VIT-AP UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE AND ENGG.

CLOUD COMPUTING

CSE4001

PROJECT REVIEW 2

Webapp Hosting

Academic Year: 2020-2021

Semester: Winter

Faculty Name: Dr. S.Karthikeyan

Student name: Kathal Aditya Rajendra

Branch/ Class: B.Tech

School: SCOPE

Reg. No.: 18BCD7008

Outline

- Abstract
- Introduction
- Project Description
- Advantages
- Methodology
- Project Flowchart
- Project Timeline
- Project Demo
- References

Abstract

The aim of this platform is to build a web application that provides an interactive environment to the various clubs and students by providing updates regarding different events and activities.

It creates an open forum which brings together all the students of similar interests by providing the clubs and student a single point of contact and making it easier for them to apply for and manage events.

The cloud service providers include Google Cloud , MongoDB , Cron Job , Heroku and AWS will be used for this project.

Introduction

VIT-AP has a portal for everything. We have portal for payments , student data management , placements, etc. but not one for clubs.

Students receive many emails from clubs and the college regarding different events which are to occur. Clubs also have to manage a lot of data to properly interact with students and maintain a healthy participation in their events.

To make this process easier we aim to provide an interface through which clubs can effectively communicate with interested students and additionally attract more students.

Project Description

This is a SaaS(Software as a Service) platform that is build upon a PaaS(Platform as a service) and uses other services such as Storage as a Service , Software as a Service. It follows the MVT architecture. Python is the primary coding language and HTML , CSS , JS for UI creation.

The project has the following objectives:

- To build a highly portable application
- To reduce the workload of clubs and college administrators
- To follow a subscription based model to drive event registration

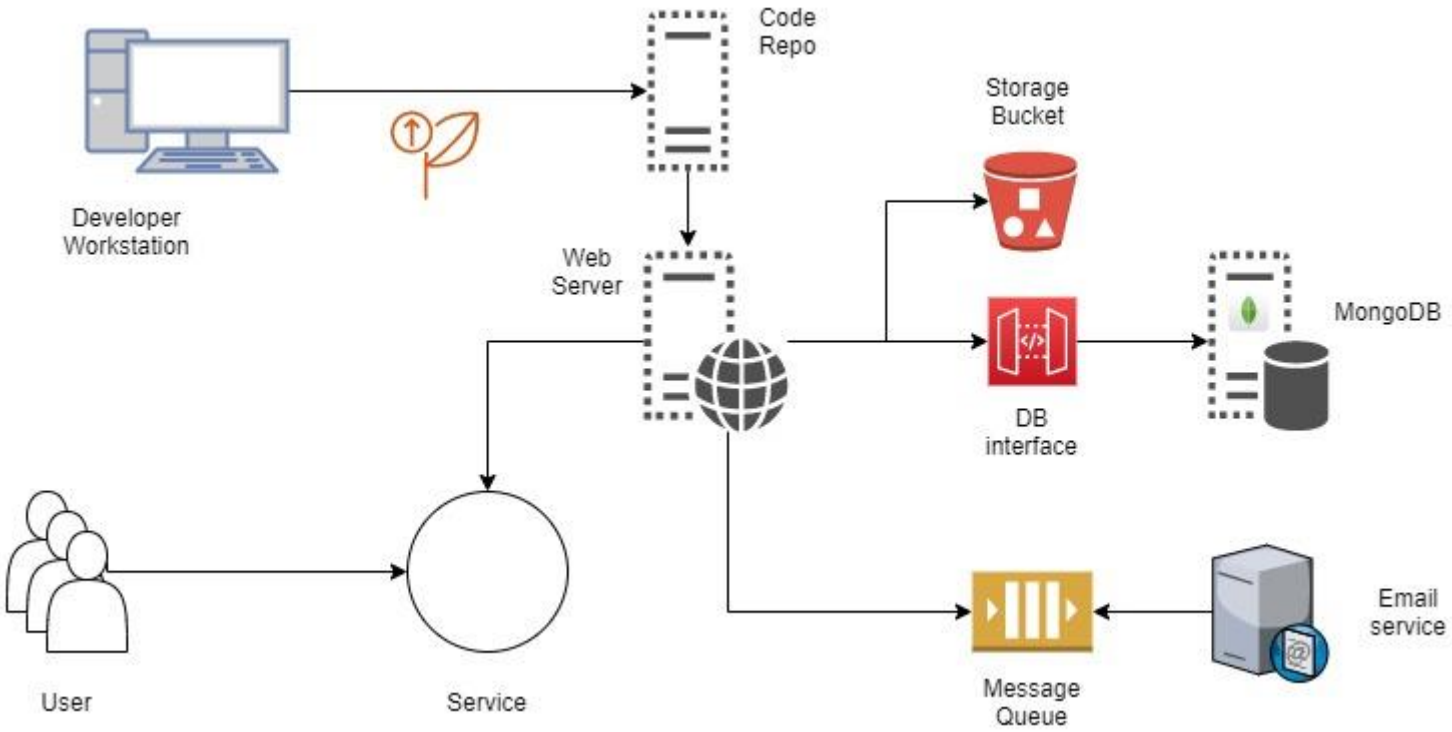
Advantages

The main advantage of this web app is that it increases the event participation in club events by following a subscription model which provides better user interaction while keeping the mails to the minimum.

Being cloud based the webapp has high portability and an mobile app can easily be made by using the same data model reducing the development time.

This also decreases the workload on the college administration and make club handling easier by abstracting the process for them.

