

CSE 546 — Fun-A-Holics

Fun Activities Creation and Participation System - Using GCP

Akshay Kumar Gunda (1217179379)

Kusumakumari Vanteru (1217031218)

Tejaswi Paruchuri (1213268054)

1. Introduction

In the time of Zoom meetings and Work from Home environment, searching for fun recreational activities (to do nearby with safety precautions or at home) to uplift the mood has become a necessity and we want to build a scalable cloud solution (using PAAS offered by google cloud platform GCP) to address this problem. This application will provide an interactive Web-Interface to help users find recreational activities (indoor and outdoor) based on criteria such as the number of group members, minimum age/maximum age information, budget, location preferences, category of activities, status of event etc. All the effort of filtering the virtual events or outdoor activities suiting their budget and other requirements will be taken care of by the Fun-A-Holics app. This app will help users find recurring events (like online yoga, gaming) created by us even on days when user created events are limited. In case of outdoor events, we have included COVID test results as a prerequisite. Only those who successfully submit the negative covid testing results or covid vaccination report at the start of the event are eligible to attend the event.

2. Background

We have used the following Technologies while developing the Fun-A-Holics application.

Technologies:

- Front-end: HTML5, CSS, Bootstrap
- Back-end: Python Flask framework
- Google cloud platform PAAS services:
 - Google App Engine
 - Google Cloud Scheduler
 - Google Cloud SQL
- APIs:
 - Google Mail API
- Database:
 - MySQL

In this pandemic situation, where many people are far from their friends and family it has been very difficult for people to do recreational activities. Fun-A-Holics provides a platform to find like minded people and engage together in events virtually or outdoors and have fun.

This application is different from existing apps like MeetUp and Backpackers because they mostly focus on outdoor activities without any safety precautions and have very limited options for indoor fun activities. This application will help users find recurring events (like online yoga, gaming) created by us even on days when user created events are limited. Adding covid test results as a prerequisite for outdoor events along with providing varied filtering criteria makes our application different from the existing ones.

3. Design and Implementation

A detailed architecture of the Fun-A-Holics application that we have designed and implemented is as follows:

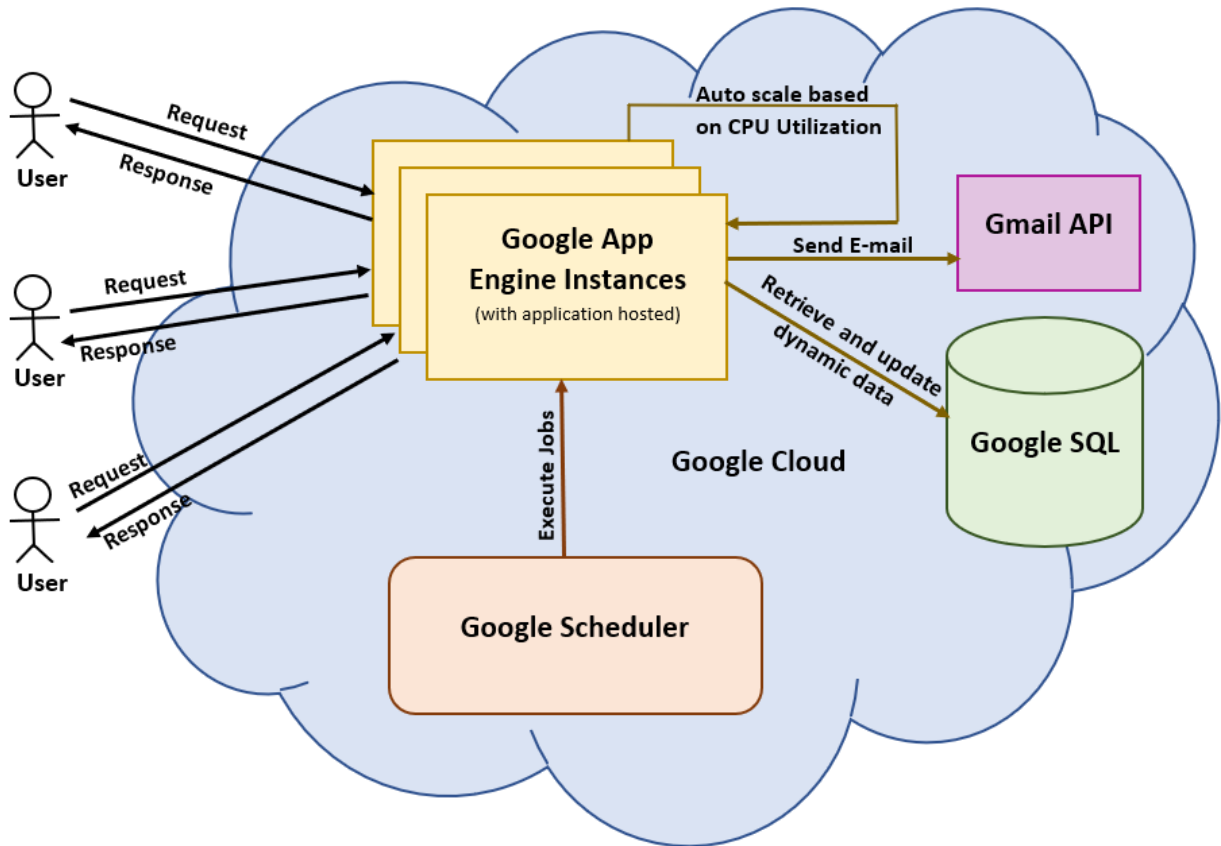


Fig 1: Architecture of the application Fun-A-Holics

The entire application is hosted and accessed using Google Cloud Platform (GCP), mainly Google App Engine (GAE). The major components of the system are:

- **GAE instances:** This is the main GCP service where our web-application is hosted. Here, we have deployed the web-app using the command 'gcloud app deploy' and used automatic scaling provided by the app engine platform. We have specified this in the app.yaml file, by using target cpu utilization as 0.5. So, if the average CPU utilization across all the instances crosses the 50% limit, then new instances will be scaled-up by the app engine automatically and scaled down after a cooling period of 60 secs. The main purpose of GAE is to host the application, receive requests from the users, auto-scale the application based on the target CPU utilization, execute jobs as directed by the Google Scheduler, manage DB operations and act like the central piece of the entire system.
- **Google Scheduler:** This is used to schedule CRON jobs calling our application api's from the Scheduler console. The main purpose of this part is to create daily, weekly events, update current status of events based on its start/end dates and delete cancelled or completed events, with the help of recurring jobs as scheduled. These jobs are executed at the specified times, by the Google Scheduler.
- **Google SQL:** We have used Google SQL relational database as our primary source for storing and retrieving users and events information. The Entity-Relationship (ER) diagram for our database is shown as below.

