**CPME – 282 – Final Project - ConnectMe**

**Introduction**

Project Name: ConnectMe  
University Name: <http://www.sjsu.edu/>  
Course: [Cloud Services](http://info.sjsu.edu/web-dbgen/catalog/courses/CMPE282.html)  
Professor: Andrew Bond   
ISA: Harini Balakrishnan

**Students:**  
Julian Simon

Paramdeep Saini

Sandhya Gadgoli

Viswanath Kambam

GitHub link: <https://github.com/jusimon/ConnectMe>

**Demo**

ConnectMe Demo

**Project Idea:**

Often we see that hiring the right candidate for a job requirement is too complicated process. This happens because of various reasons like unstructured interviews process, inefficient resume screenings and pre-interview calls. Recruitment professionals and hiring managers often pick candidates dependent on abstract, instead of employment related, criteria.

**Project Solution Description:**

Our easy-to-use pre-employment tests improve the new-hire screening process by identifying the most qualified candidates that accurately represent themselves. Leveraging the AWS cloud services, we have implemented our application to be highly available and scalable solution.

It's smart to utilize an structured interview that can alleviate the risk of bad selection of a resource. However, that choice must incorporate different types of assessments. By keeping the face to face interview as a final step in hiring process and evaluate initially based on online tests in various subjects/technologies, can improve candidate sourcing, interviewing and applicant tracking for a streamlined hiring process.

* Recruitment Management tool is important to have a state of art recruitment system that assists the companies in initial screening of the candidates based on the profiles provided by the different companies.
* This tool will provide a list of trusted resources that has in depth knowledge about the technical skills that companies are actually looking forward to and at the same time initial screening of the candidates will be done based on the inputs/questionnaires provided by the company.
* Performance of each individual candidate will be recorded and provided the same to the candidate and Recruiter.

**Features List /. Project Requirements & Use cases**

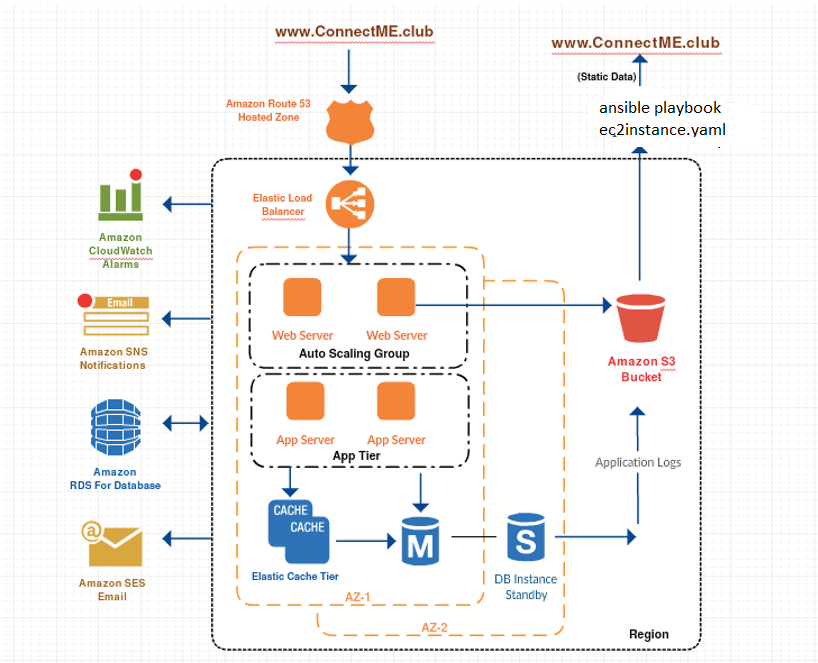
1. Applicants register under our portal which enables them to take the test. A new user record is created in **RDS**. If an already existing user tries to sign up, he is displayed alert message “Already Registered”.
2. Username and password are validated, and Authorized users can only successfully login which will be indicated by alert message.
3. We have role-based login which controls the access to different pages.
   1. **ConnectMe Admin**
      1. Manages Entire Portal and has Super admin access
      2. Add and Delete Tenancy
      3. Add and Delete Candidates
      4. Manage Questionnaire
      5. Publish Questionnaire
      6. Evaluate Tests
      7. View and Publish results
   2. **Recruiter**
      1. Manage Questionnaire
      2. Publish the results
   3. **Candidate**
      1. Login
      2. Update Profile
      3. Take test
      4. View results
4. Myproject Page acts as main activity area and Different users can perform the activities described above as per their respective roles.

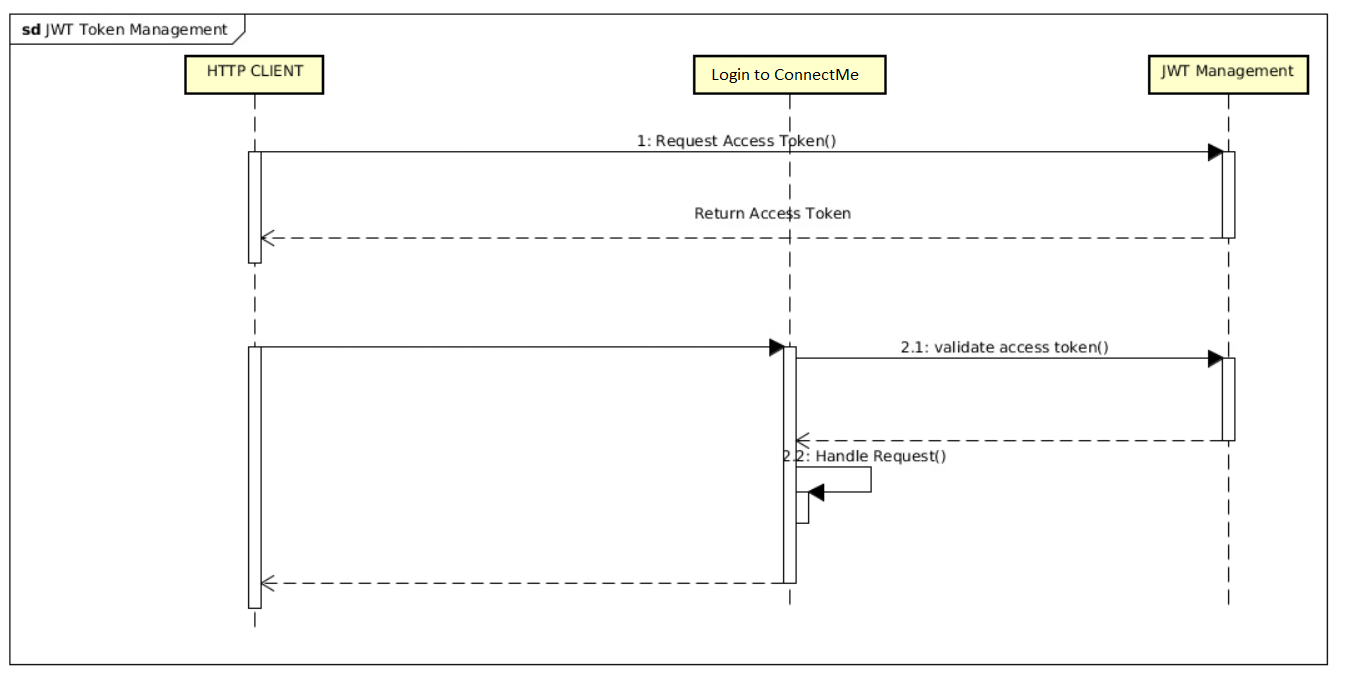
**Amazon Cloud Services Leveraged:**

1. Amazon Machine Learning with RDS is used to display predictive test performance pattern by the candidates on the question set.
2. Amazon Polly used for reading the questions to the candidate.
3. Route 53: The IP address of the application will be resolved by this Domain Name Server. Used route-53 to resolve IP address of our domain to the elastic beanstalk /EC2
4. Elastic Beanstalk:
5. Using Ansible playbooks for deployment on AWS EC2.
6. Autoscaling Group: Auto-scaling group provisioned to scale up EC2 during peak demand or higher availability and scalability.
7. CloudWatch: To set up monitoring on the Autoscaling group instances.
8. S3:
9. IAM:
10. SNS: Notification any failure in autoscaling group detected by CloudWatch.