Methodology



Modules

Database Setup - 100%

User Authentication and Management Module - 100%

Admin Module - 100%

Event Module - 100%

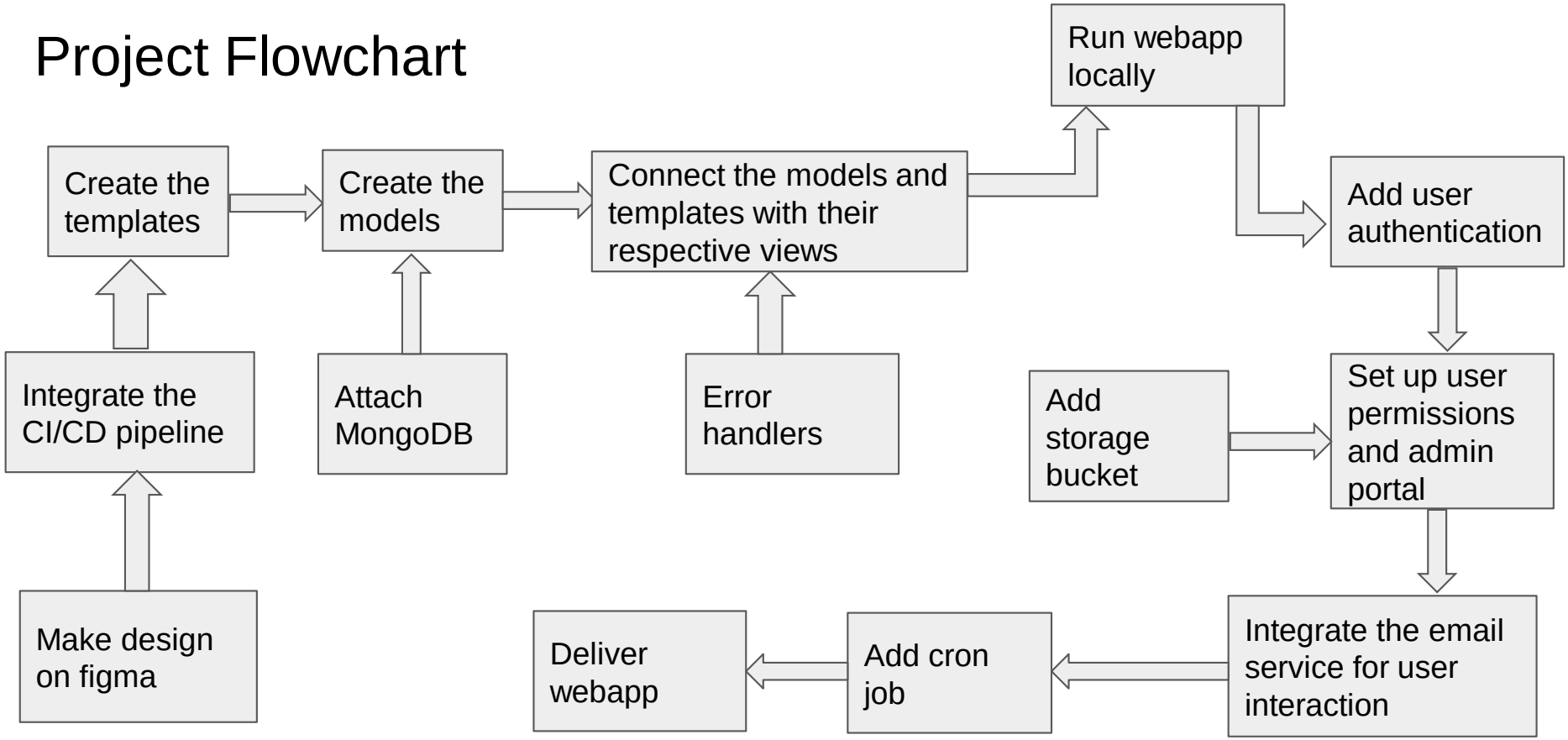
Club Module - 100%

Database Interface - 100%

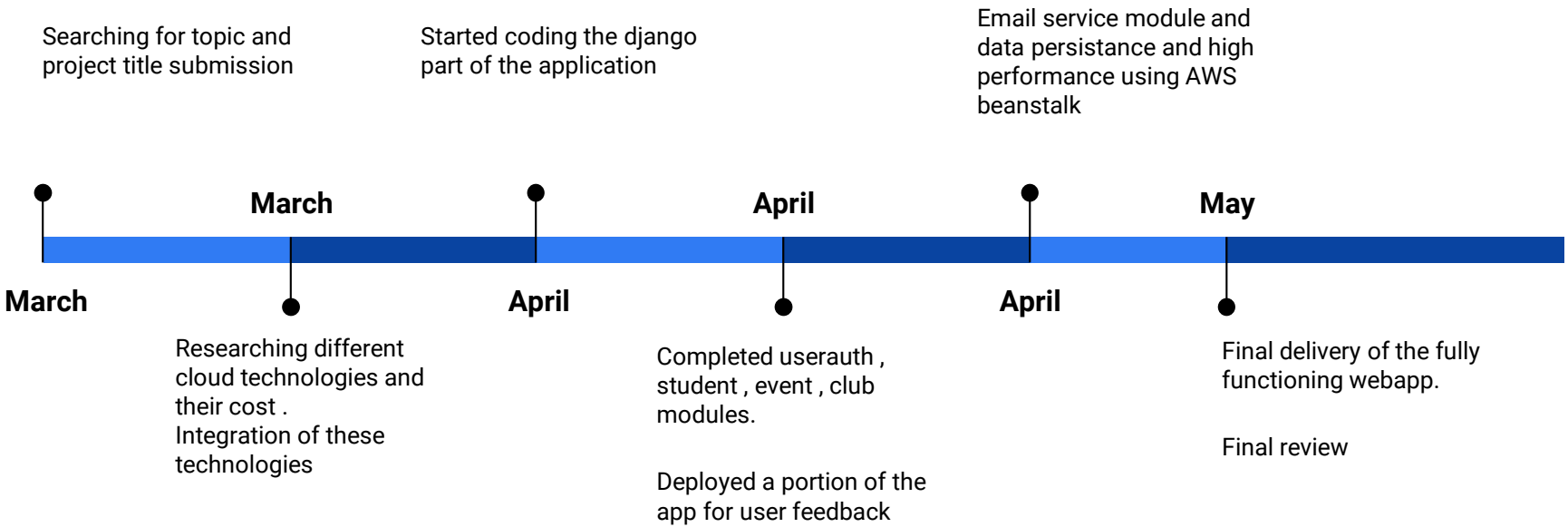
Email Service - 100%

Cron job - 100%

Project Flowchart



Project Timeline



Project Demo

The main aim of this project is to create a platform that will reduce the time club spend on sending email and other event related tasks. It will also provide college authorities a single point of contact for all clubs making the process easier for both parties.