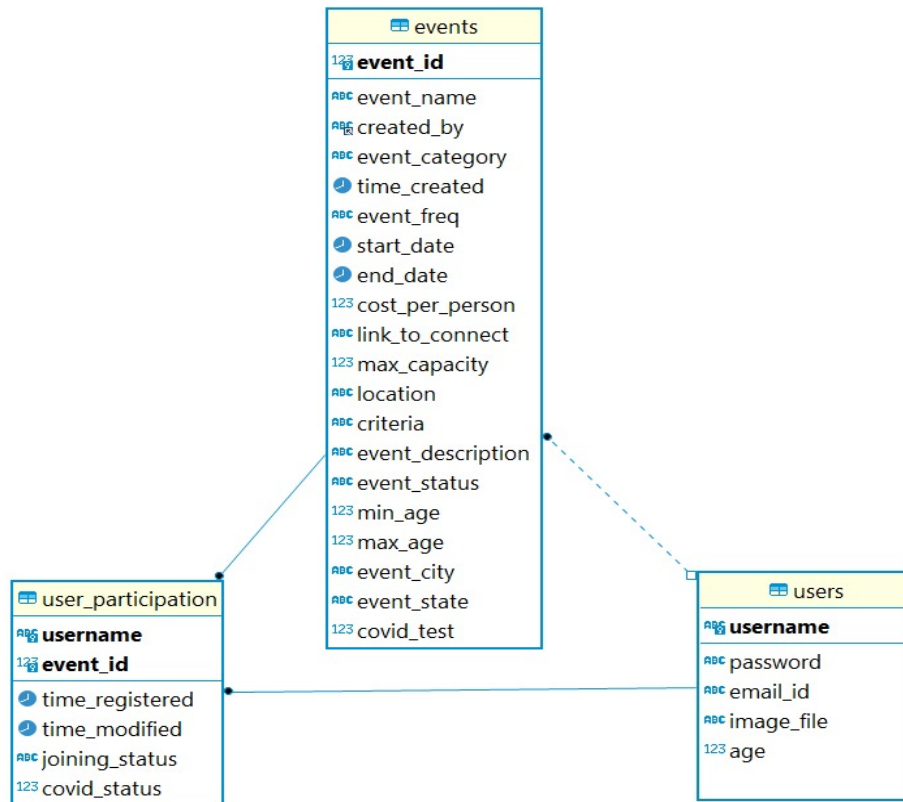


Fig 2: DataBase architecture used in project Fun-A-Holics

Here, we have used 3 tables - users, events and user_participation table. The 'users' table has all the basic user information like username, age, image file name, email id and password which are used by the application. The 'events' table contains the major chunk of data having the entire events' information like the date created, the category of the event , starting and ending dates, description, etc. This is the table used to display the events on the homepage of the application, and is continuously accessed to retrieve all the current active events. Finally, the user_participation table contains the details of the events registered by each of the users.

- **Gmail API:** This API is used to notify users when they deregister from a particular event or if any event gets cancelled. We have integrated our web-application with this api, to send emails to users based on the email-id which they have given in the login information.

These are the 4 main features of Google Cloud Platform that we have used to build our solution, mainly the Google App Engine's Platform-as-a-Service(PAAS) functionalities. Google cloud provides three scaling options, autoscaling, basic and manual scaling options. We implemented the automatic scaling feature for resource optimization. As all the services we are offering in this application are in the same application, multiple requests will increase the CPU Utilization which will become the bottleneck for this application. So excessive CPU utilization, throughput and concurrent requests are handled through our autoscaling solution[1].

Our application is a highly scalable solution that helps users to create and register for recreational activities during the pandemic situation. We have also considered safety measures like covid test, vaccination report as prerequisite for outdoor activities. This solves the situation of a cloud-based web-app for fun activities utilizing PaaS services offered by Google Cloud Platform.

Considering COVID situations where WFH has become normal and safety has become a concern, in order to have fun with safety measures, this app will help people to participate and create recreational activities (both indoor and outdoor) as per their interest. This is how it was significantly different.

4. Testing and evaluation

Below are the features we have implemented as a part of this project and the same were tested to check for successful implementation.

- Providing Registration/Login functionality for users to maintain their profiles.
- View upcoming activities based on category (like gaming, hiking.....)
- Provide a list of activities based on the filter criteria (like location preference,min age/max age, budget, category of activity, number of people joining...) requested by the user.
- Users can join the activity they are interested in.
- Users can create a new activity so other users can join.
- Display all the upcoming activities the user has joined.
- Cancel created activity and notify the users who joined it.
- Dropout from an activity the user has joined and notify the creator of Activity.
- Auto scaling based on CPU Utilization.

All the above mentioned functionalities can be tested using <http://cse546p2-309902.wn.r.appspot.com/>. Below are the screenshots showing the testing of functionalities and autoscaling.

- Providing Registration/Login functionality for users to maintain their profiles.

Fig 3: Screenshot showing the registration functionality

⚠ Not secure | cse546p2-309902.wn.r.appspot.com/login

Fun-A-Holics Home About

Login

Email

tparuchu@asu.edu

Password

....

☐ Remember Me

Log In

[Forgot Password?](#)

Need an Account? [Sign Up Now](#)


Fig 4: Screenshot showing the login functionality

On clicking on register or login on accessing the application and by filling the required details users will be successfully able to register and login themselves

- View upcoming activities based on category (like hiking, movies, art ...)