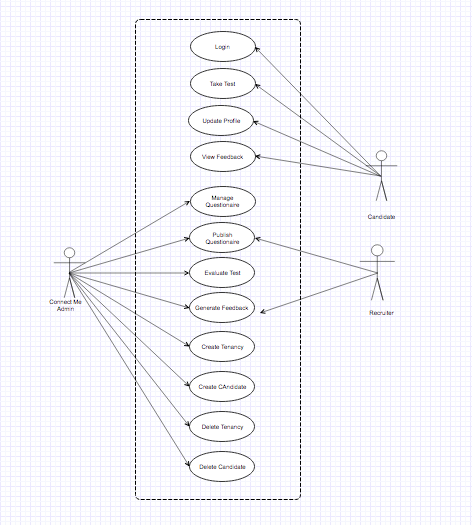
**System Architecture & Workflow modules**

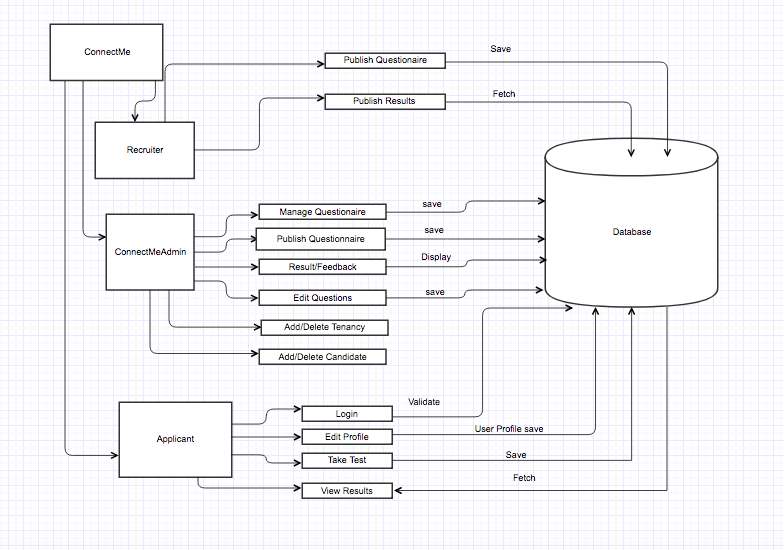
**Architecture Diagram**

****

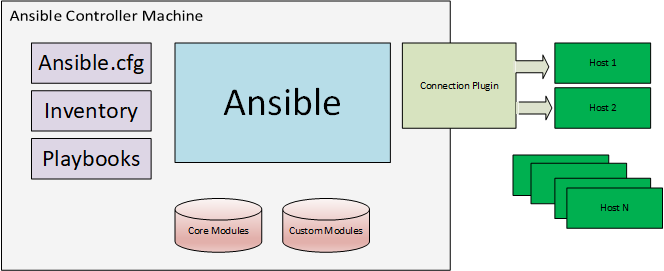
