Cloud Computing Project

Digital Notice Board

Group 19

Ravi Rajpurohit (1002079916)

Vedanti Ambulkar (1001829121)

Priyank Gupta (1001629364)

Introduction

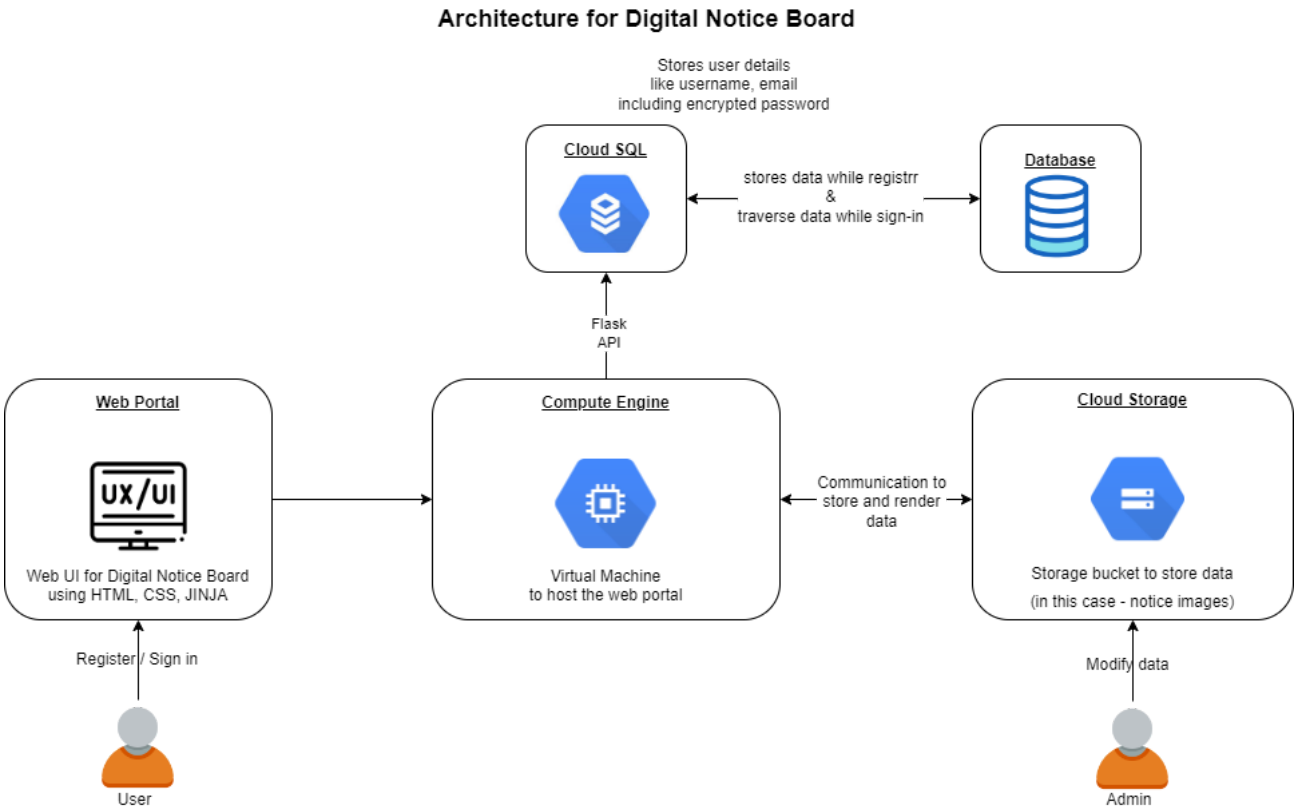
A notice board is a platform where people can be notified about upcoming/happening events. In each department at the University of Texas at Arlington, there are notice boards that contain the respective departments' notices. For instance, the linguistic department has put up a notice for a language study in the linguistic department. Now, in order to be updated with all such announcements, a student needs to be physically present at the notice boards.