On selecting one of the categories and applying a filter shows the upcoming activities based on the start date of the selected category.


Fun-A-Holics Home About Login Register



Akshay 04/21/2021, 01:35 AM

Test Event


Test Event



Janet 04/25/2021, 09:00 AM

Day trip to Sedona


This is a day trip to Sedona devils bridge hike and other famous spots



Fun-A-Holics 04/28/2021, 07:00 AM

DayTrip to Sedona


This is a day trip to Sedona devils bridge hike and other famous spots



Thomas 04/30/2021, 12:00 PM

CamelBack mountain hike

Day trip to an exciting hike of the Camelback Mountain



Tejaswi 05/01/2021, 07:00 AM

Daily 'A'-Mountain Hike

This is the daily hiking event to 'A'-Mountain.

Filter Events

You can filter the events using the following fields.

Category

hiking

Criteria

select

Min age

0

Max age

60

Event Status

select

Max capacity

50

Event City

Event State

1 2

Fig 5: Showing events based on category hiking

- Provide a list of activities based on the filter criteria (like location preference,min age/max age, budget, category of activity, number of people joining...) requested by the user. Events are displayed based on the filter criteria selected by the user.

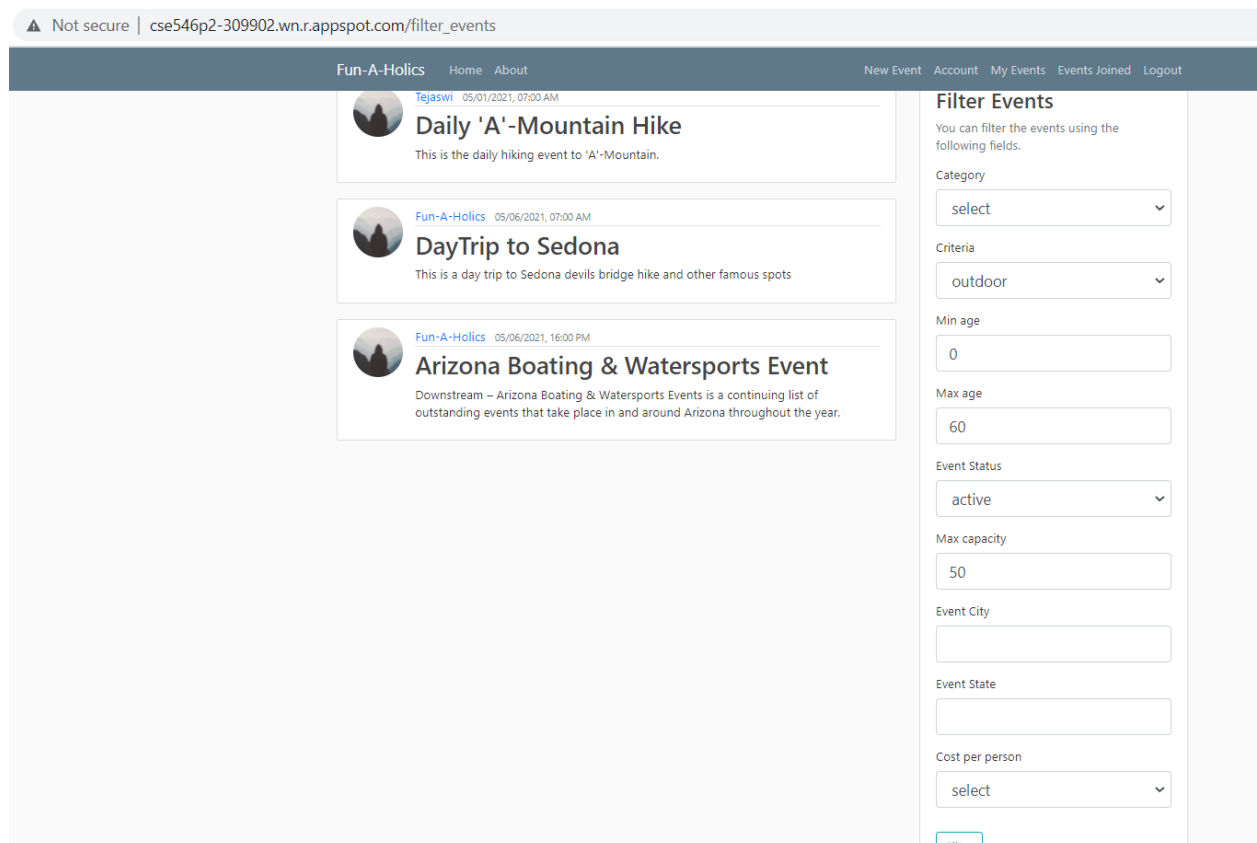


Fig 6: Showing the list of available events after applying filter

- Users can join the activity they are interested in if they qualify for the event.