****

**UML diagram:**

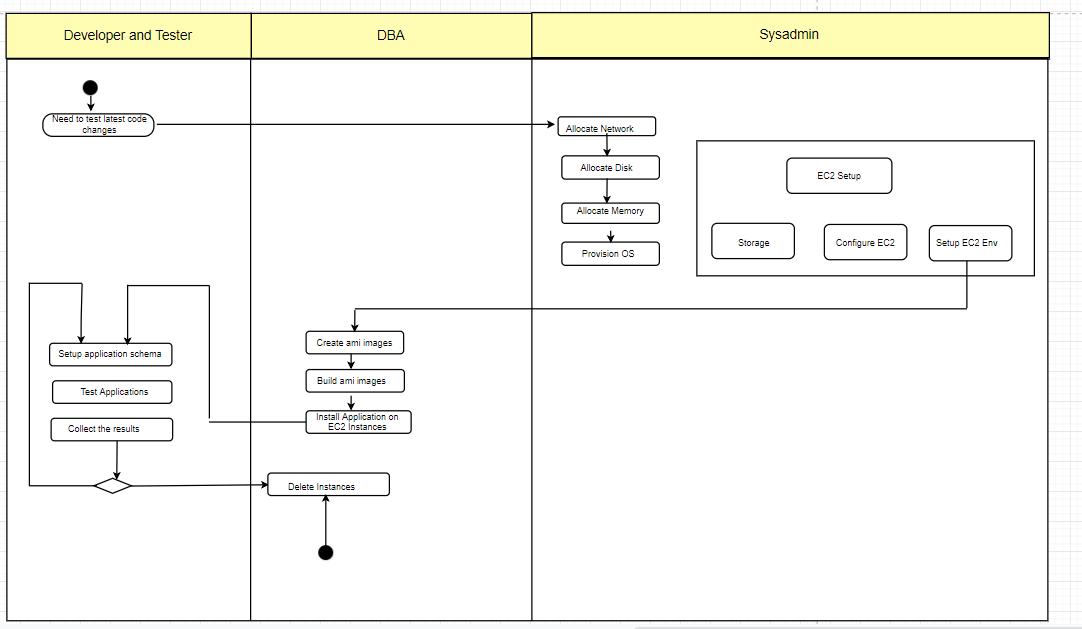




**Deployment Diagram:**

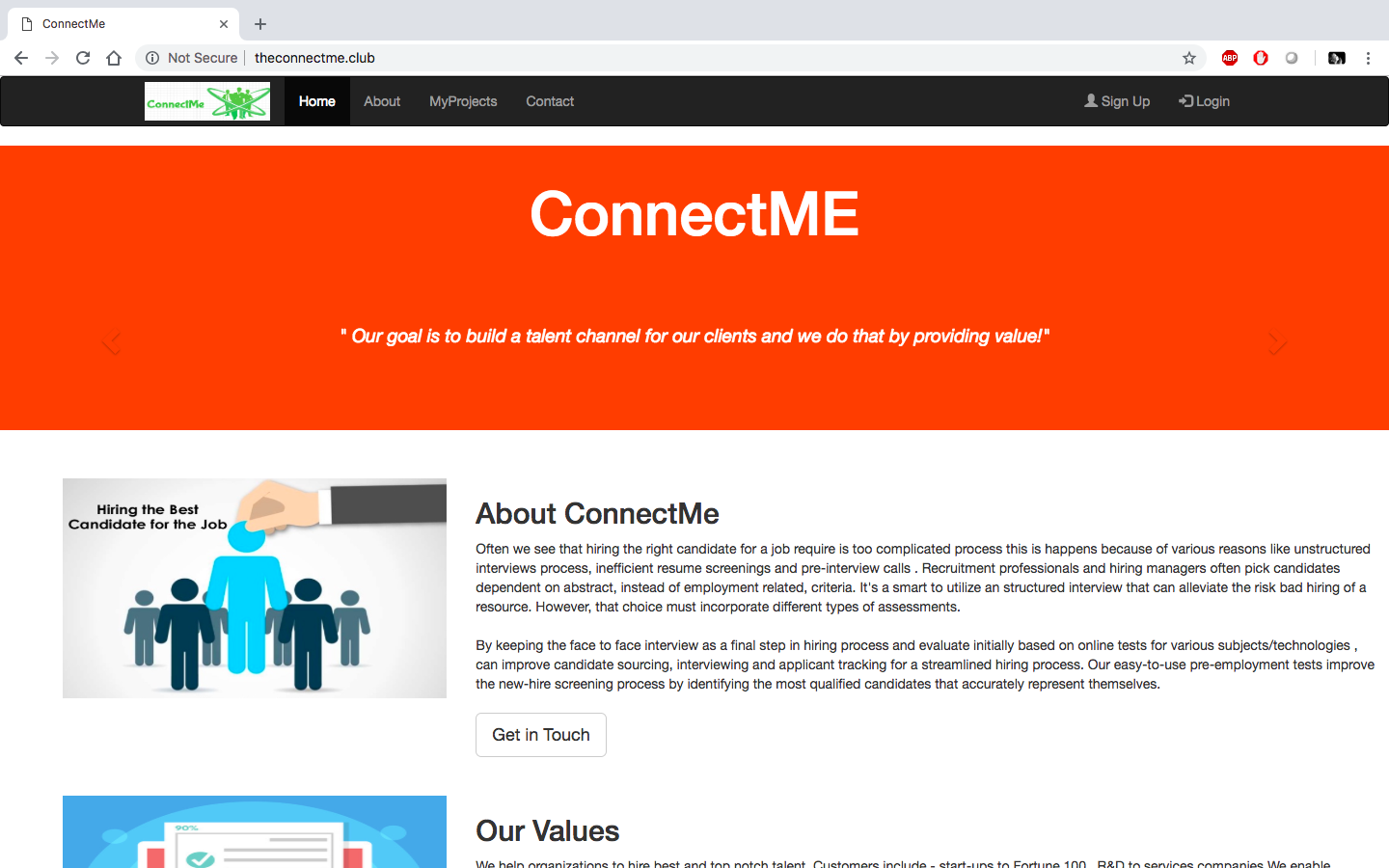


**Sequence Diagram for Spinning new instances of application:**

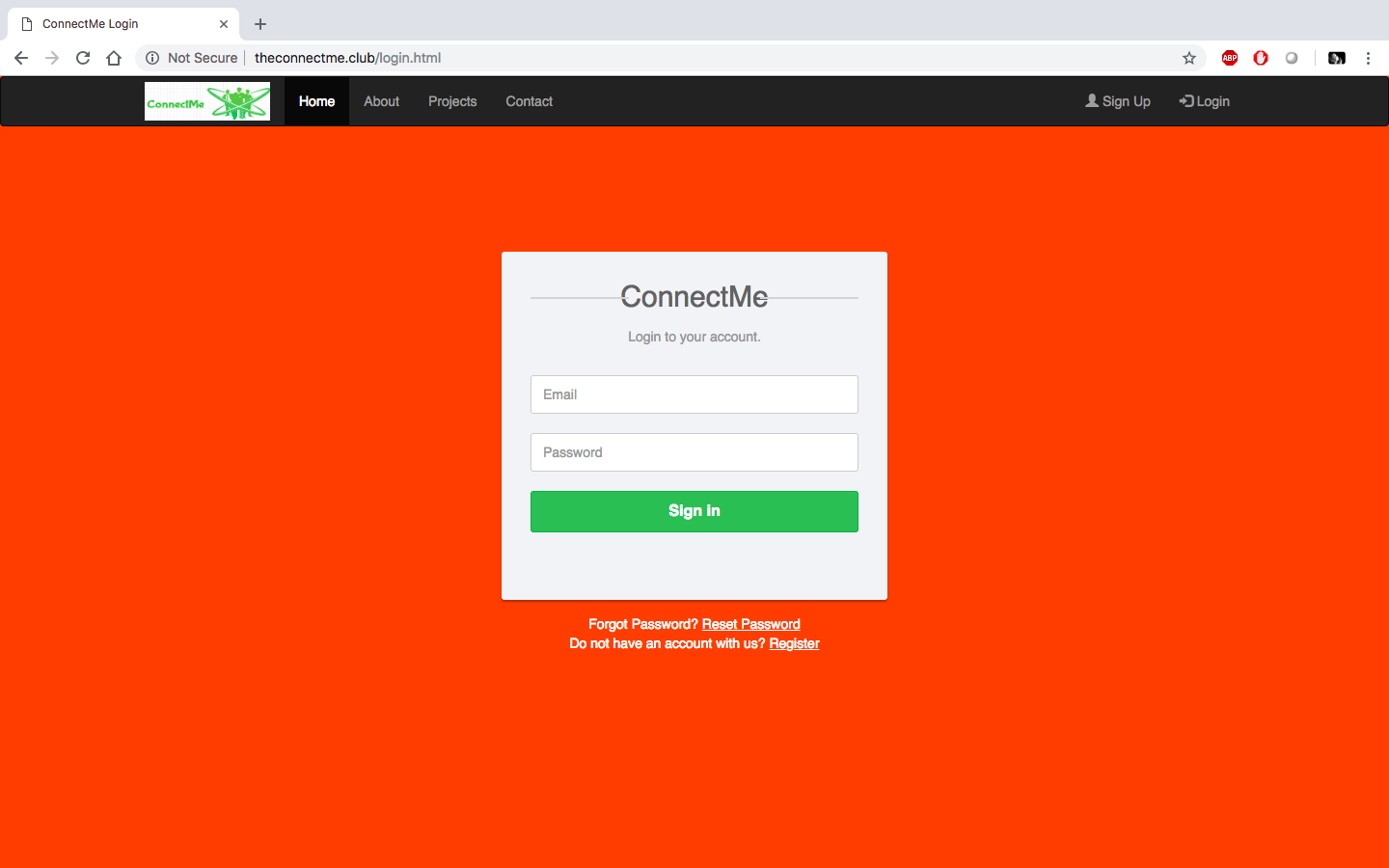