This brought us the idea of a digital notice board. The idea is to create a web portal and curate all the notices in one place. This enables all the users to easily access information. The notices can be sorted by department, or upload time. This system eliminates the problems discussed above and helps students save time. It also helps in smoothening the notice board management.

Components Used

- Python:
    - Jinja: python framework used to create UI elements and validation for the web application
    - Flask: python framework used to create API for the application to communicate with the cloud
    - HTML: used to create web UI
    - CSS: used to style the web UI
- Compute Engine:
    - A virtual machine instance used to host the application on a remote server. This makes the application accessible over the internet allowing users to use the service.
- Cloud SQL:
    - A relational database management service used to store user information - name, username, email, and encrypted password.
    - A MySql database instance stores the user information while the user registration and validates it while the user signs in.
- Cloud Storage:
    - A google service used to store objects. For this project, it is being used to store the notice images that are uploaded using the digital notice board service.

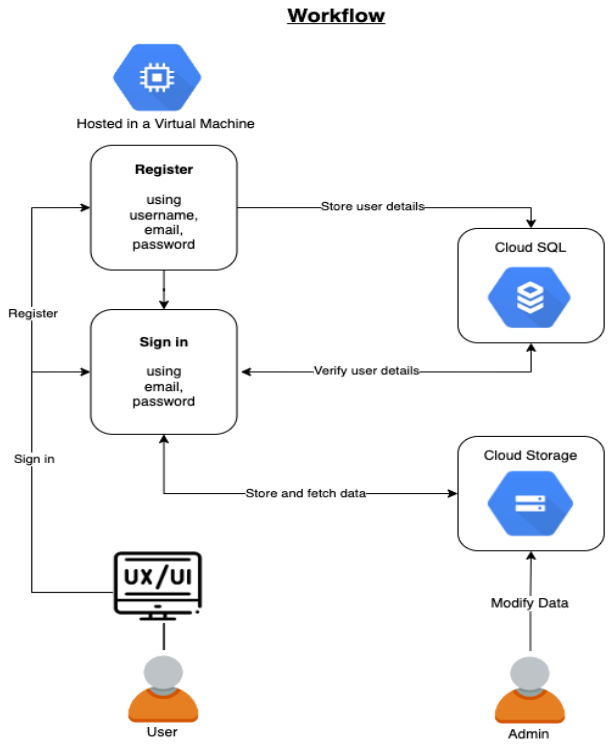
Architecture



The architecture of the application shown in the above figure displays how the components are connected and outline the data flow.

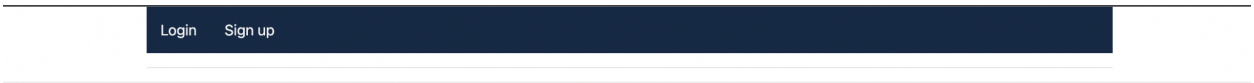
- Users will interact with Web Portal's user interface which is deployed on a virtual machine instance using Compute Engine.
- The User's data - name, username, email, and encrypted password are stored inside the Cloud SQL database instance. This instance is a MySQL-based database. It flows the data in and out while user registration and signing in respectively.
- Cloud Storage instance is used to store the object type data collected by the application. The users will be uploading images that need to be stored somewhere for storage and rendering. Cloud storage bucket is a cost-efficient and responsive method to store such data in an object model.