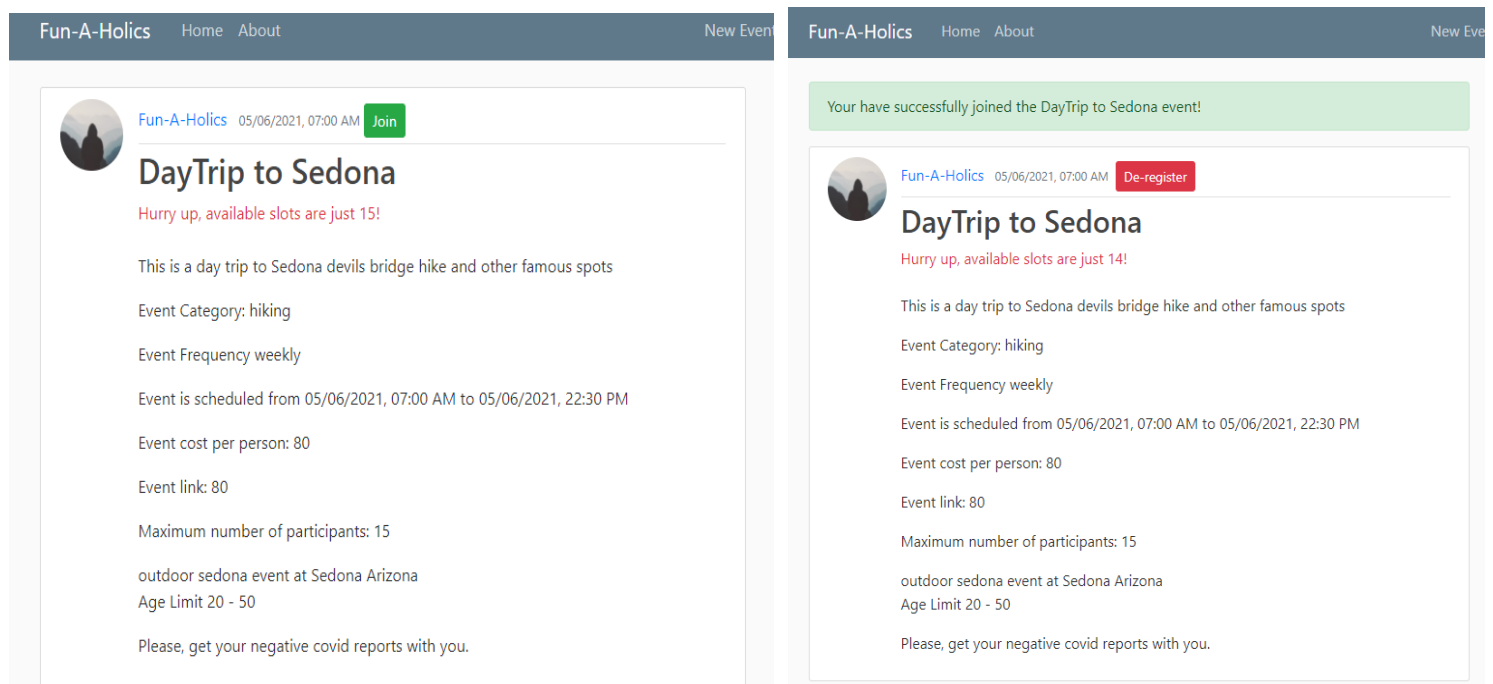


Fig 7: Screenshots showing the successful registration into a event

- Users can create a new event so other users can join.

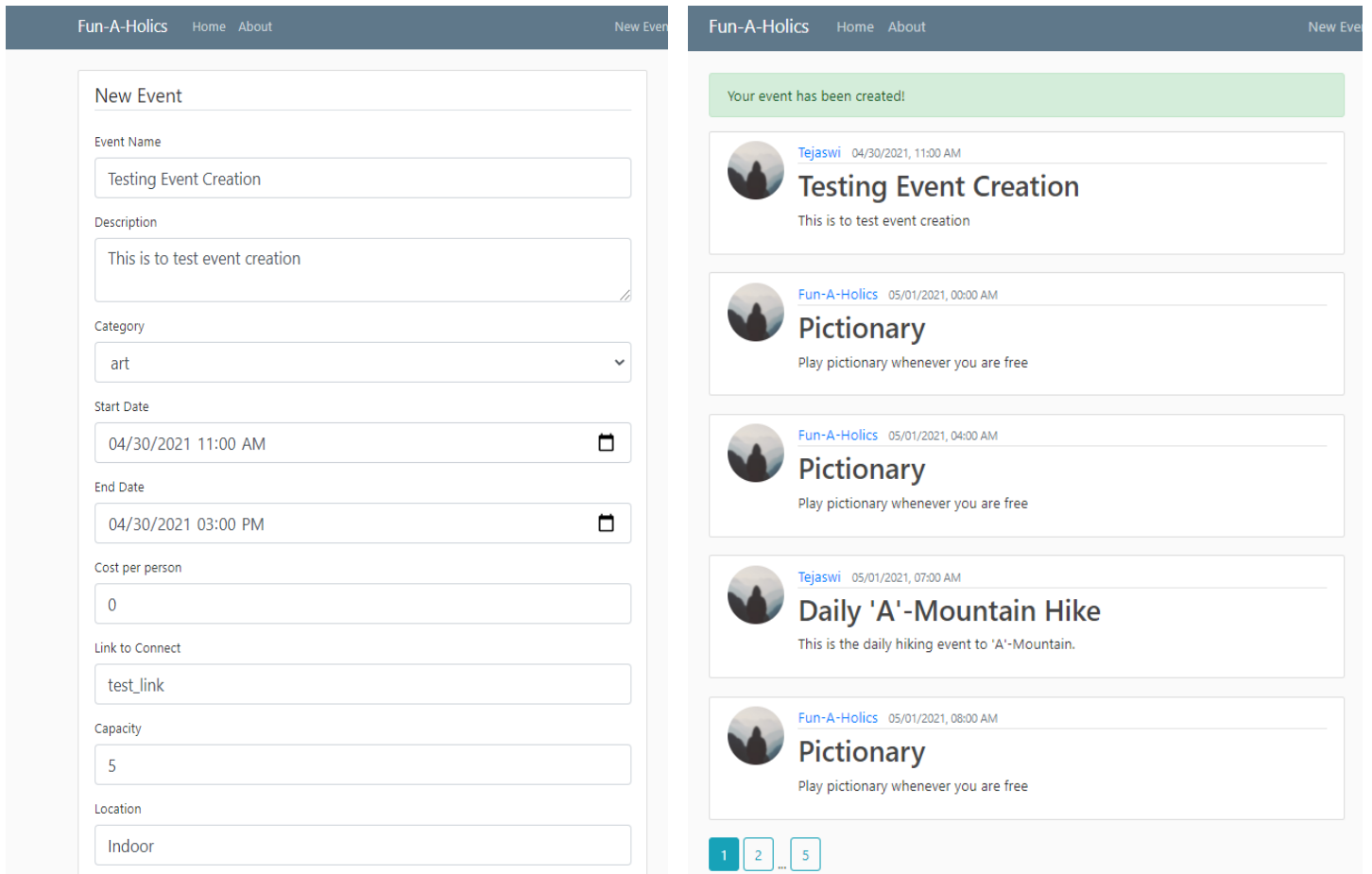


Fig 8: Screenshots showing the successful creation of a event

- Display all the upcoming activities the user has joined.