**Application Screenshots /Project Accomplishments**

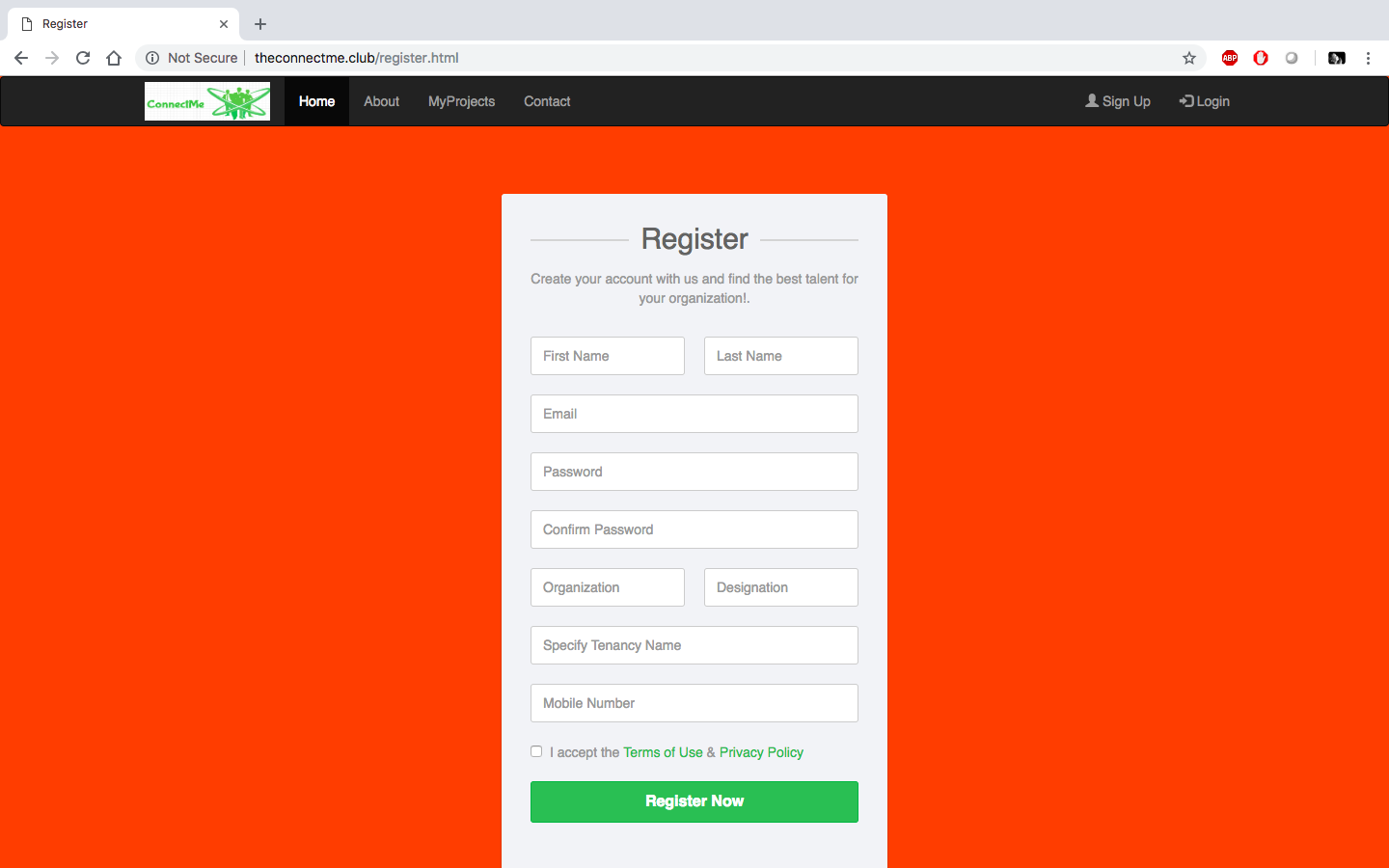
* Home Screen

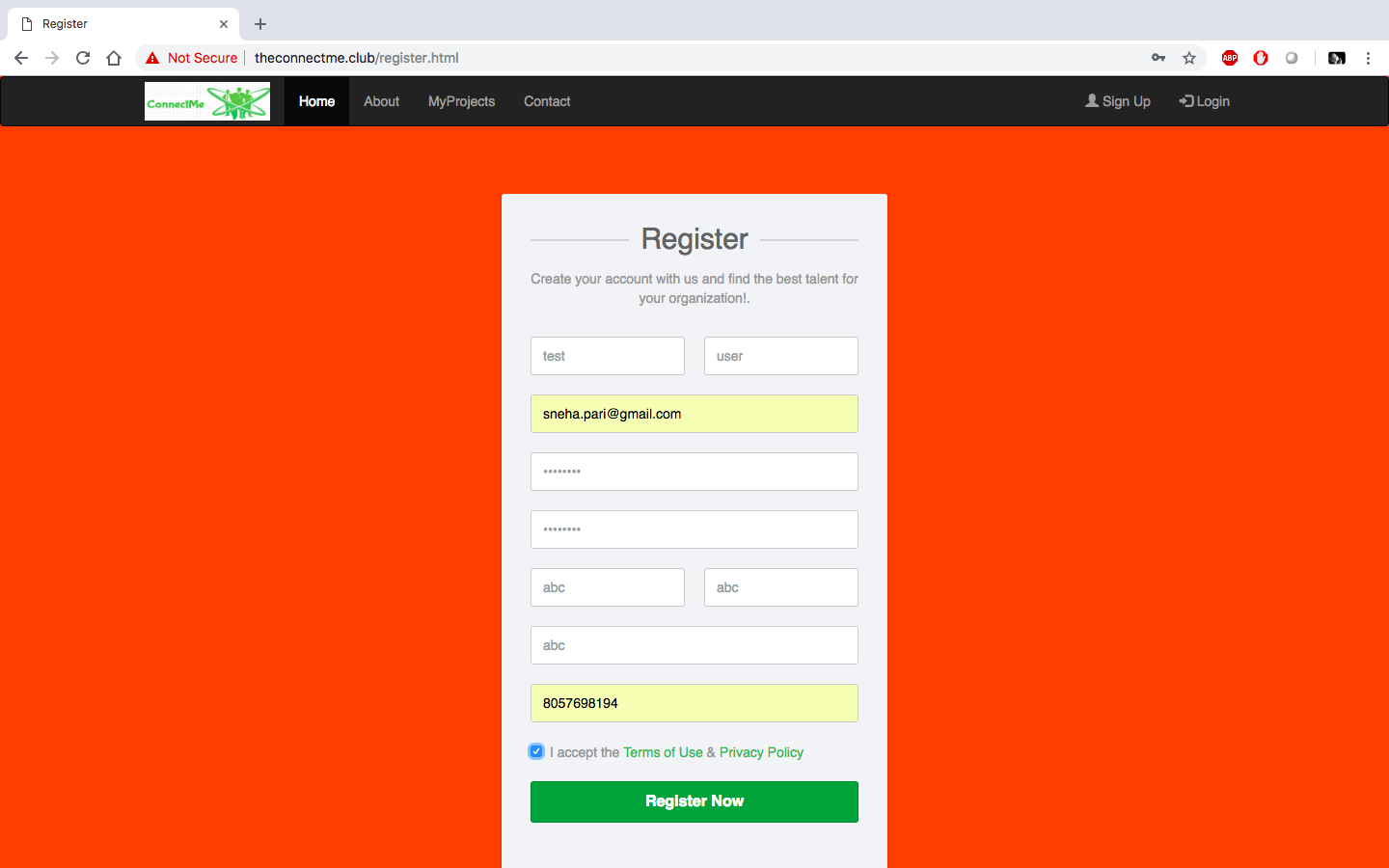


Login Screen

