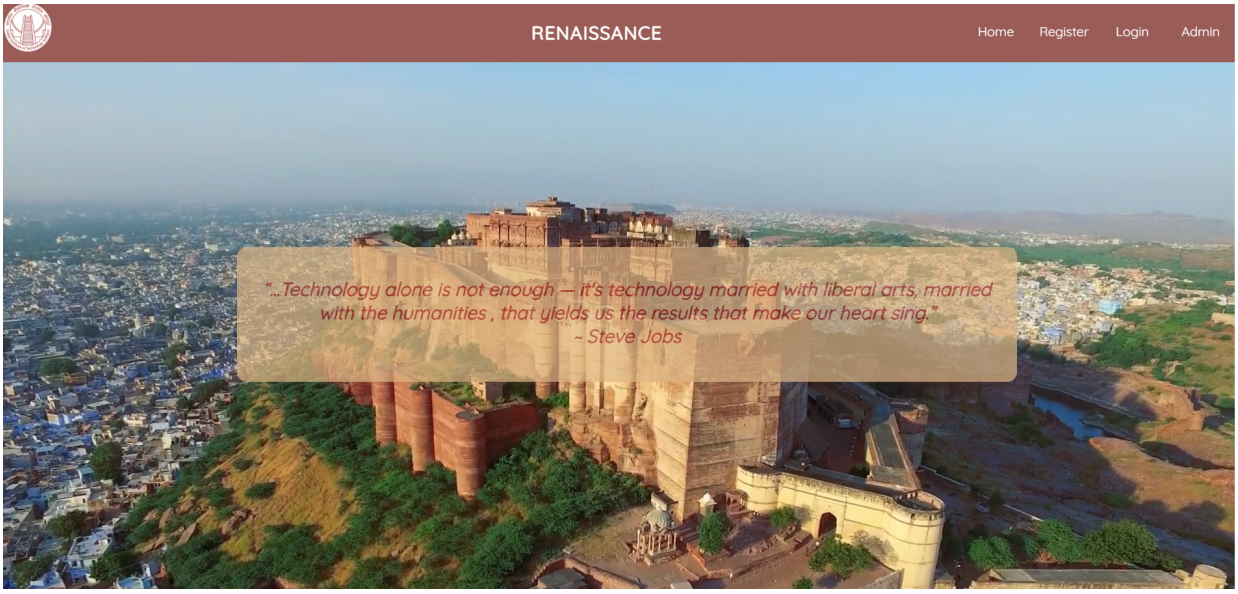
Currently the provider has an option of three Job types, but the admin can add the Job and the provider page will be updated.

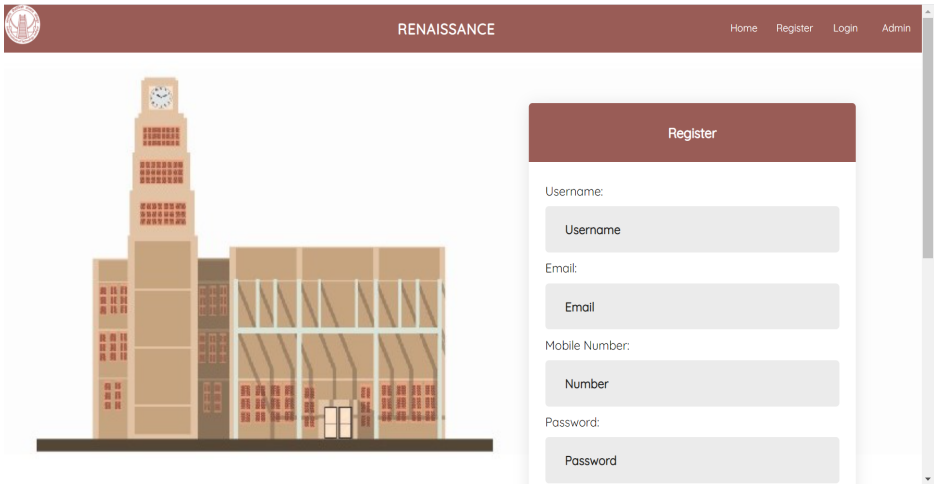
1. Writing Description: Choosing this job the provider has to provide an Image and Title of the Job. The client can write the description and the provider will be able to see the answer.
2. Adding Images: The provider will have to add title and description about the image that he wants and the client can add multiple images and the provider can view all the images.
3. Annotating Image: Choosing this job the provider has to provide an Image and Title of the Job. The client with the help of the editor can annotate the Image and all the annotations will be saved in the database in the form of pixel coordinates. The provider can download the data in JSON format.