So for this project demo first we will look at the final platform and understand all the features provided to various users. Then we will look at the final architecture implemented in this project and deep dive into each part of that architecture.

Application Demo

Landing Page

- This page is accessible to all users. When the user is login in to application the sing in button turn red to sign out. For the user to sign out
- Register with us leads to sign up page
- Events and Clubs redirect to their respective event and club pages
- Clicking the name in the navbar leads back to the home page. This is true for all the pages in the project.
- When the user is not login into the application the sign in button is shown and upon clicking that button it redirects to sign in page.
- Home button is also shown when the user is login into the application , which redirects to the custom home page for that user.


Start something that matters
Stop wasting valuable time in event registration
Register With Us


Start something that matters
Stop wasting valuable time in event registration
Register With Us

Sign Up Page

- This page is used for registration of new users onto the platform.

- As this application is VIT-AP focused , the user needs to provide the following information to create an account
  - Roll No
  - Name
  - Email I
  - Password for future login
- If a user is already a member then they can click the link provided below the image. They will be redirected to login page.
- If same email id is used twice the platform doesn’t allow one to register.
- Also a email verification mail is send to all the new users and they are not allowed to login until they verify their email id’s.

Sign up

Your Roll No.

Your Name

Your Email

Password

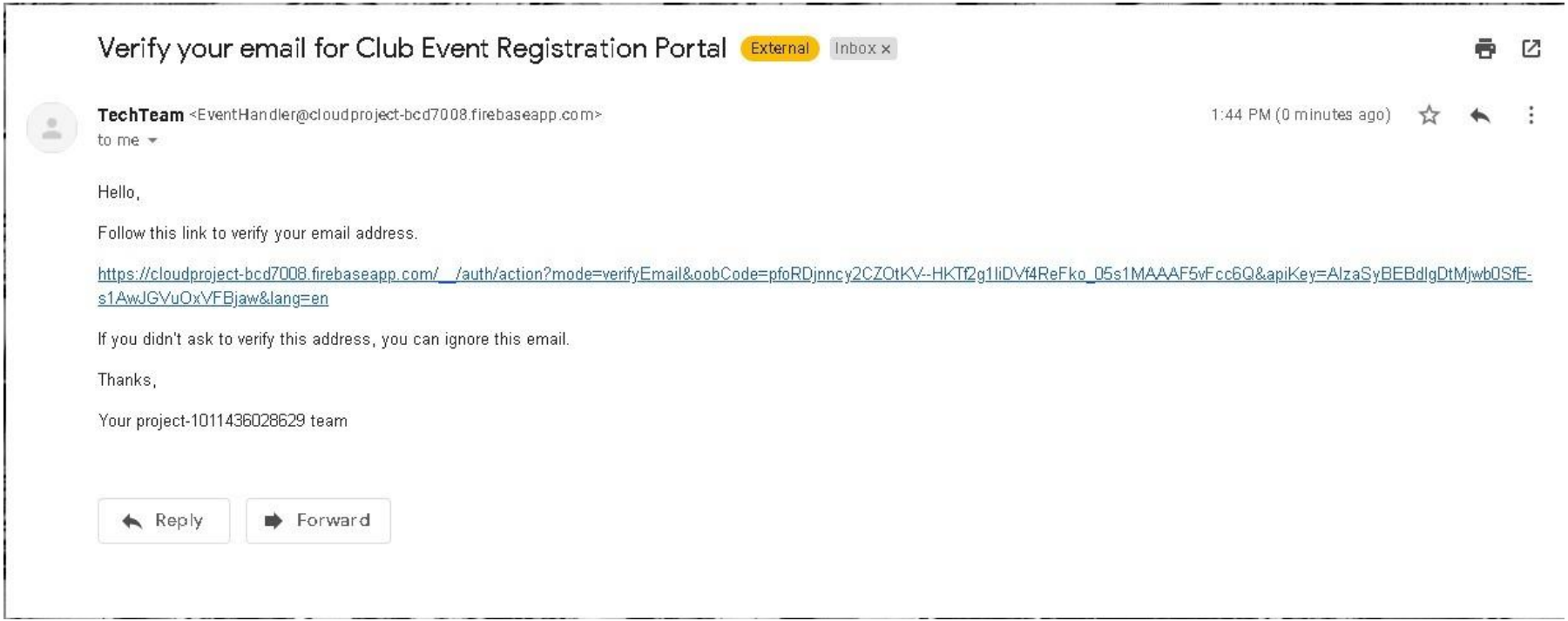
Register

I am already member

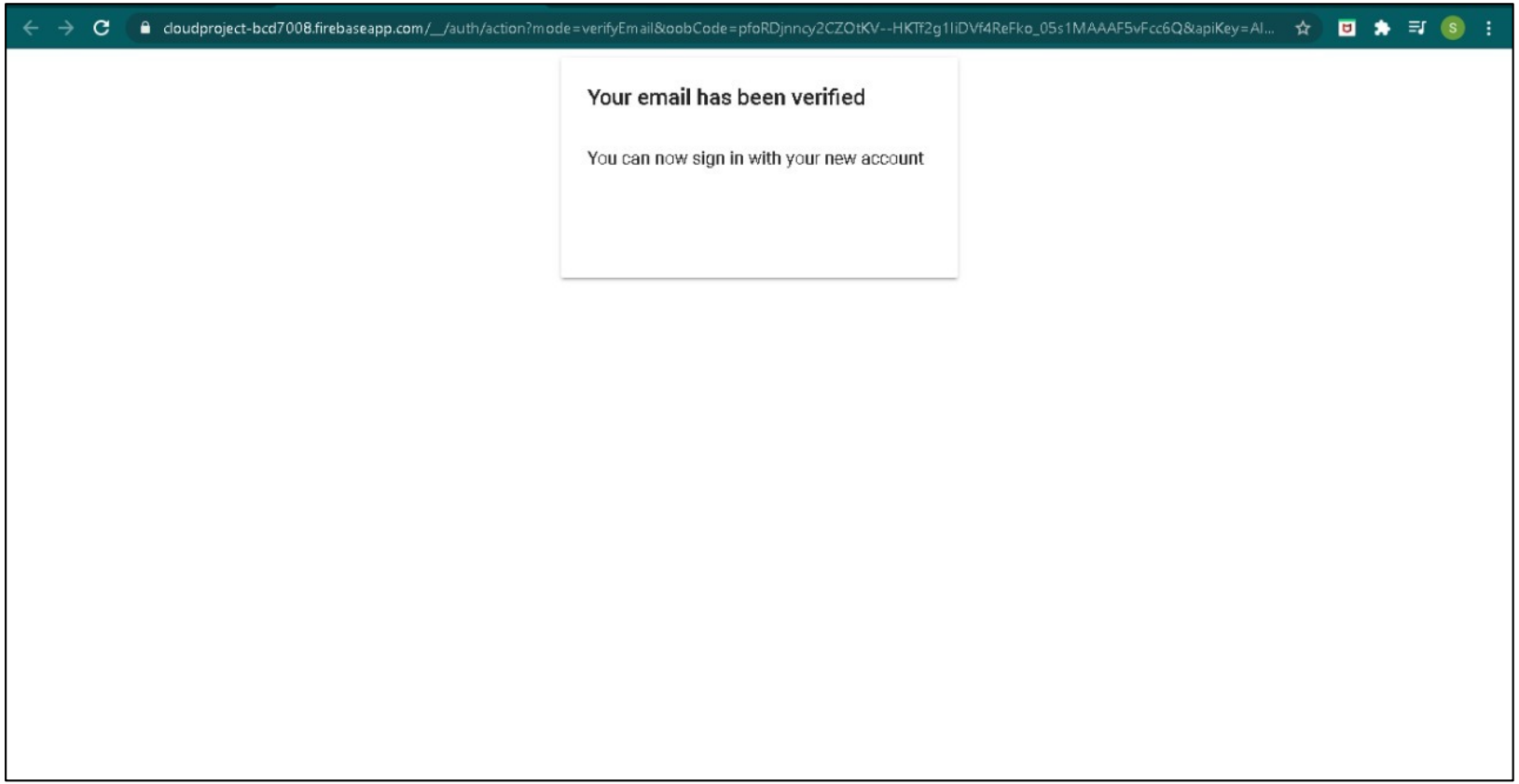
Thank You!

A confirmation email has been send to your given email id. Verify your email id to use the services of this portal

Home Page



I have used my friends email address to perform few tasks. I have done so because my email address is already registered as an admin user and one email can be linked with only one type of user access. So I have used another persons account to display student user tasks.


cloudproject-bcd7008.firebaseapp.com/__/auth/action?mode=verifyEmail&oobCode=pfoRDjnncy2CZOtKV--HKTf2g1IiDVf4ReFko_05s1MAAAF5vFcc6Q&apiKey=AI...
Your email has been verified
You can now sign in with your new account

Sign In Page

- The user need to give its registered and verified email Id and password to login into the system.
- If the email id is not authenticated by clicking on the link website will not allow the person to authenticate himself and throw an error.
- All users that includes Students, Clubs, College Administration will use this single sign on page.