Workflow

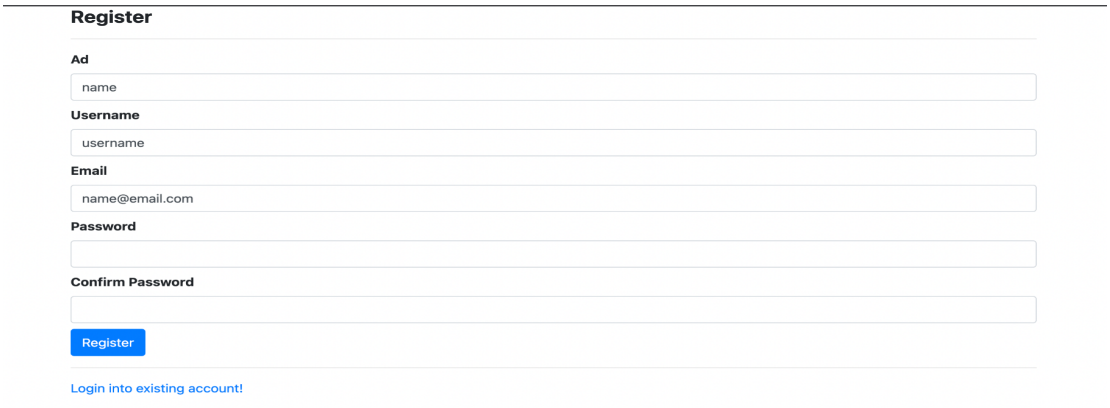


The following steps explain the workflow of the application from the user's perspective in detail.

- The application starts with an option for the user to register or sign in if already registered.

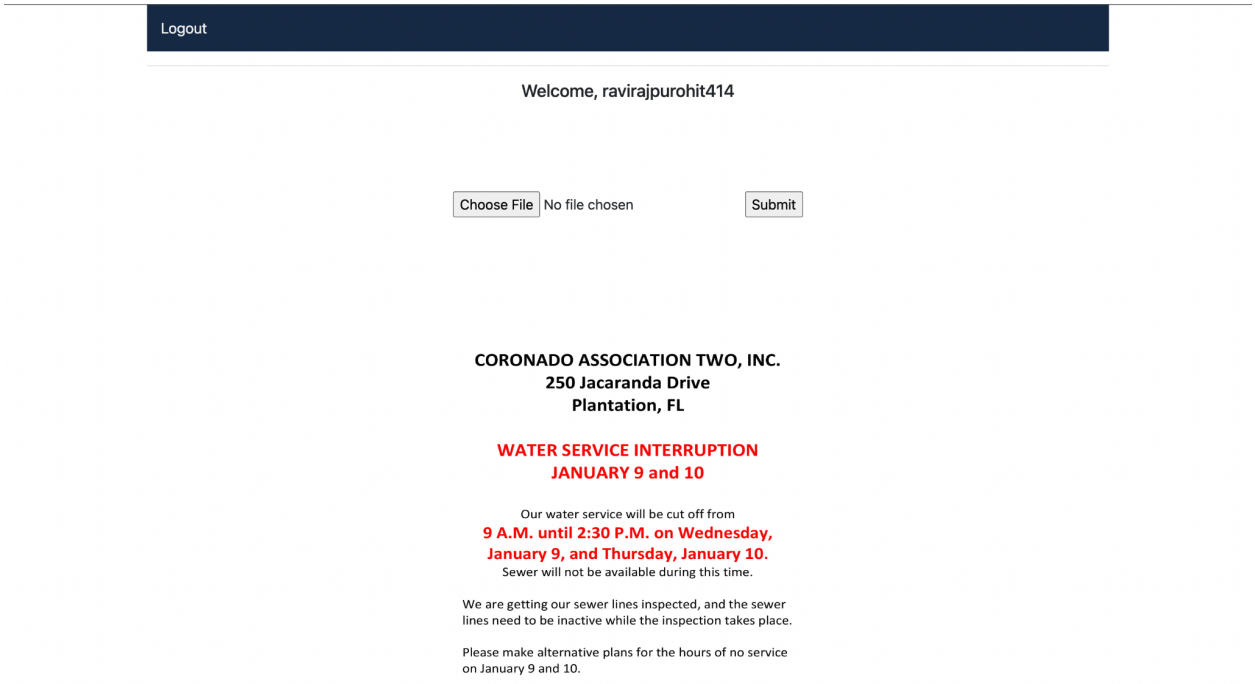


- The register option gives the user fields to enter information including - name, username, email, and password.