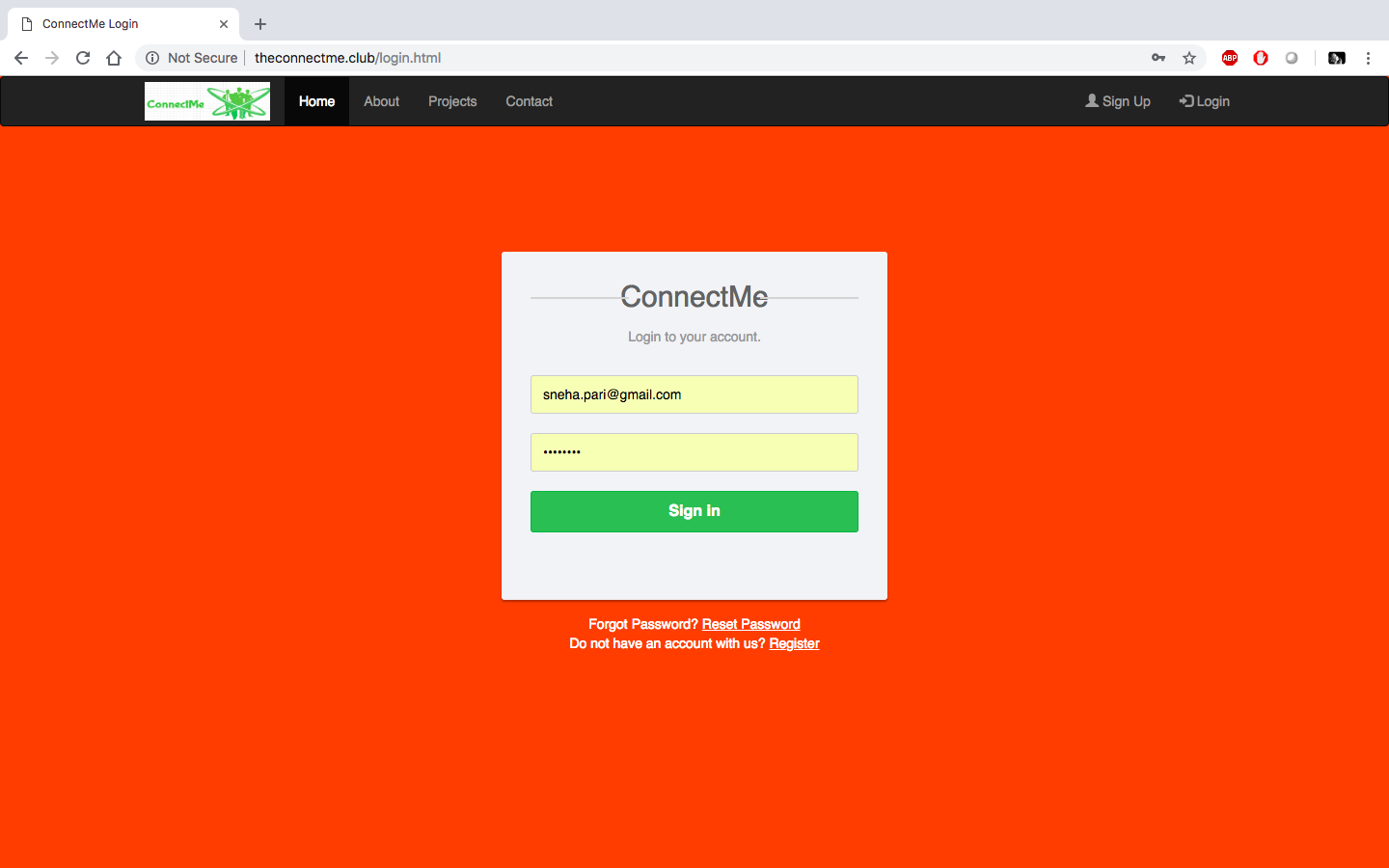


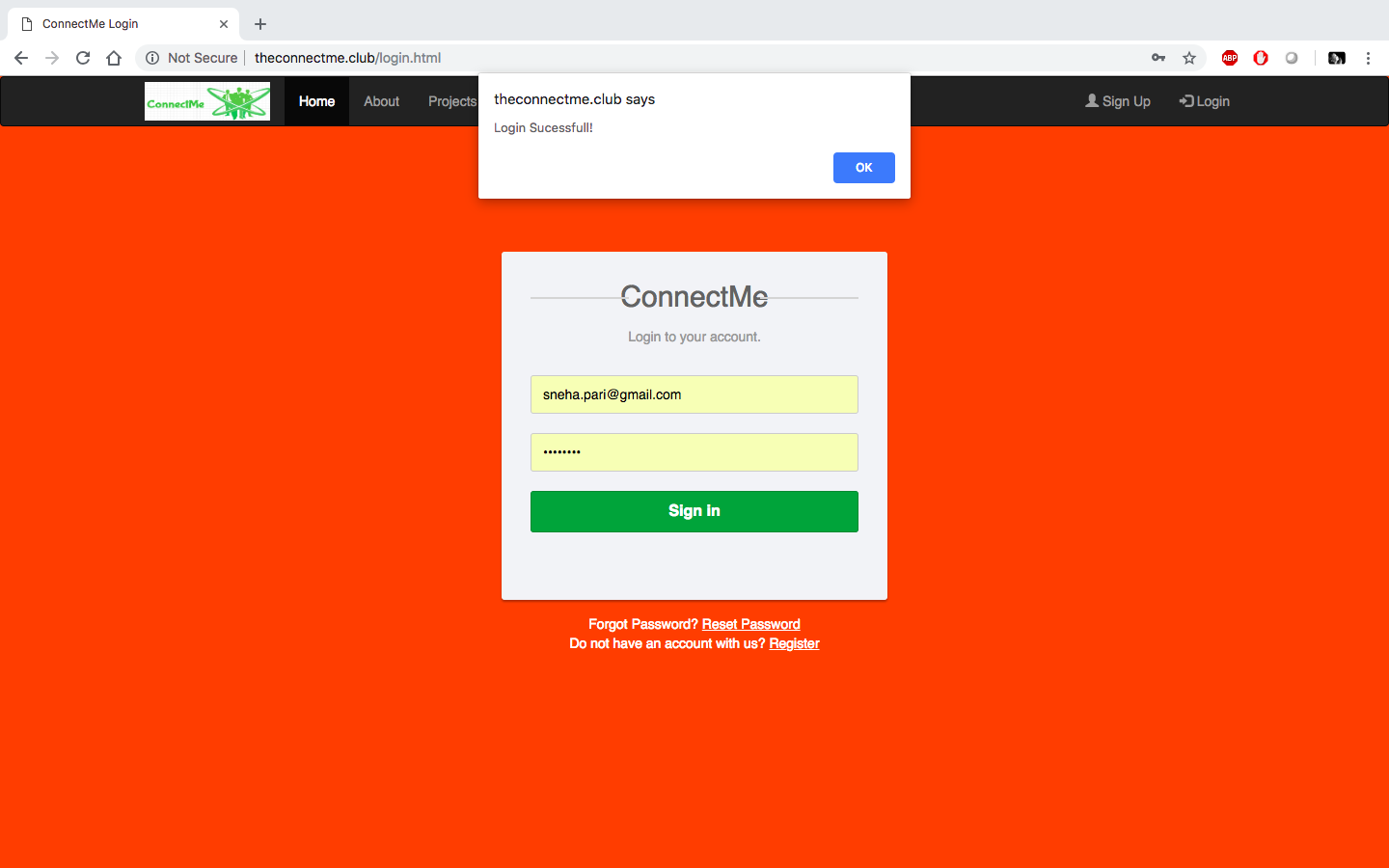
SignUp Screen







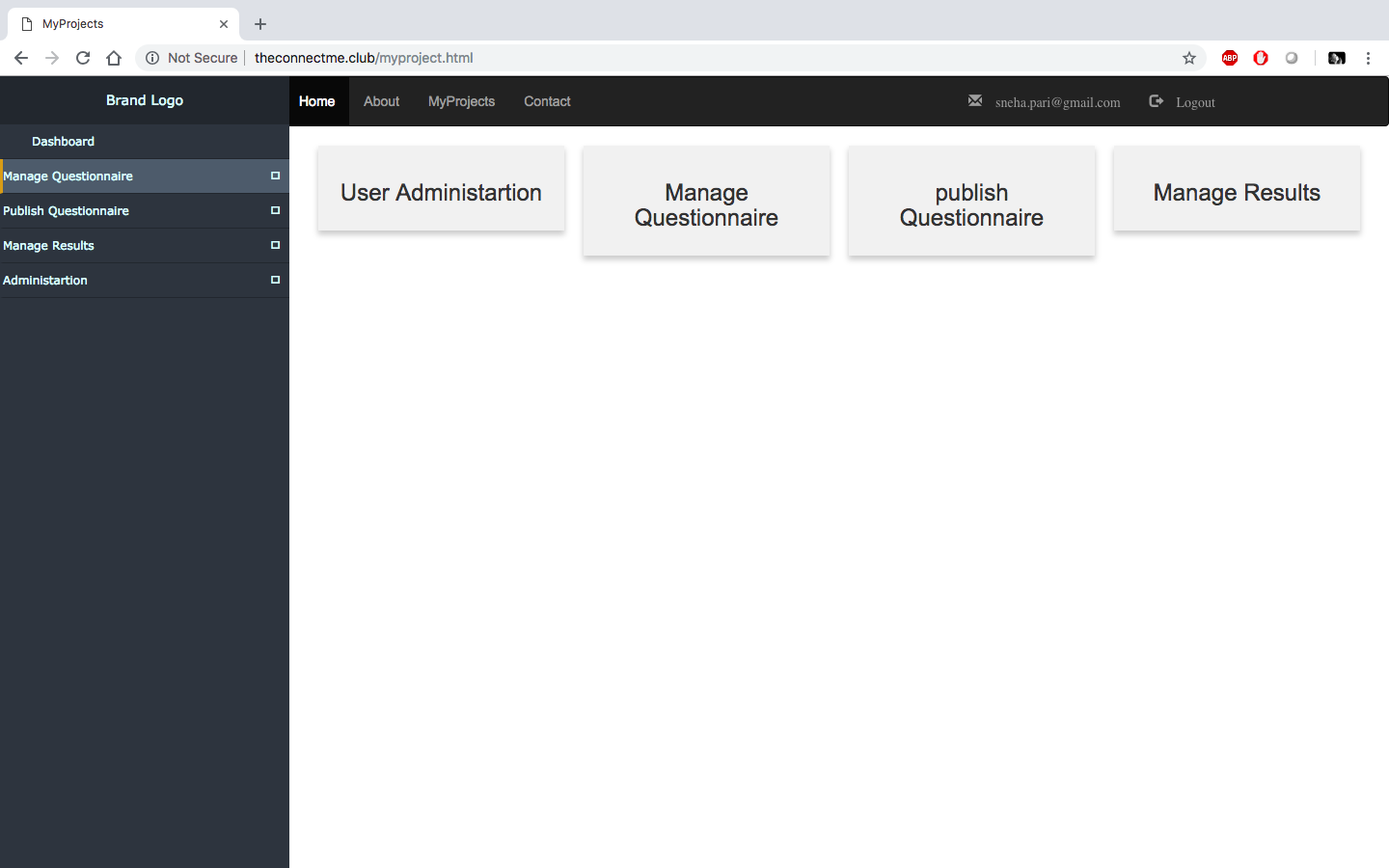




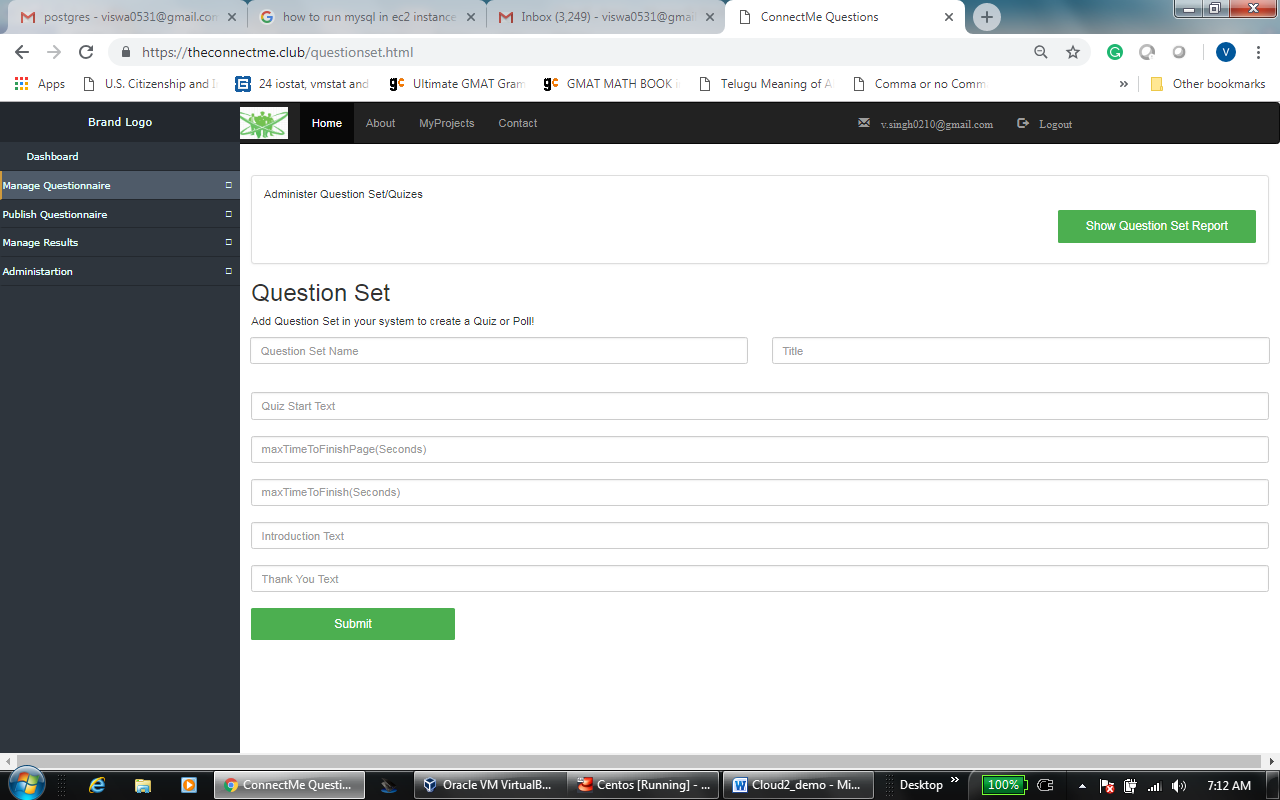
Logged In User Home screen:



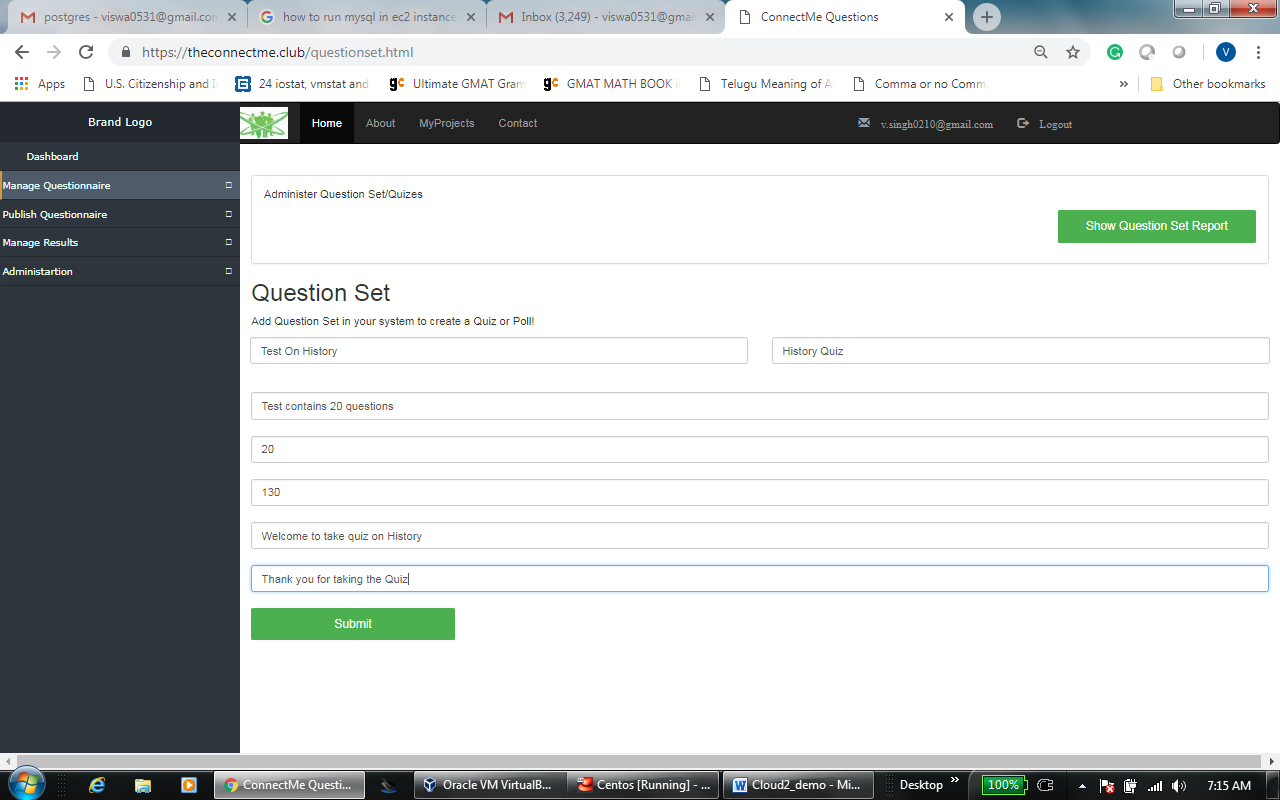
MyProjects Screen

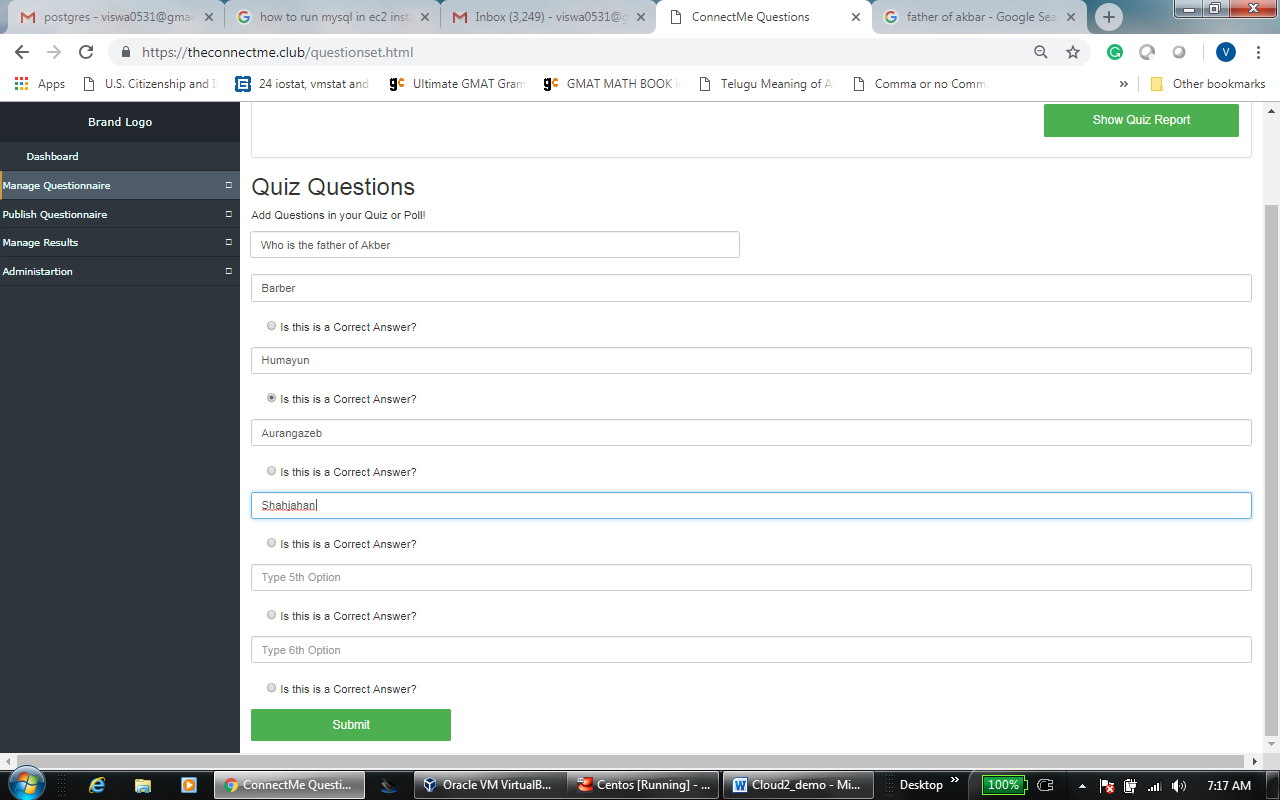


Manage Questionnaire

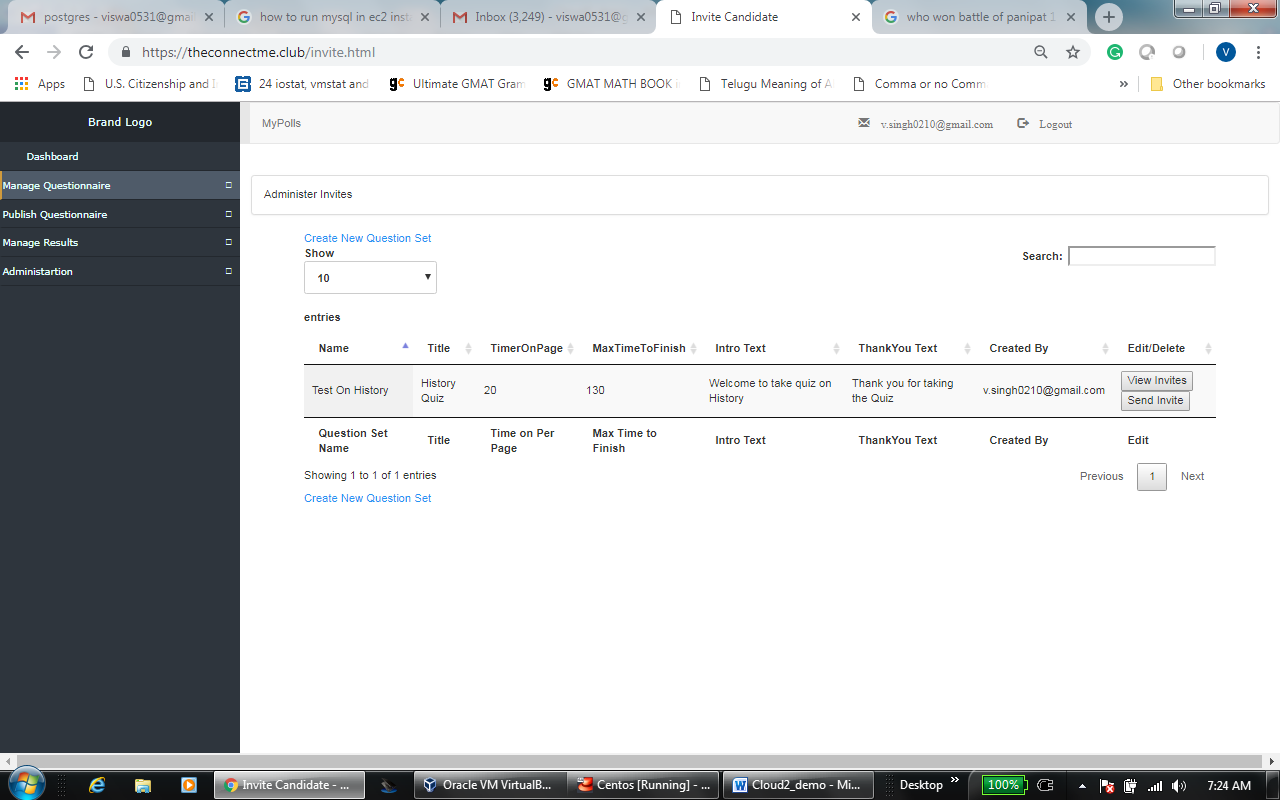


Create Question Set

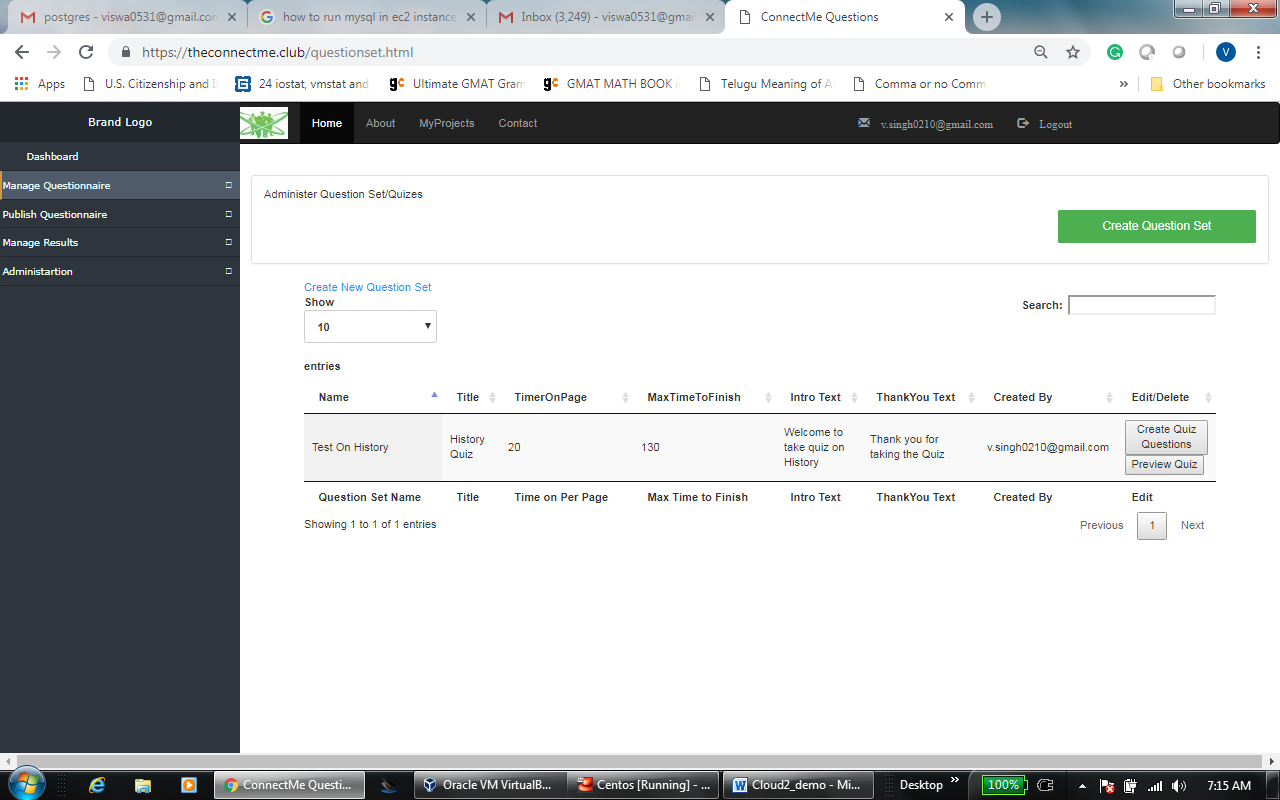




Published Quiz



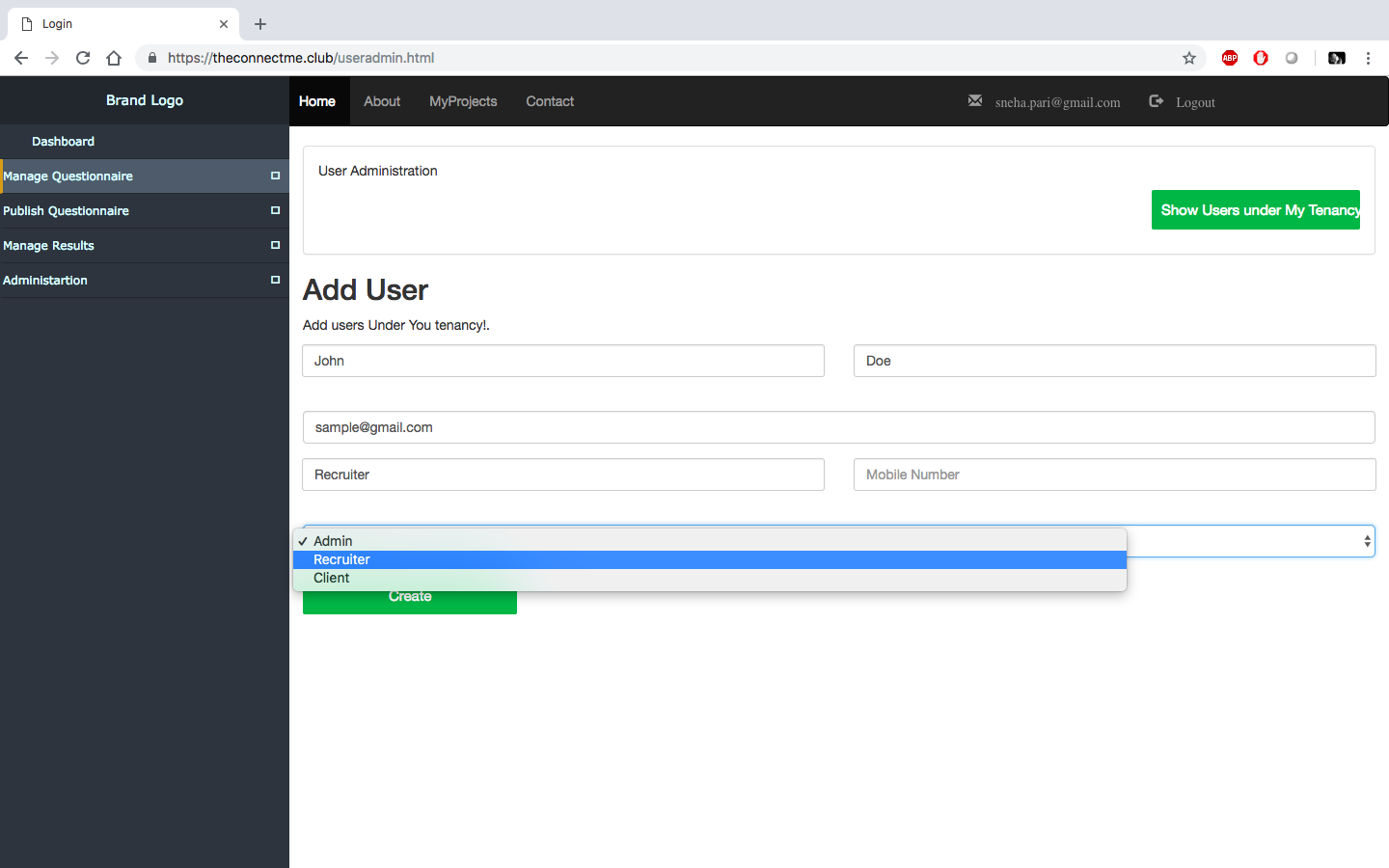
Publish Questionnaire

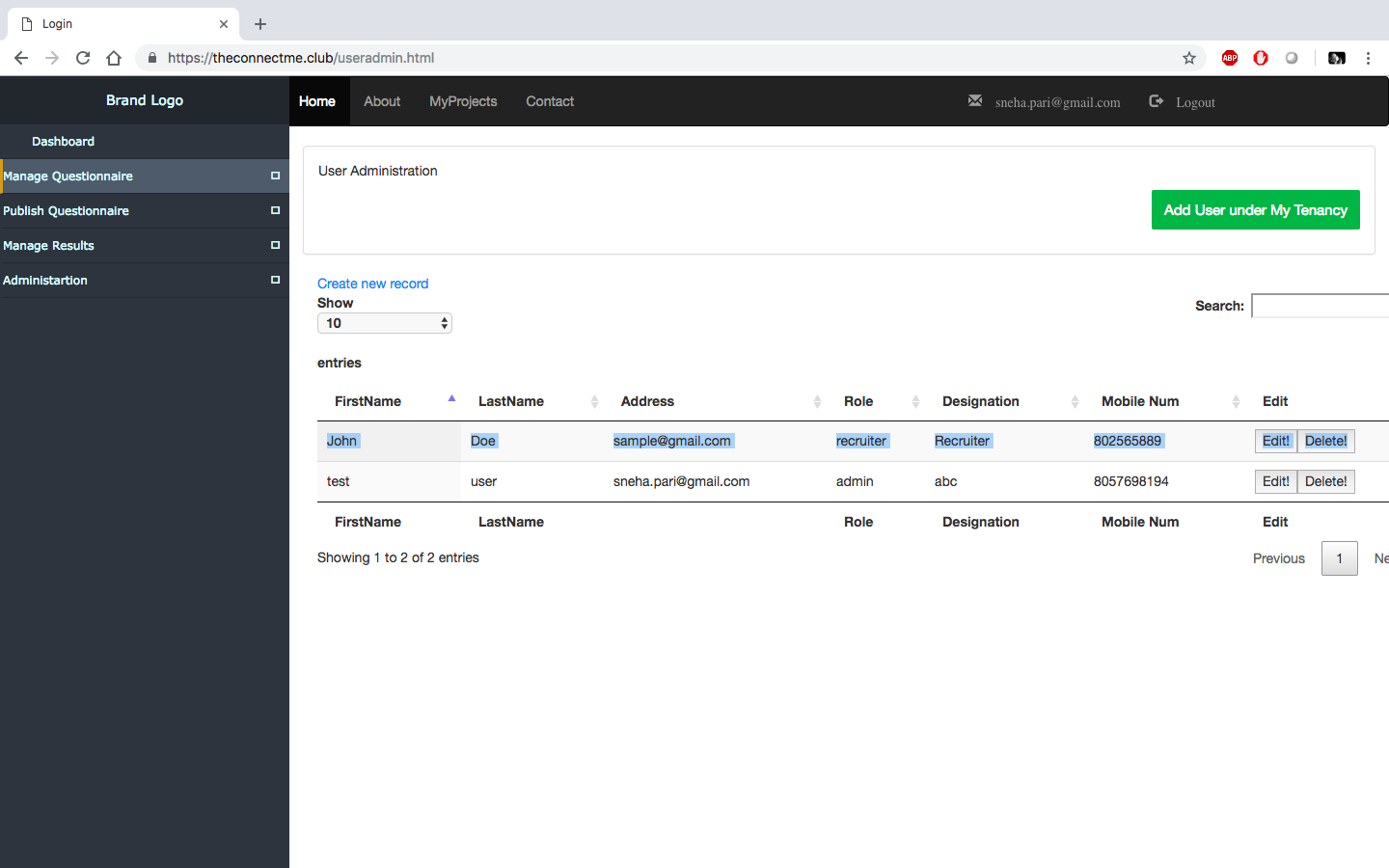


Manage Results

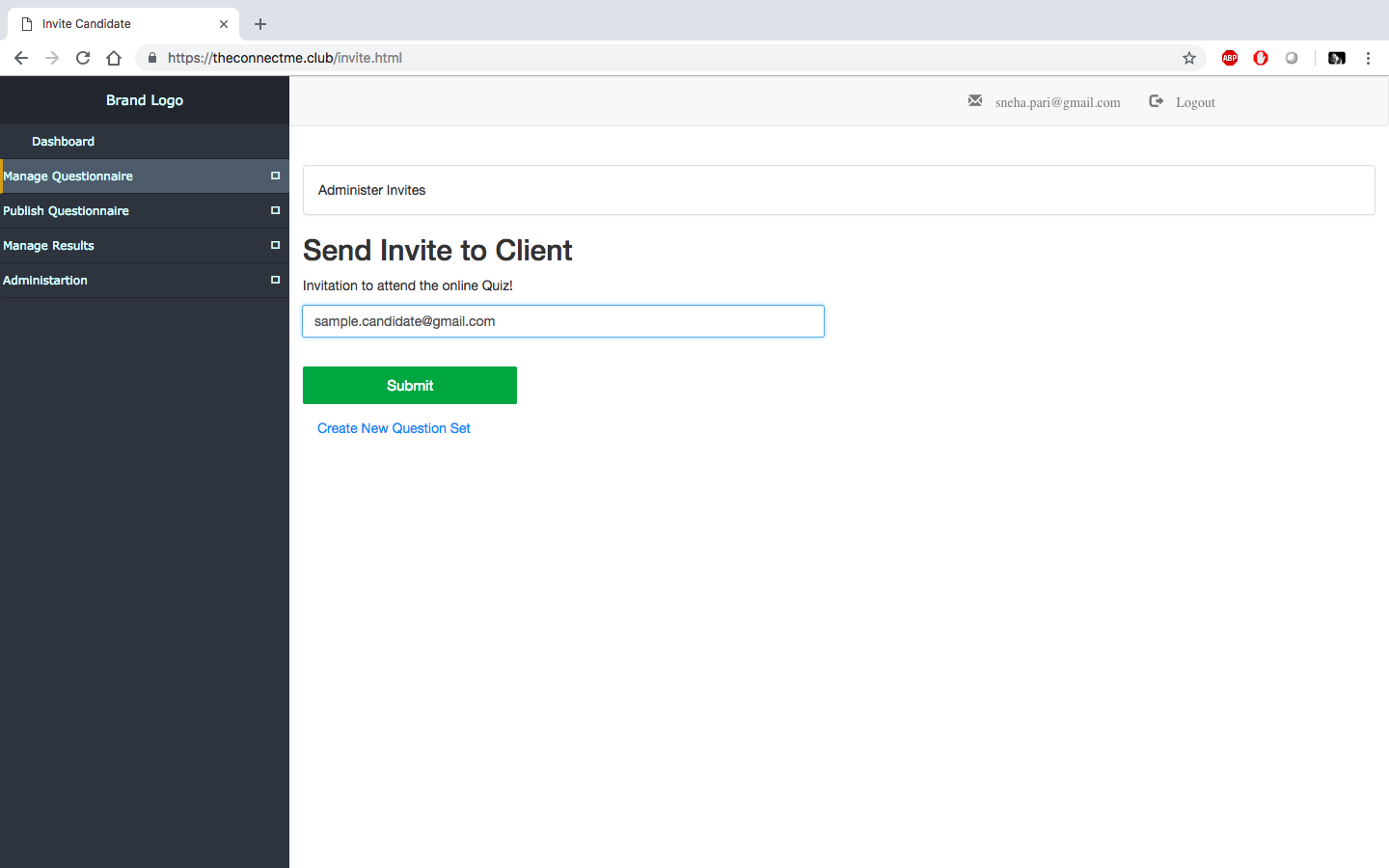
Administration

Add User Under My Tenancy





Send Invite to client



AWS Machine learning:



**Pre-requisite Set Up**

Resources to be configured on AWS:

1. EC2, Auto Scale Group,
2. ELB
3. Route 53
4. Polly
5. Amazon Machine Learning
6. RDS
7. Lambda
8. CloudWatch
9. CloudTrail
10. SNS