Sign In

Your Email

Password

login

I am not a member

Invalid Username or Password

Sign In

Your Email

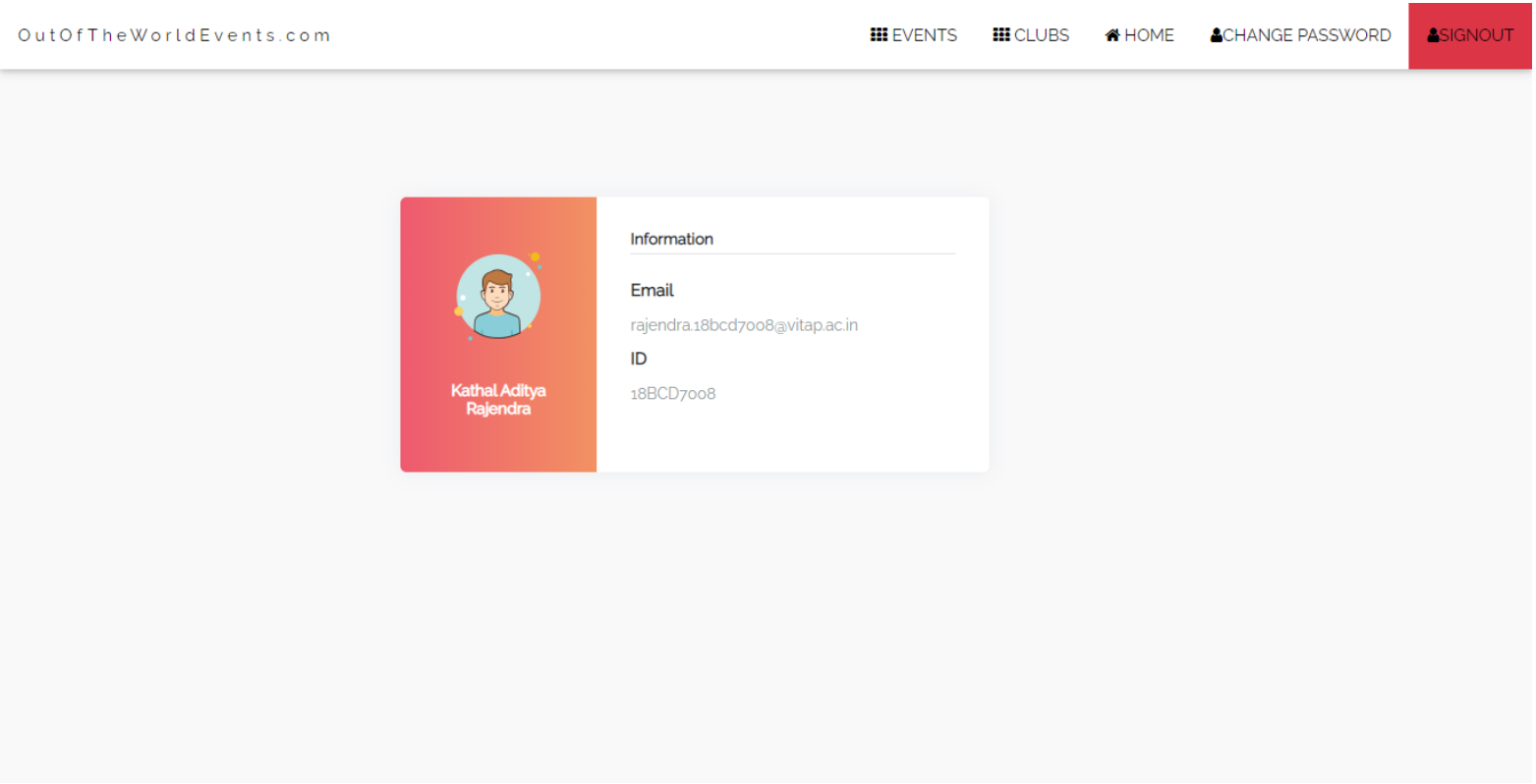
Password

login

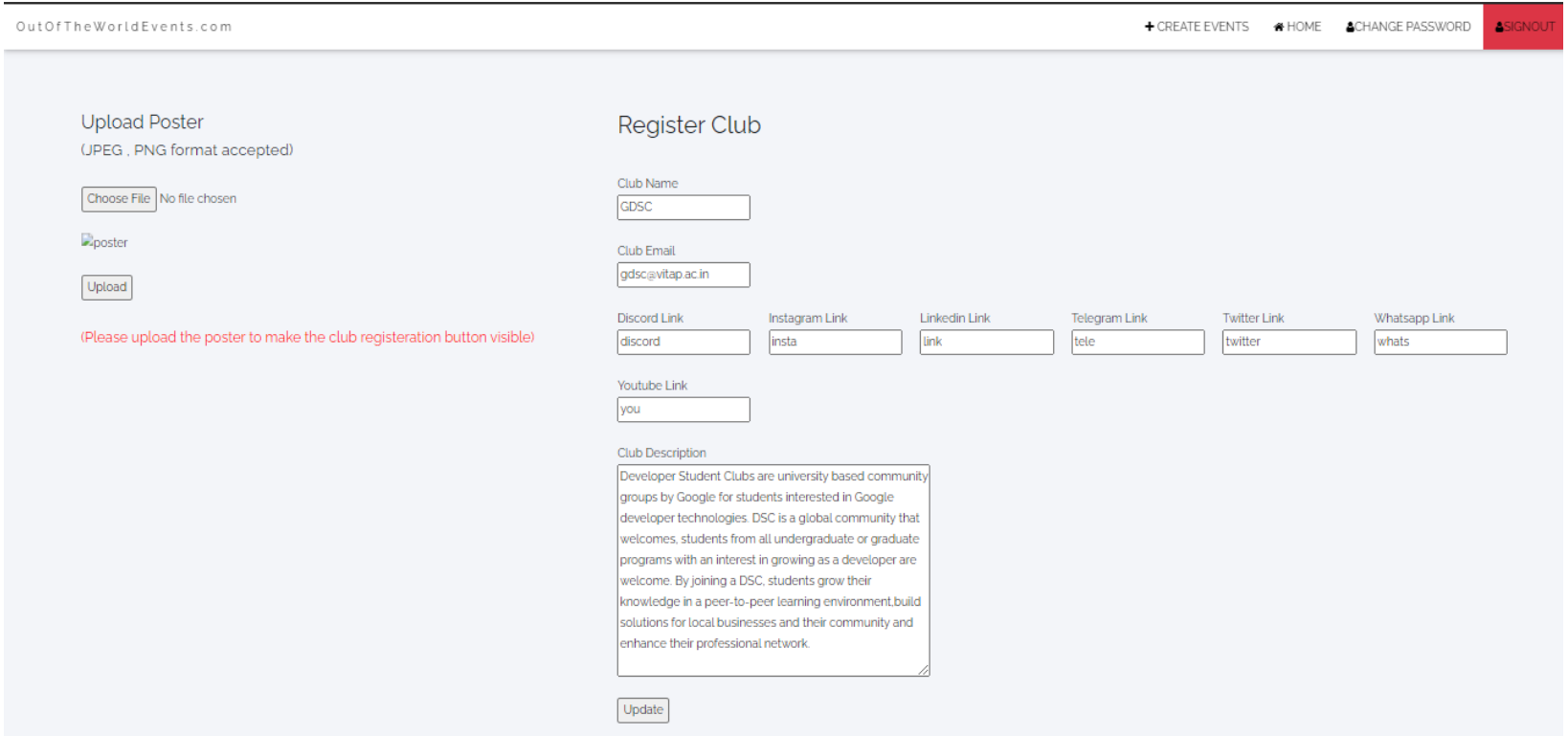
I am not a member

Profile Page

- Profile Pages are loaded as per the users except admin that login into the system.
- All users are provided an option to change password on their respective profile page.
- For students they provide general information about their account and the data that is associated with their account.
- For clubs they are able to change the information that will be shown on their page to other users and also upload new club posters.
- Admins are provided the change password option on their home page itself as there is no separate profile page for them.