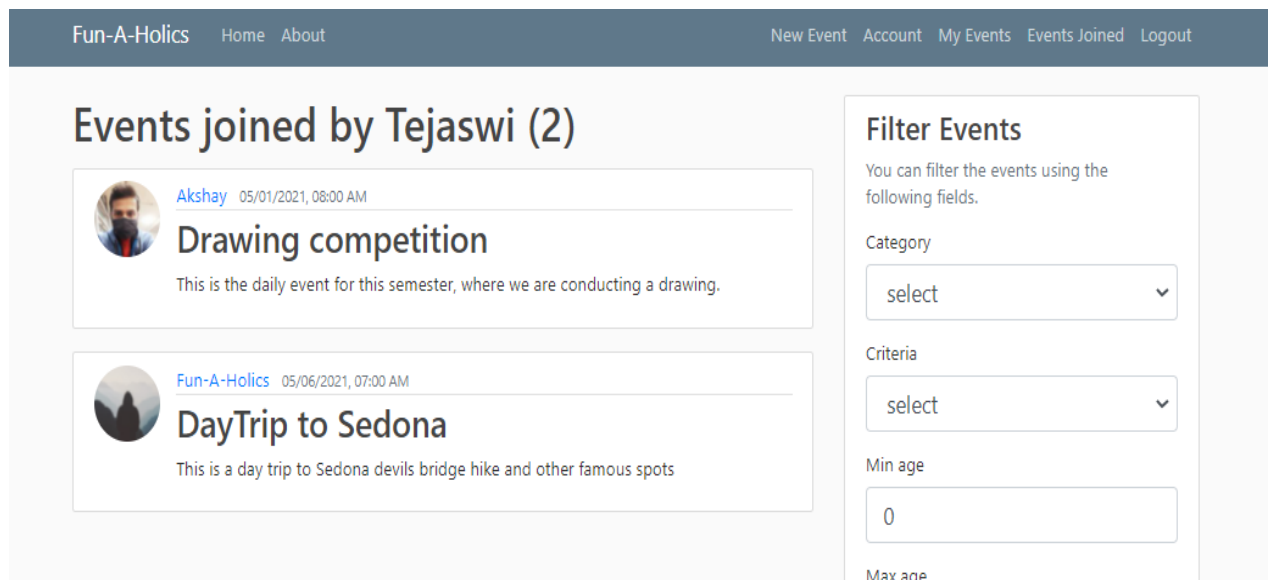


Fig 9: Screenshot showing events user has joined

- Cancel created activity and notify the users who joined it.