Softwares to download locally:

1. Server Side: Python, Flask, AWS Python SDK,
2. Client Side: HTML5, CSS,Ajax, Bootstrap, Java Script
3. Database: RDS MySQL
4. Web Server: NGINX

**Deployment Instructions on Local**

1.Install CentOS and NGINX

2. Clone the git repository

1. Copy the NGINX config to /etc/nginx/ngix.conf file from template/ngix
2. Create symbolic like to /usr/share/nginx/html to <user ws>/template/html

3. Create python virtual environment for python 2.7

3.Install Docker : Check and make sure docker service is running.

4. Create SQL docker image

1. docker pull mysql
2. mkdir -p /u01/mysql-db

5. Create the users and MySql database

1. docker run --name mysql-inst -e MYSQL\_DATABASE=mysql\_prod -e
2. MYSQL\_USER=prod -e MYSQL\_PASSWORD=welcome1 -e
3. MYSQL\_ROOT\_PASSWORD=welcome1 -p 3306:3306 -v /u01/mysqldb:/var/lib/mysql -d mysql:latest --character-set-server=utf8mb4 --collation-server=utf8mb4\_unicode\_ci
4. Jump into docker and allow permission for the users
5. docker exec -i -t mysql-inst /bin/bash
6. ALTER USER 'root' IDENTIFIED WITH mysql\_native\_password BY 'welcome1';
7. ALTER USER 'prod' IDENTIFIED WITH mysql\_native\_password BY 'welcome1';

7. Connect with the user to the sql

mysql -uroot -p -h 127.0.0.1

8. Create SQL tables. Copy past the commands from sql/create\_tables.py file.

9. Run python application

  python run.py

1. start browsing 127.0.0.1 and create admin user and recruiter user.