- Upon registering, the password is encrypted using SHA256 encryption and all the fields are stored in a MySQL database created from the Cloud SQL instance.
- If the user has already registered, he/she can log in instead.

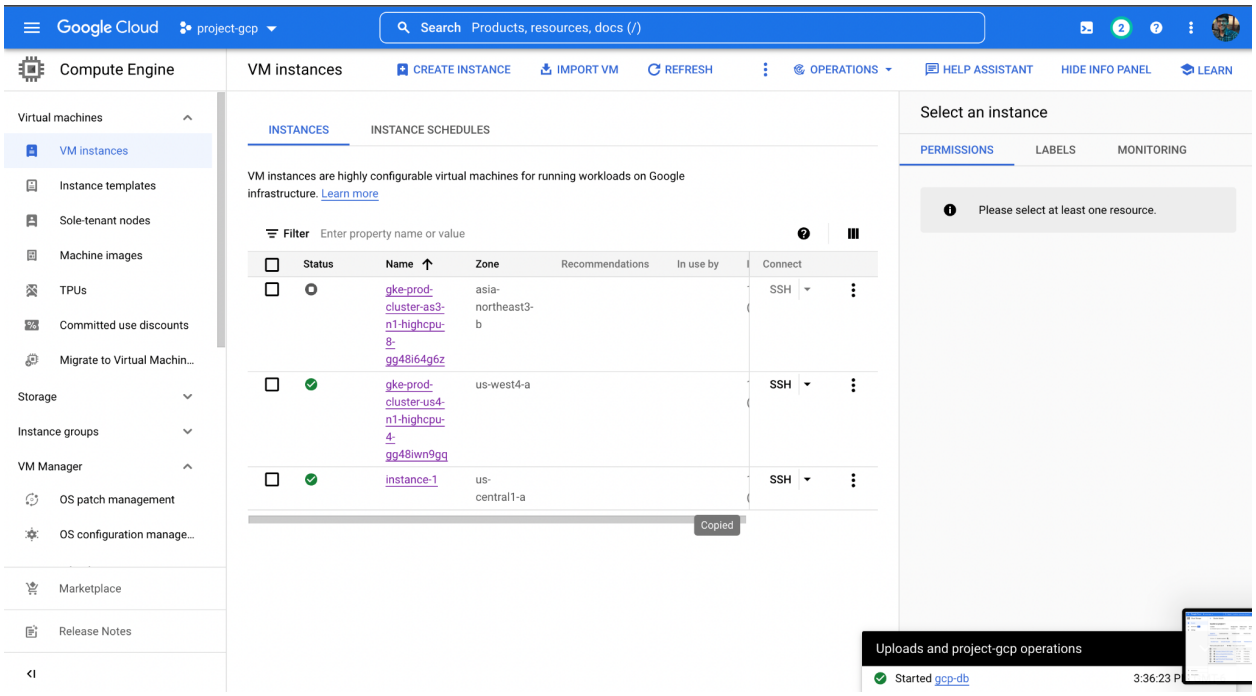
- After user authentication, once the user logs in they can interact with the application. The web user interface is presented.