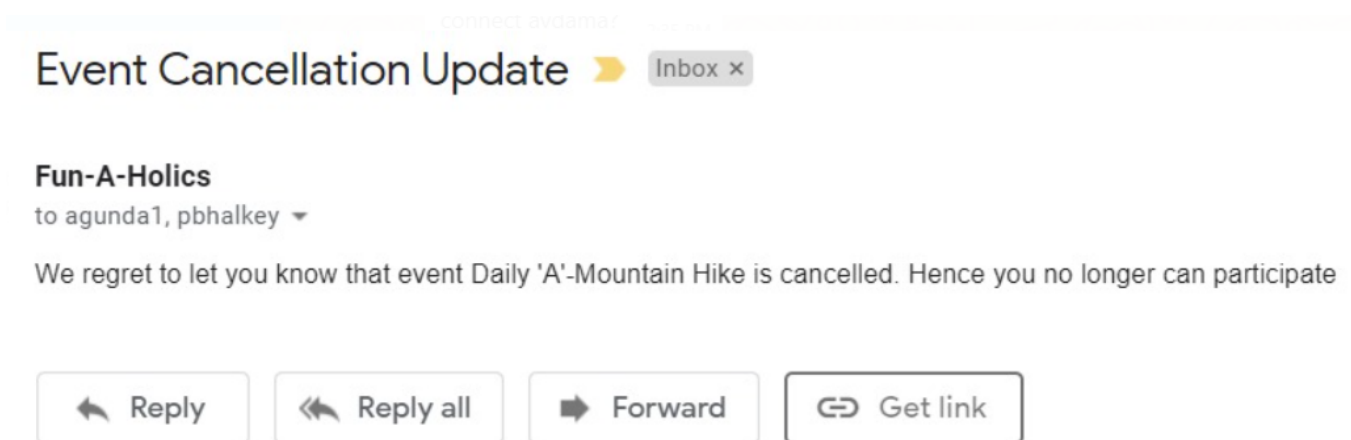
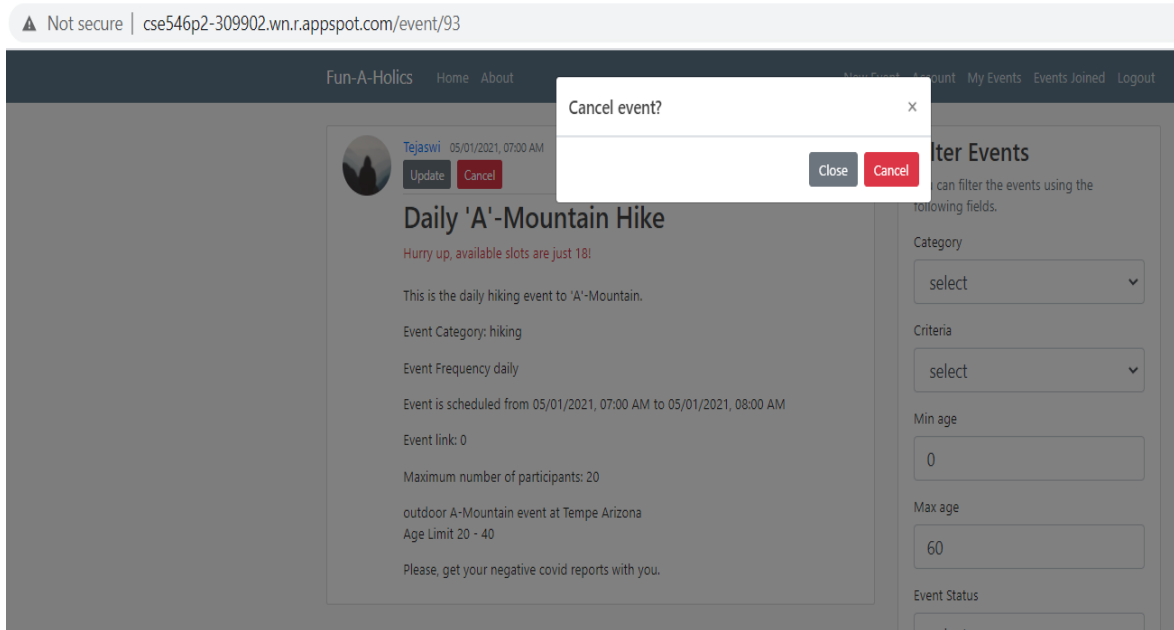
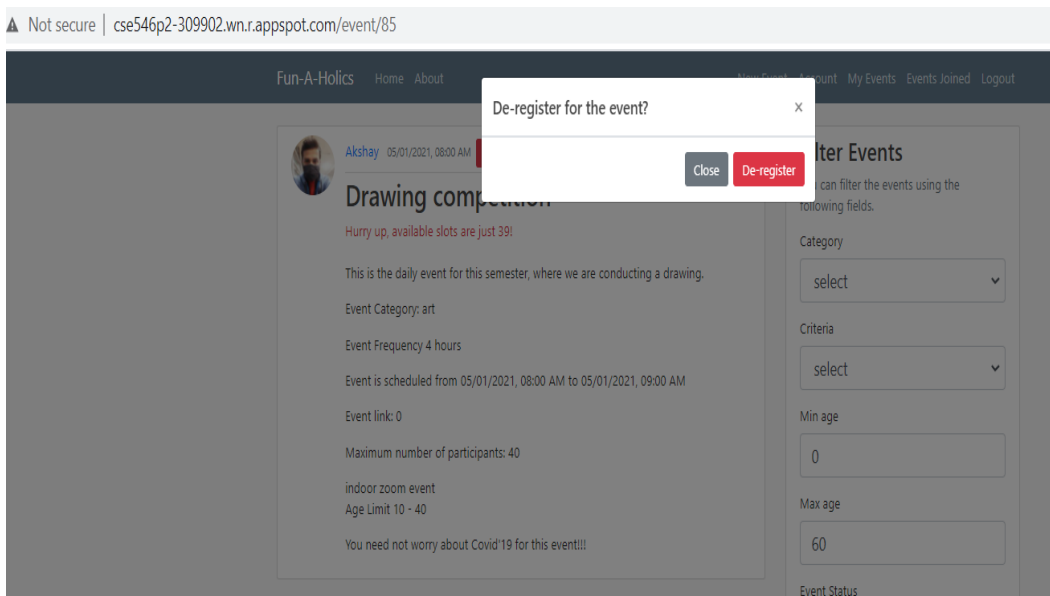