Student Profile Page



Club Profile Page

Events Page

- This page shows all the events that are live on the platform.
- For users that authenticate themselves with the platform and have subscribed to any club , the events of these clubs are shown in preference to the others.
- For un authenticated users register button click will redirect them to the login page.
- View more for both the users will open a detailed view of the event. This link can also be shared by clubs for quick reference to their event.
- For events in which register button is not shown that indicates the user has already registered for that event or registration for that event are closed.

Subscribed Club Events

Login to view subscribed clubs events

All Events

Android Study Jam

Expert Panel

BITBYBIT

CN

For un authenticated user

Subscribed Club Events

Android Study Jam

GDSC

View More

Expert Panel

GDSC

View More Register

BITBYBIT

GDSC

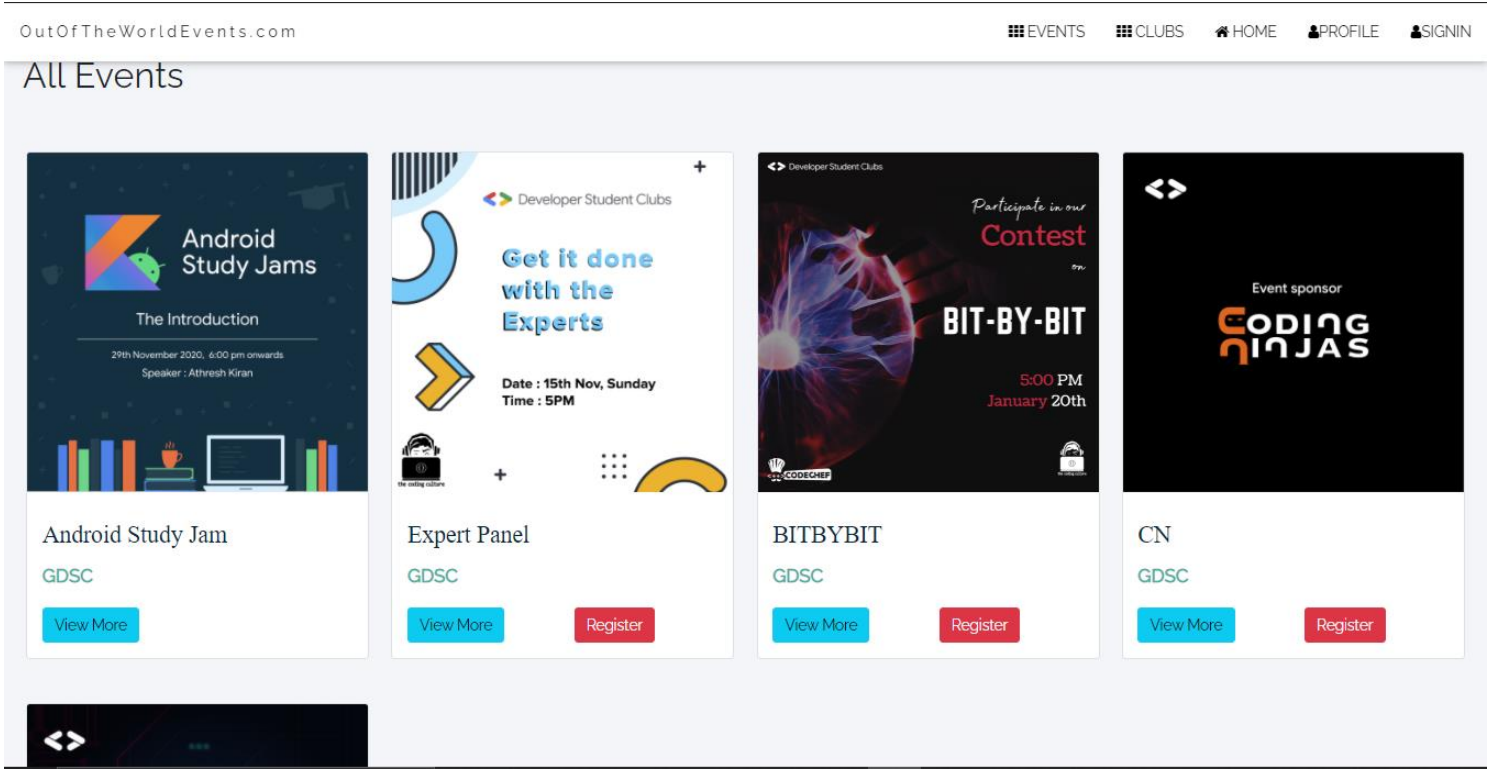
View More Register

CN

GDSC

View More Register

For authenticated users


OutOfTheWorldEvents.com
EVENTS
CLUBS
HOME
PROFILE
SIGNIN
All Events
Android Study Jams
The Introduction
29th November 2020, 6:00 pm onwards
Speaker : Athresh Kiran
Developer Student Clubs
Get it done with the Experts
Date : 15th Nov, Sunday
Time : 5PM
Developer Student Clubs
Participate in our
Contest
on
BIT-BY-BIT
5:00 PM
January 20th
CODECHEF
Event sponsor
CODING NINJAS
Android Study Jam
GDSC
View More
Expert Panel
GDSC
View More
Register
BITBYBIT
GDSC
View More
Register
CN
GDSC
View More
Register

All Events

Event Page

- Event pages shows every detail about the event that is needed to be known to the user and provided by the club.
- Clubs can also close the event even before its last date.
- User are required to login to register for an event.
- If a user is already registered then un register option is shown.
- Email is send upon successful event registration.
- Email is send using a dummy account setup for time being.