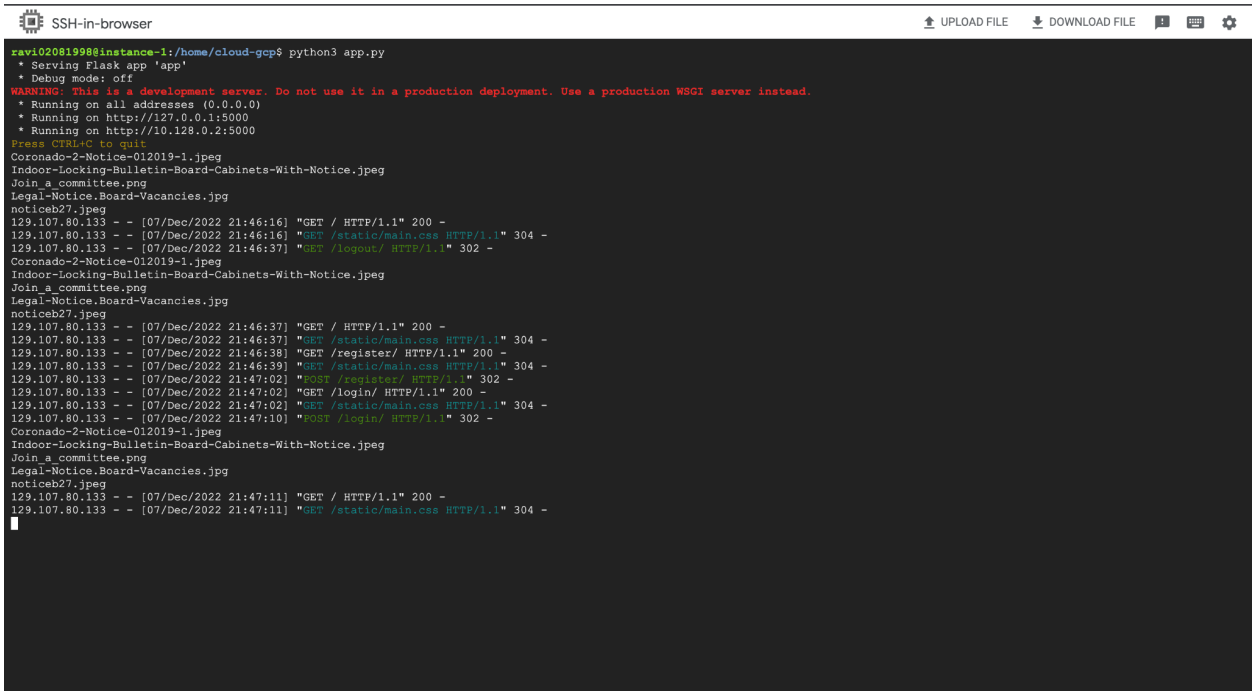
- The user can now, look at the current notices on display and also can upload a notice if desired using the file upload mechanism implemented in the web portal.

The following steps explain the application's workflow from the backend.

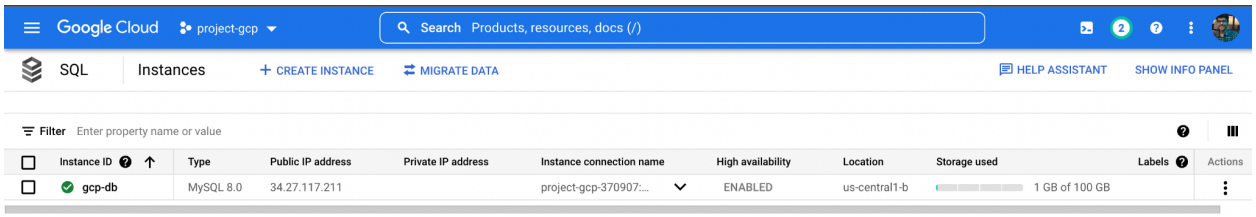
- The first step is to start the virtual machine instance in which the application is hosted. In the image below, instance-1 hosts the digital notice board application.