Fig 10: Screenshot showing event cancellation by event owner and notifying users that joined event

- Dropout from an activity the user has joined and notify the creator of Activity.





Fun-A-Holics

to agunda1 ▾

We want to let you know that user Tejaswi has de-registered from event Drawing competition.

Fig 11: Screenshots showing deregistering from event and sending notification to event owner

- Auto scaling based on CPU Utilization.

Two figures below show CPU utilization and number of instances for a particular period of time. We have implemented autoscaling based on CPU utilization. So whenever CPU utilization crosses 50% it's clear the number of instances has scaled up and when CPU utilization is less they have scaled down.

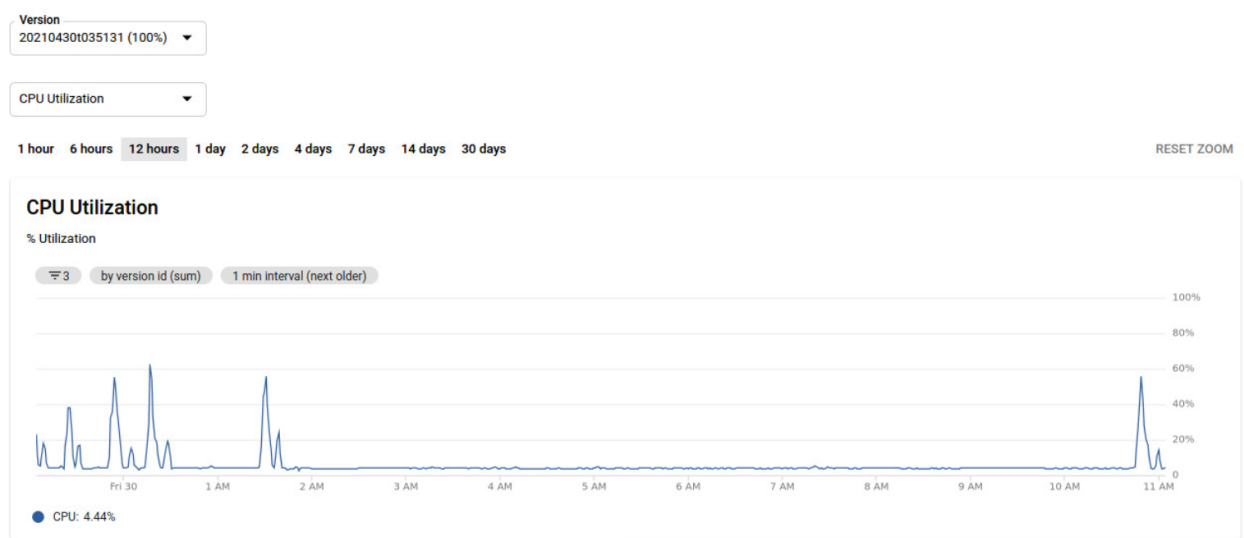


Fig 12: CPU utilization over a period of time

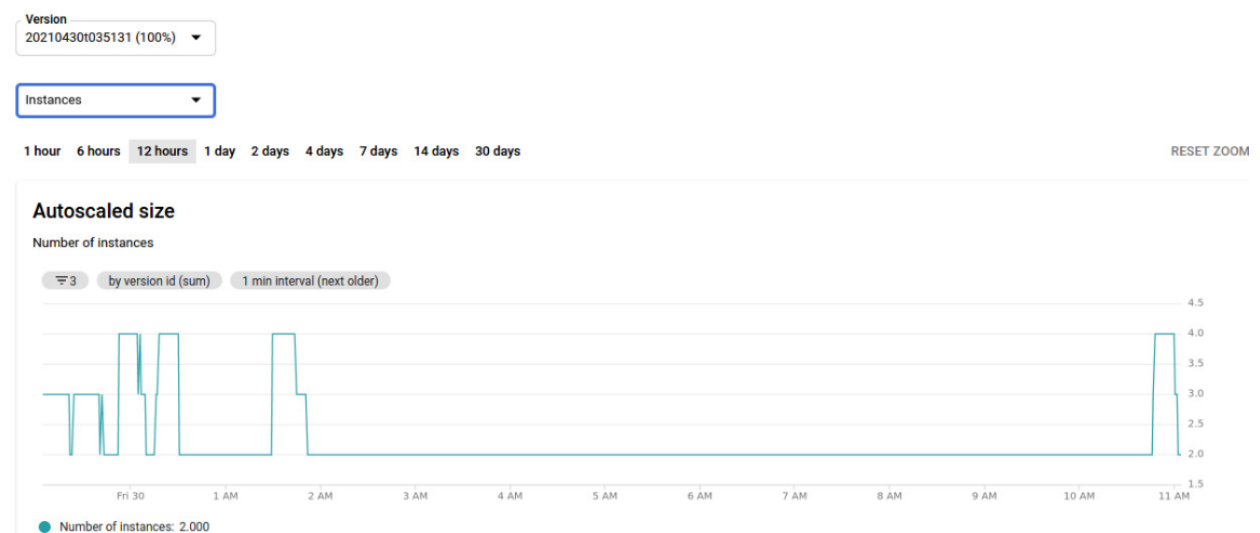


Fig 13: Number of instances scaled up and down over a period of time

- Additional Testing Screenshots:

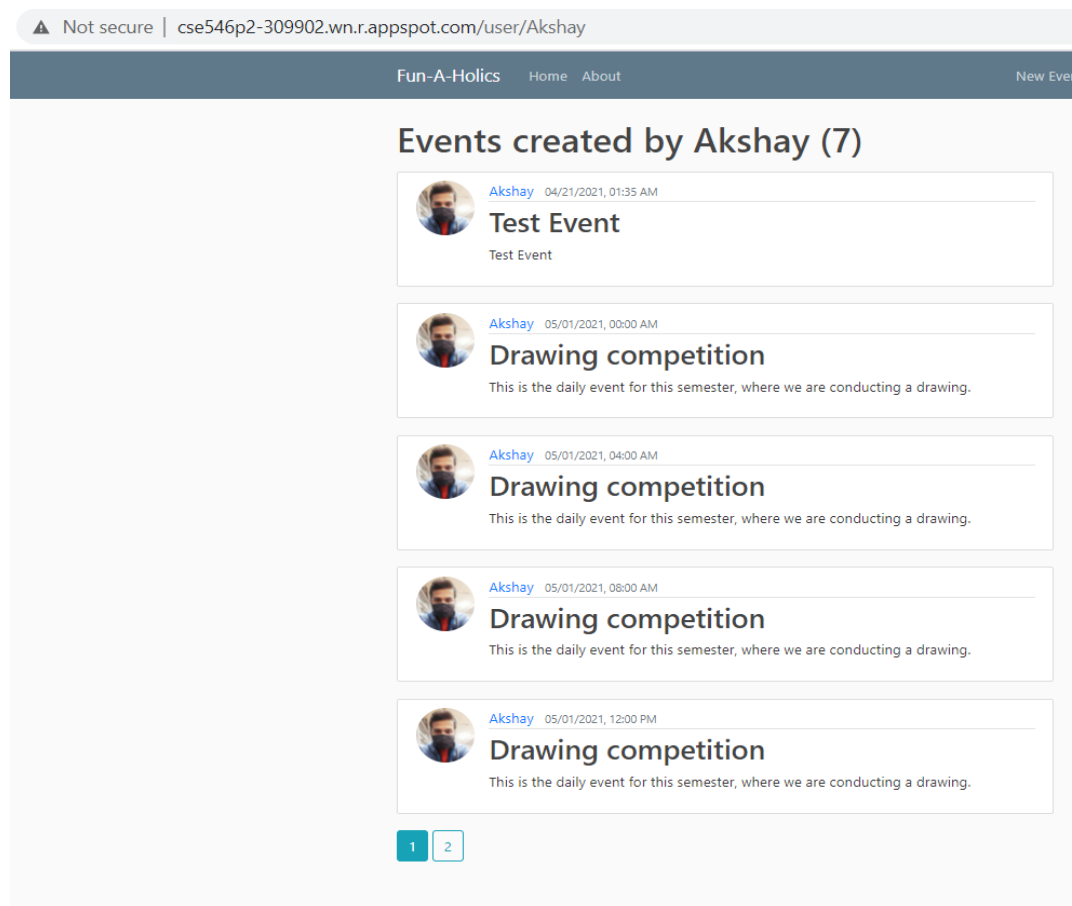


Fig 14: Screenshot showing the events created by particular user on clicking username in events home