BITBYBIT

ClubName:	GDSC
EventLocation:	VIT-AP
EventStartTime:	June 30, 2021, 8:34 p.m.
EventEndTime:	June 17, 2021, 8:34 p.m.

Login to register for the event

Event Details

BITBYBIT

ClubName:	GDSC
EventLocation:	VIT-AP
EventStartTime:	June 30, 2021, 8:34 p.m.
EventEndTime:	June 17, 2021, 8:34 p.m.

Login to register for the event

About the Event

Learn the basics of competitive coding

After clicking the down arrow

BITBYBIT

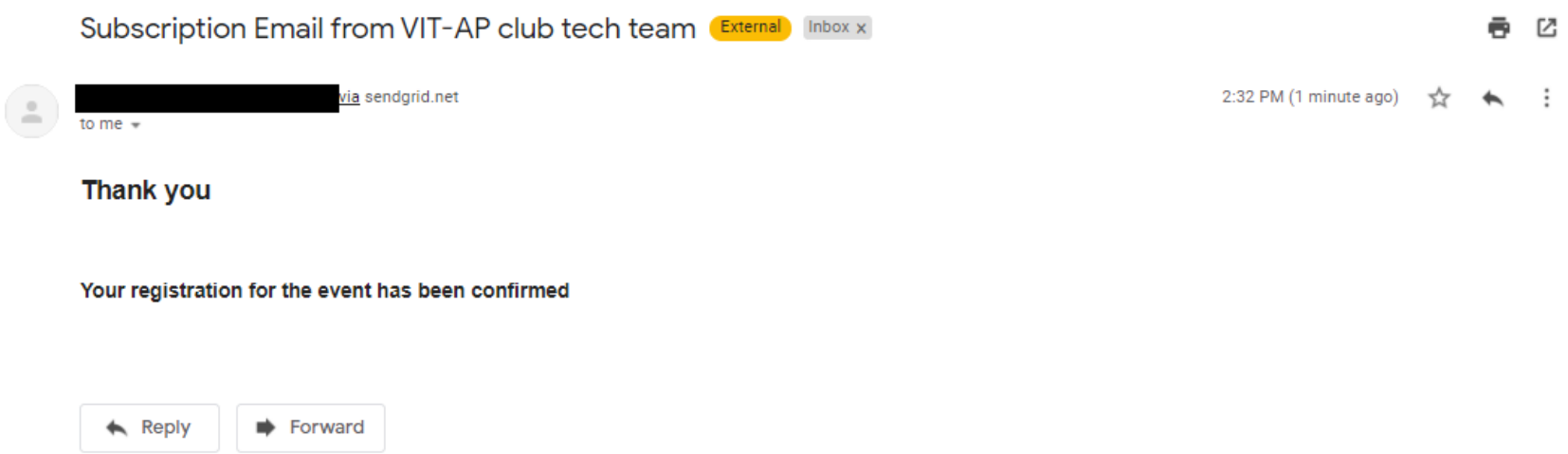
ClubName:	GDSC
EventLocation:	VIT-AP
EventStartTime:	June 30, 2021, 8:34 p.m.
EventEndTime:	June 17, 2021, 8:34 p.m.

Register

For authenticated user

```
Microsoft Windows [Version 10.0.18363.1440]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\Ak\Desktop\django_project\cloudheroku>c:/Users/Ak/Desktop/django_pr
oject/CloudProject/sandbox/Scripts/activate.bat

(sandbox) C:\Users\Ak\Desktop\django_project\cloudheroku>py eventreceiver.py

 [*] Waiting for messages:
 [x] Received b'rajendra.18bcd7008@vitap.ac.in'
```

Queue Update upon clicking register and Email



BITBYBIT

ClubName:	GDSC
EventLocation:	VIT-AP
EventStartTime:	June 30, 2021, 8:34 p.m.
EventEndTime:	June 17, 2021, 8:34 p.m.