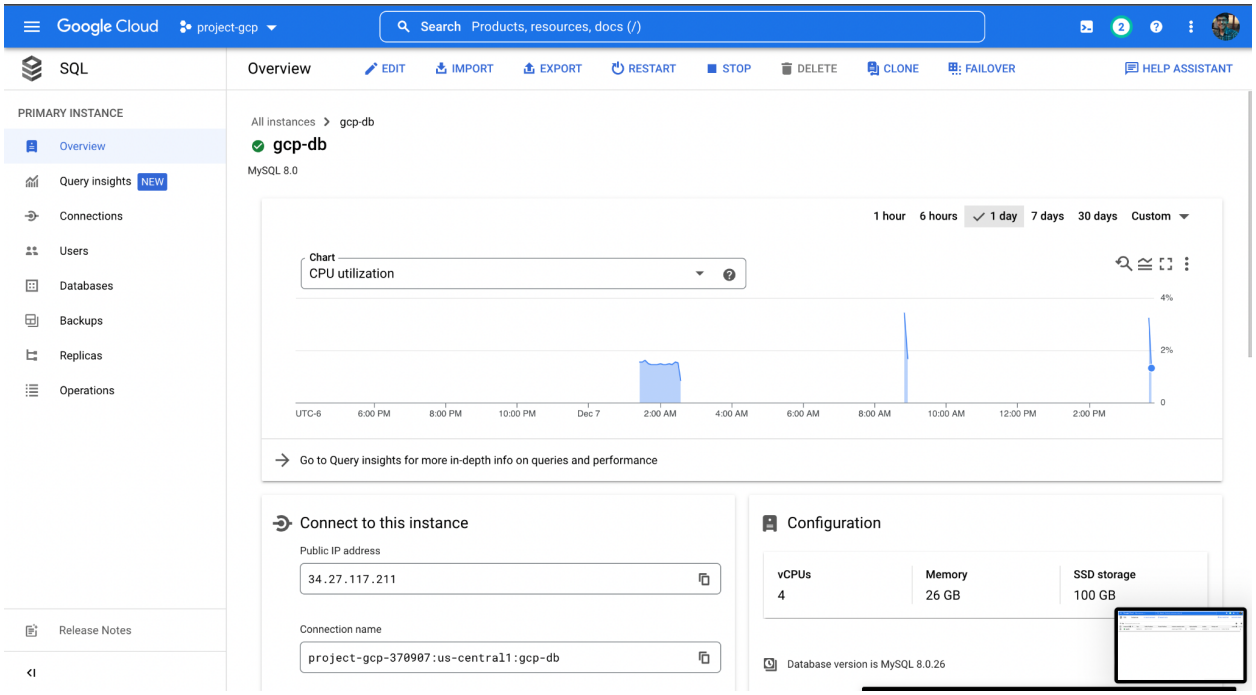
- After starting the VM instance, we need to run the flask application to make it accessible to all the users.



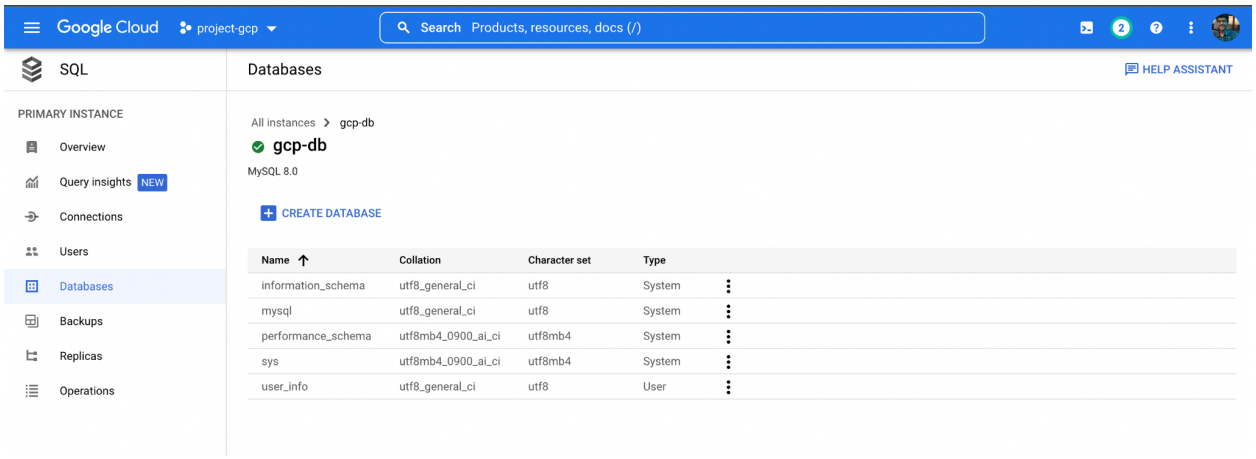
- Now that the application is available to the users, they will register and sign in. To accommodate this, we need a Cloud SQL server instance.