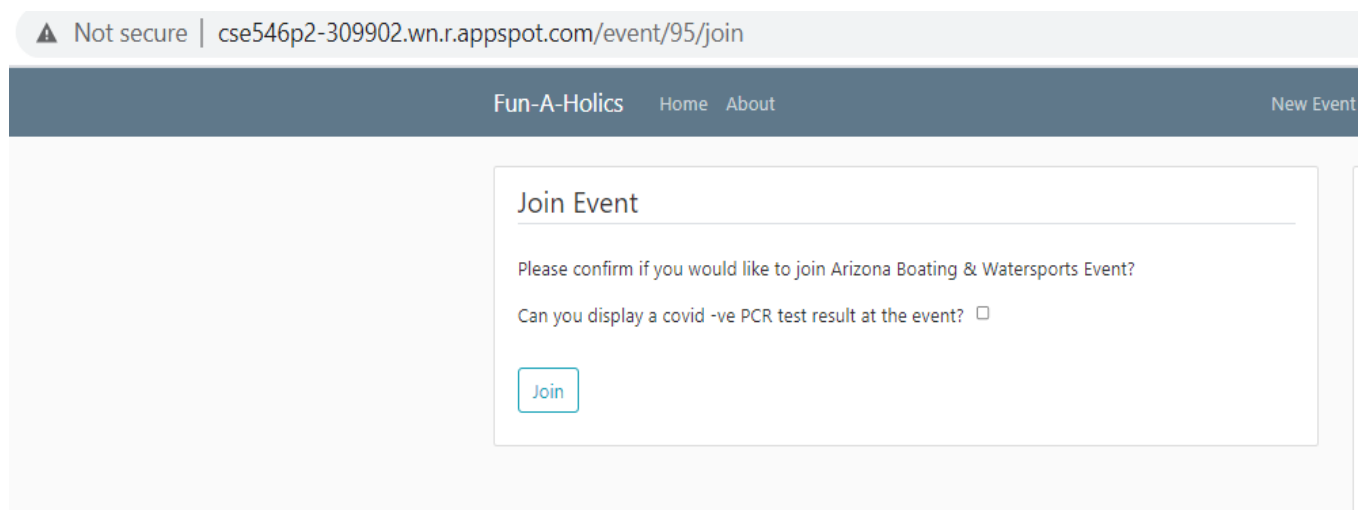


Fig 15: Screenshot showing confirmation of providing covid testing to join a outdoor event

Google Cloud Platform CSE546P2								
Cloud Scheduler Jobs REFRESH								
SCHEDULER JOBS APP ENGINE CRON JOBS								
Filter	Filter jobs							
App Engine URL ↑	App Engine Host	Description	Frequency	Last run	Result	Logs	Run	
/cronUpdateEventStatus	cse546p2-309902.wn.r.appspot.com	update event status	every 30 mins (UTC)	Apr 30, 2021, 4:32:00 PM	Success	View	RUN NOW	
/insertDailyJobs	cse546p2-309902.wn.r.appspot.com	create recurring events that happen daily	every 24 hours (UTC)	Apr 29, 2021, 9:03:19 PM	Success	View	RUN NOW	
/insertWeeklyJobs	cse546p2-309902.wn.r.appspot.com	create recurring events that happen weekly	every monday 09:00 (UTC)	Apr 29, 2021, 9:04:05 PM	Success	View	RUN NOW	
/jobsDeleteCancelled	cse546p2-309902.wn.r.appspot.com	delete cancelled events	every 24 hours (UTC)	Apr 29, 2021, 9:04:52 PM	Success	View	RUN NOW	

Fig 16: Screenshot showing the schedule job in Google Cloud Platform

5. Code

We have implemented the functionalities for this application using the Python flask framework [2]. The following are the main files:

- **models.py:**

The models file connects with the SQL database, which stores the information regarding the users, events and user_participation.

- **forms.py:**

The forms file consists of information regarding various forms that we have used to collect data. The Flask framework [2] will use this forms file to generate the front-end form pages.

- **routes.py:**

The routes file receives the API requests to the application, and processes the requests communicating with the database, then redirects them to the necessary front-end template which needs to be rendered.

- **functions.py:**

This functions file implements various database functionalities performing CRUD operations on the database, that are necessary as a part of project requirements.

- **load_generator.py:**

It contains the load generator code which will call a loadTesting API to simulate CPU utilization in instance to test autoscaling.

- **MailingService.py**

The code in this file is used to send e-mails to the users, notifying them about updates regarding the events they have joined and/or created.

- **cron.yaml**

The cron file schedules jobs such as inserting a few daily and weekly events, deleting completed/cancelled events and also updates the current status of the events that are implemented as a part of this project.

- **app.yaml**

It contains details related to autoscaling and the python version used for the application.

- **requirements.txt**

It contains all the dependencies that are required for the successful deployment and functioning of the application.

- **templates folder:**

The templates folder consists of the Front end template files which are rendered by the Flask framework by passing the required data from SQL database to this template.

- **static folder:**

The static folder consists of user profile pictures and other static data such as bootstrap and css files that are used in the application.

Google App Engine Deployment:

Deploy the code in google app engine instance and execute `gcloud app deploy`. It will install all the required dependencies that are mentioned in `requirements.txt` and on successful deployment generates a url which can be used to access application. This is done following the reference from [3]

After that, execute `gcloud app deploy cron.yaml` to create scheduled jobs.

6. Conclusions

The project addresses the problem of zoom fatigue faced by many remote workers. The brainstorming sessions on choosing our project promoted critical thinking, team discussions, work planning and allocation etc.

Further, the project is a great learning experience through hands-on implementation to develop a cloud native application on google cloud, leveraging google cloud services. Learned google cloud technologies through hands on implementation. The RESTful web application leverages the HTTP protocol. Also, we implemented the auto scaling feature to scale our solution. We consider the scalability of our product from the initial design phase and a fully cloud native application is easy to scale through autoscaling. The User Interface was designed to enhance the user experience and provide easy navigation to access our backend services.

To improve our solution, we can implement containerization of the services and use kubernetes to address the autoscaling and high availability concerns of the popular services. Once a user is registered for an event, a calendar invite should get added to the email sent to the user. One more improvement is that the active events can be suggested based on the location of the user using GPS or location api's. Also, new events can be recommended to the users based on their previous events registered or interested.

7. References

[1] *Autoscaling GCP application instances*

<https://cloud.google.com/appengine/docs/standard/python/how-instances-are-managed>

[2] *Flask: Python web development framework* <https://flask.palletsprojects.com/>

[3] *Google App Engine Deployment for python web app* <https://www.youtube.com/watch?v=qeSpDwA2qcU>