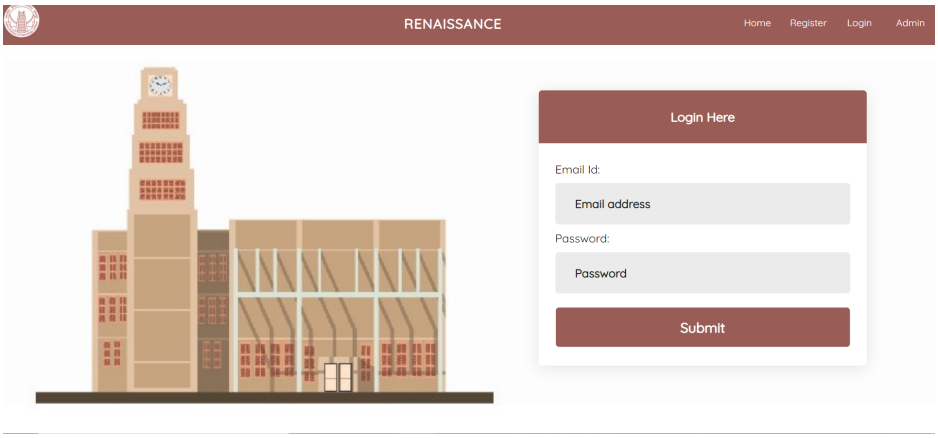
FUTURE WORKS:

- Deploying the Webapp
- Adding more Job Types
- Making it more scalable

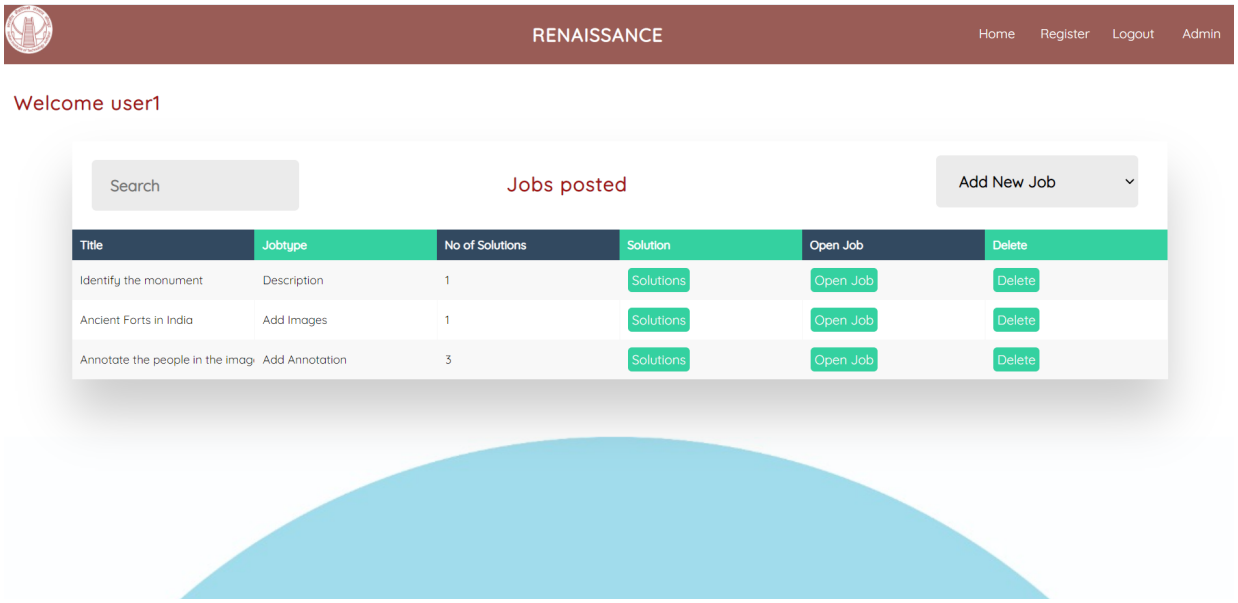
SCREENSHOTS



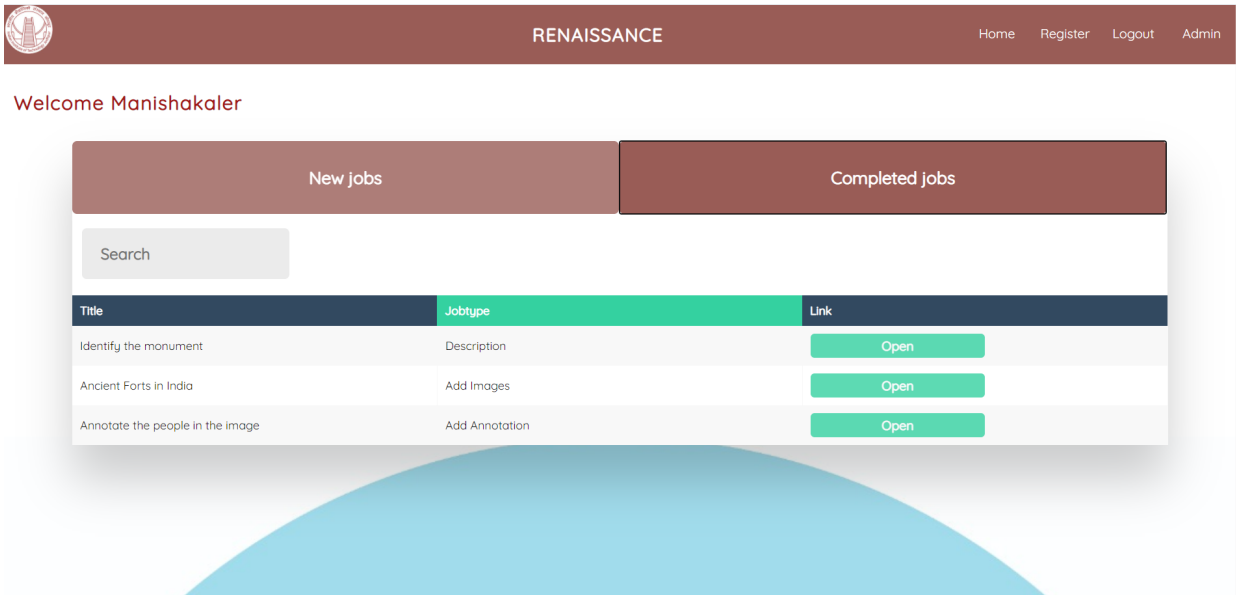




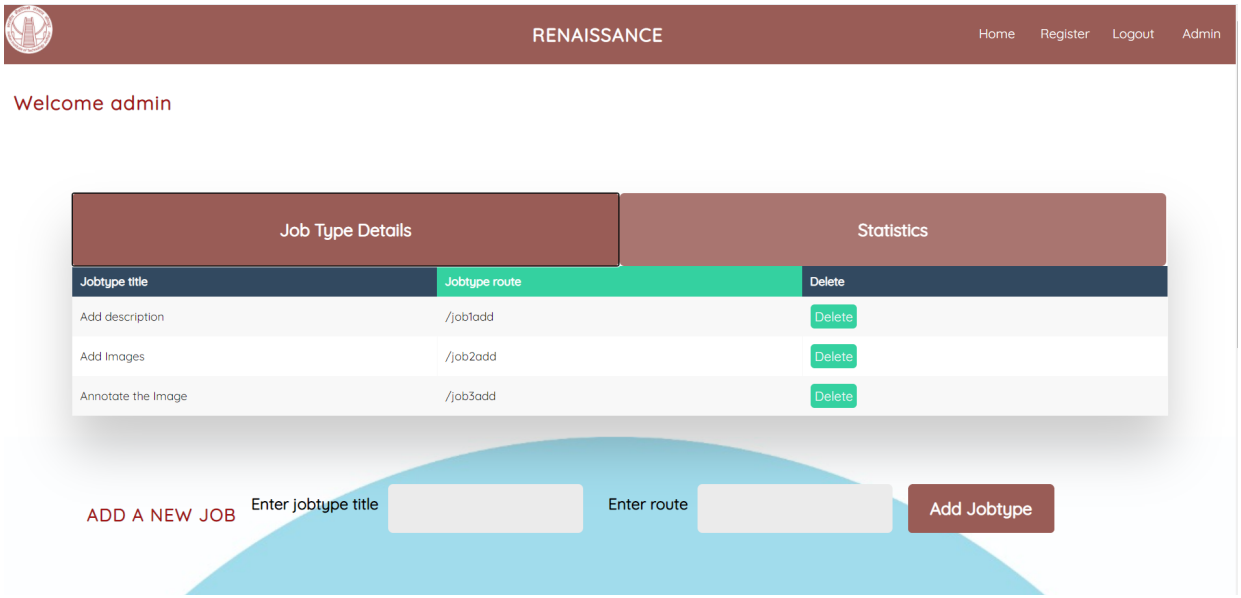
Job-Provider

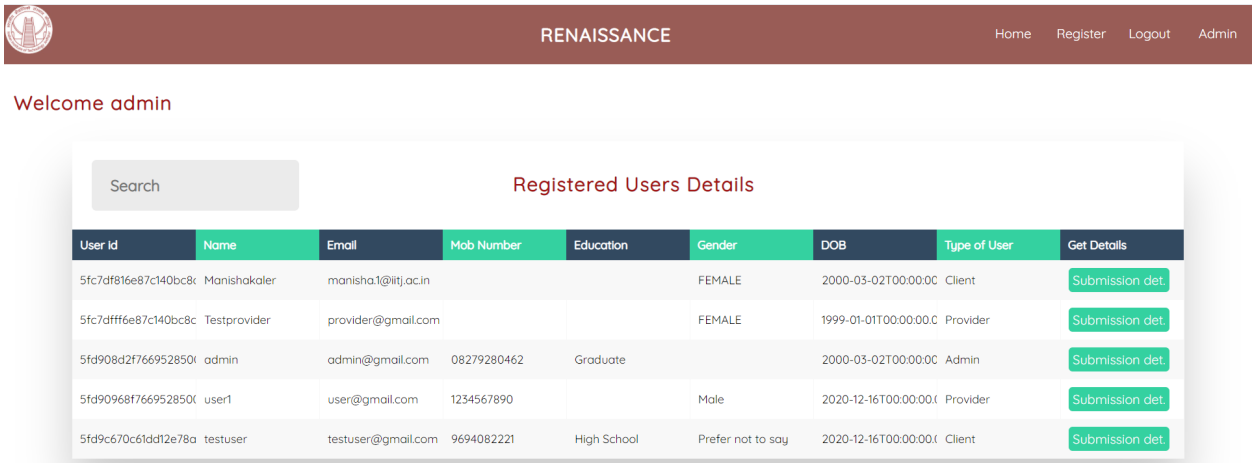


Job Holder



Admin:

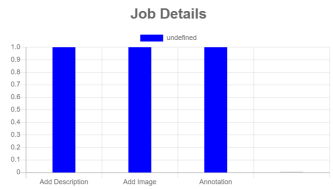




Description Job = 1

Add Images Job = 1

Add Annotation Job = 1



Job Types:

1. Description

Welcome Manishakaler

Title: Identify the monument