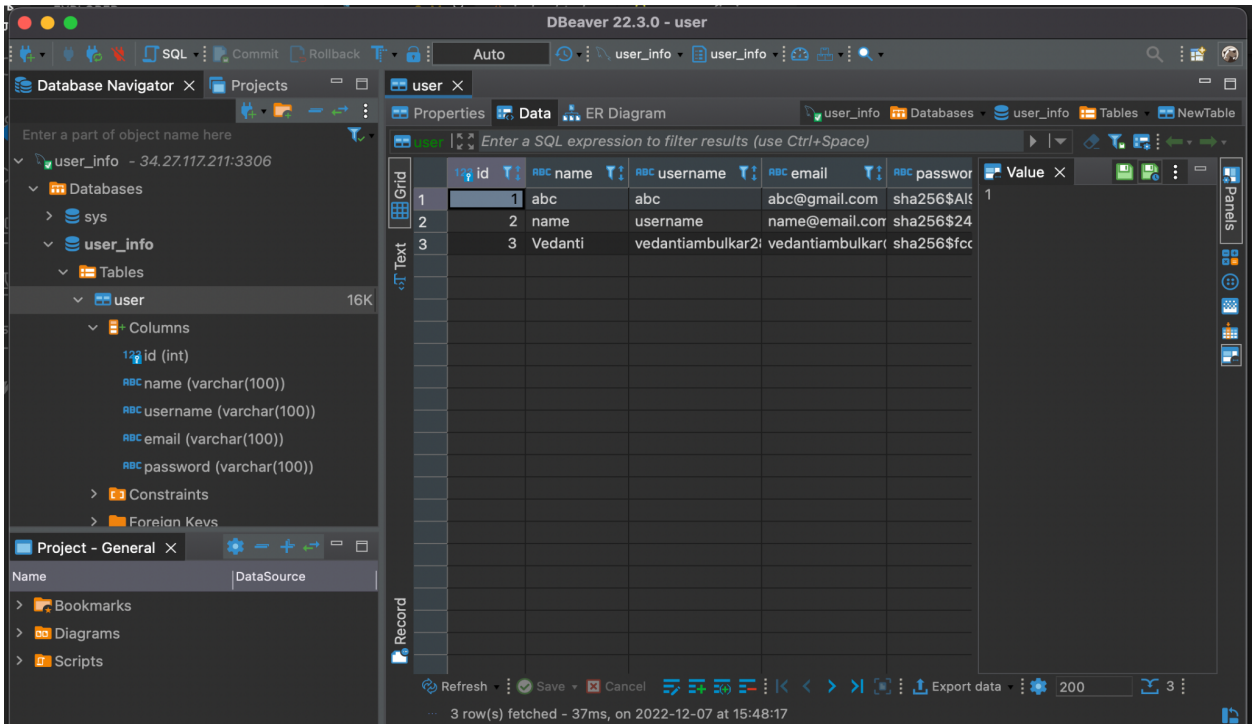
- The details of the cloud SQL instance are mentioned below.