Description:Name the monument and where it is located?

Solution:

Write your solution

Submit Job

2. Database collection

RENAISSANCE

Home Register Logout Admin

Welcome Manishakaler

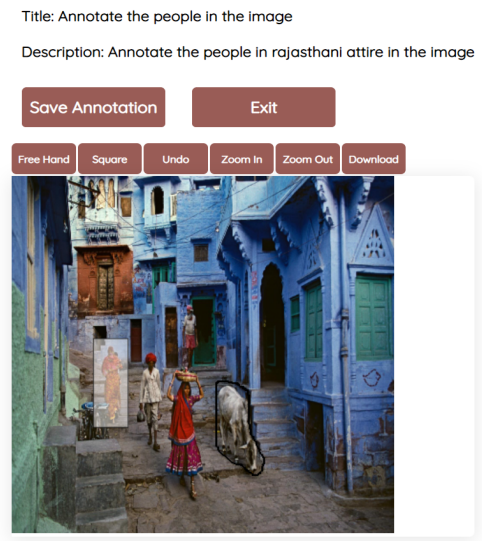
Title: Ancient Forts in India

Description: Provide with ancient Images of forts in India

Choose Files No file chosen

Submit Job

3. Annotation



References :-

- https://reactjs.org/ react_librarry/uses
- http://www.html5canvastutorials.com/
- https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API