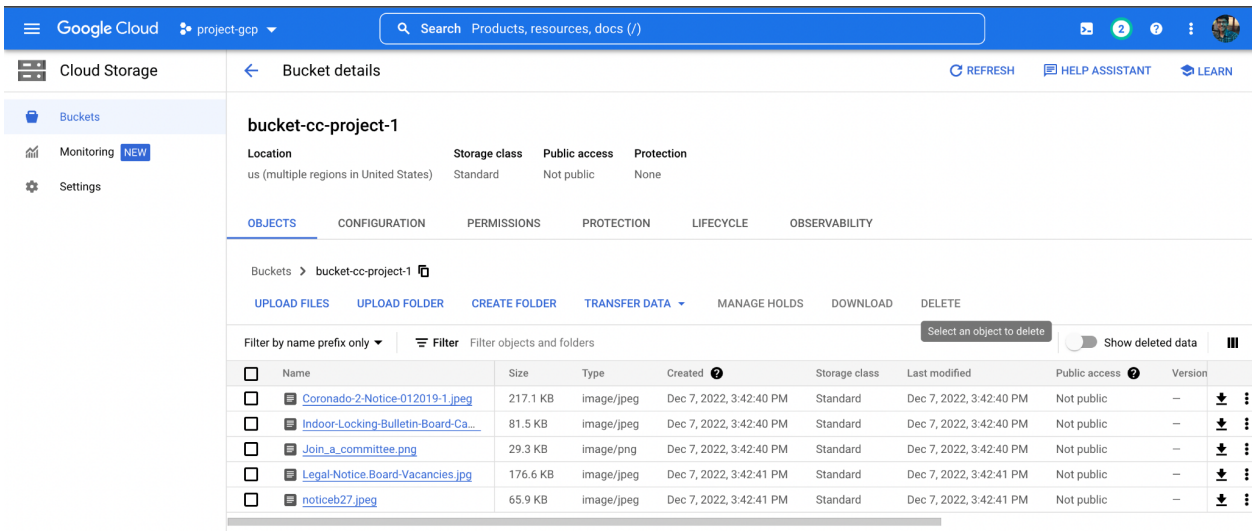
- A MySQL table is created to store the incoming data from user registration. This table also verifies the data when the users log in.



- When the user register, the information is stored using Flask API and is reflected in the database table shown below. As mentioned earlier, the password is encrypted using SHA256 encryption.



- After signing in, the user will see the web portal user interface and the current notices will be displayed. Users can also upload notices. These notices are stored in the Cloud Storage bucket.



- Flask APIs are called in order to upload notice images to the storage and retrieve them when the user signs in.

Future Scope

Currently, the application is a simple web portal using cloud computing services. With some enhancements, it can be a highly sophisticated professional application.

We plan to improve the application soon for the following aspects.

- Add a “Report” feature to provide users with the ability to report bad posts
- Automatically detect the expiry date for a notice in the image using text analysis machine learning technique and archive the expired notices from the portal
- Provide an interface to sort notices by date posted or expiry date
- Build a sophisticated admin control user interface for organization login so that multiple organizations can use this service

Individual Contribution

Ravi Rajpurohit, Vedanti Ambulkar, and Priyank Gupta worked on this project as a team. All of us were an active part of all the components related to this project. We discussed ideas and implementation on weekly basis and brought this project to a logical conclusion together. To mention, the following was the contribution made by individuals.

- **Ravi Rajpurohit** contributed to the project idea, architecture, web user interface, API link setup between the cloud storage bucket and the application, cloud storage bucket setup, project proposal, mid-term evaluation, and final report writing
- **Vedanti Ambulkar** contributed to the web user interface, API setup for user register and sign-in, Cloud SQL database configuration setup, project proposal, and project presentation
- **Priyank Gupta** contributed to the virtual machine instance setup, architecture documentation, password encryption, testing of the application, mid-term evaluation, and final report writing

Source Code

https://github.com/ravirajpurohit414/cloud-gcp

The source code for the web UI is uploaded in a public GitHub repository. The virtual machine instance pulls code from here and reflects the changes in the user interface or API handling logic.