Unregister

When already registered

Android Study Jam

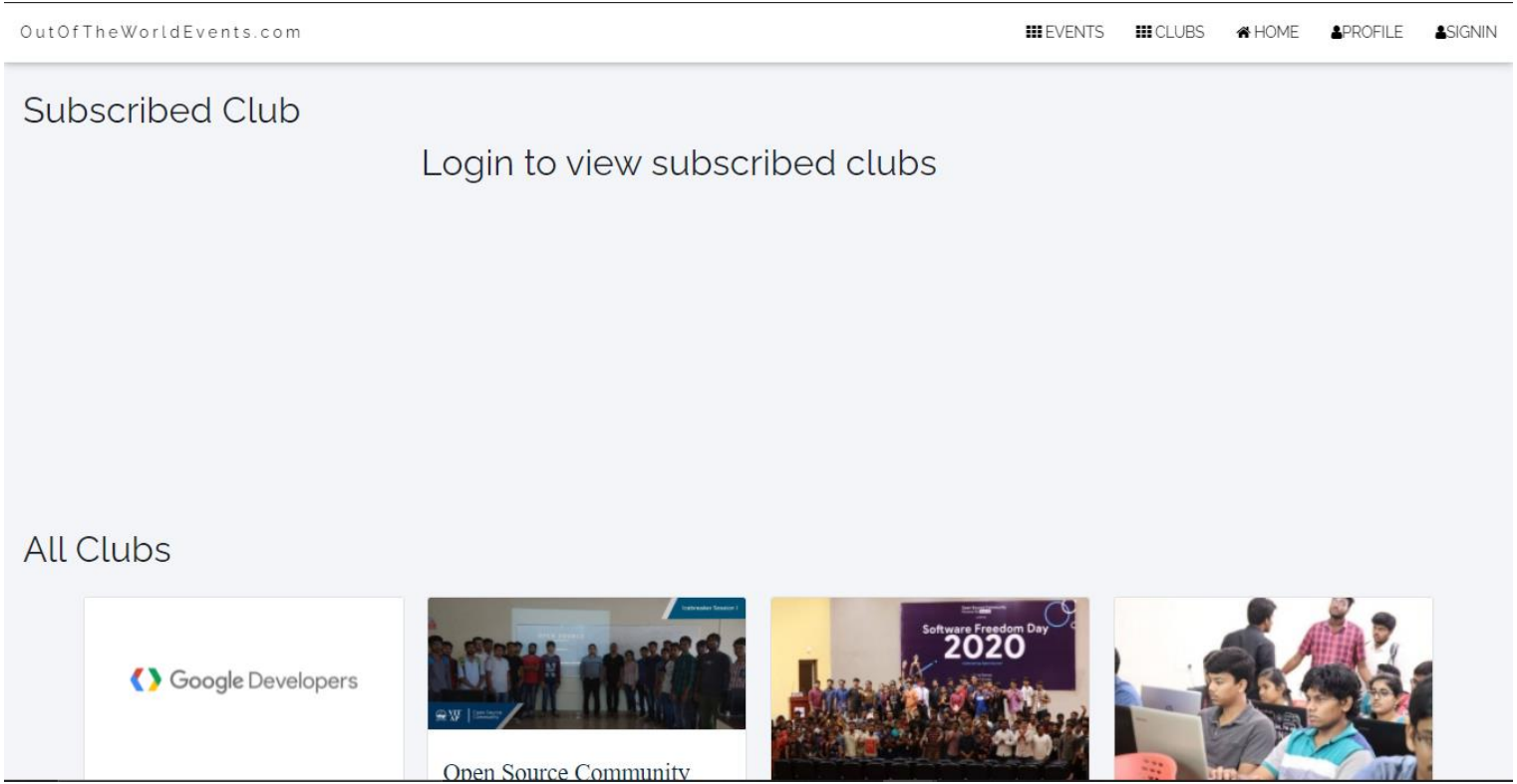
ClubName:	GDSC
EventLocation:	Pune
EventStartTime:	June 15, 2021, 1:41 p.m.
EventEndTime:	June 16, 2021, 1:41 p.m.

Registrations for this event are closed

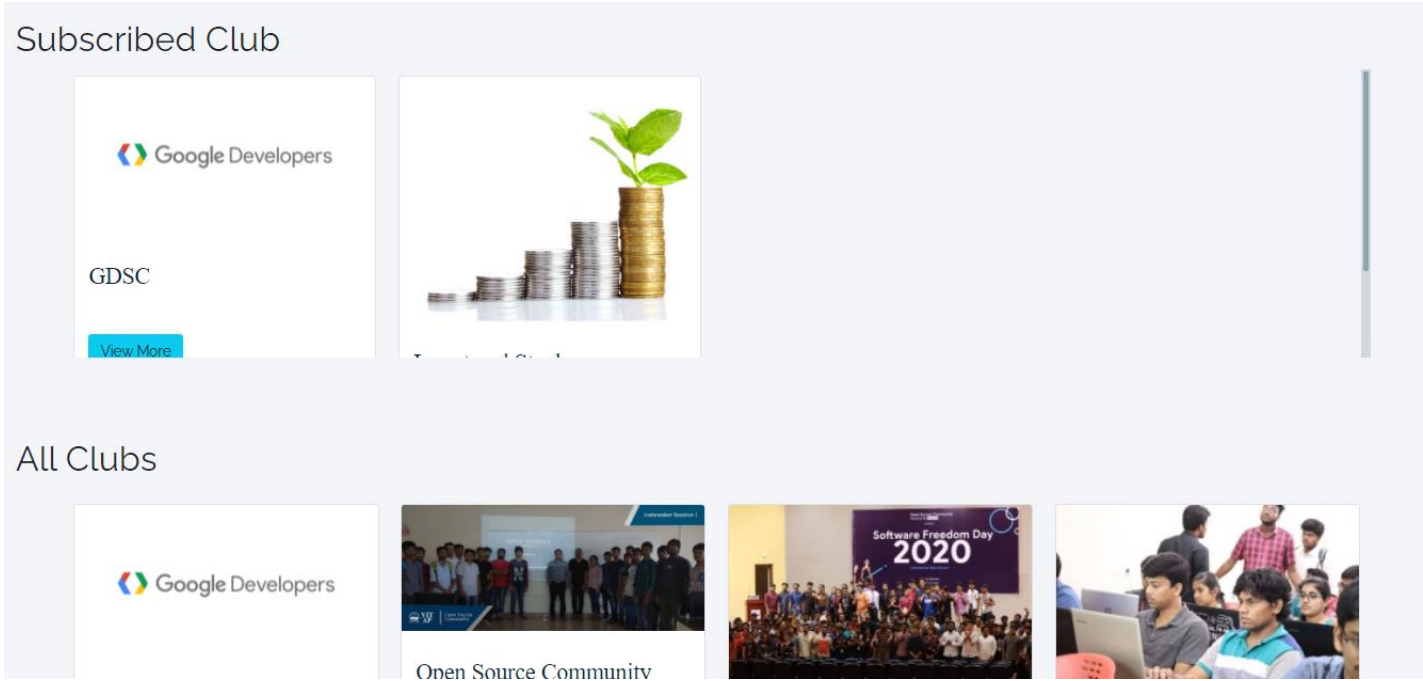
Event when registration is closed

Clubs Page

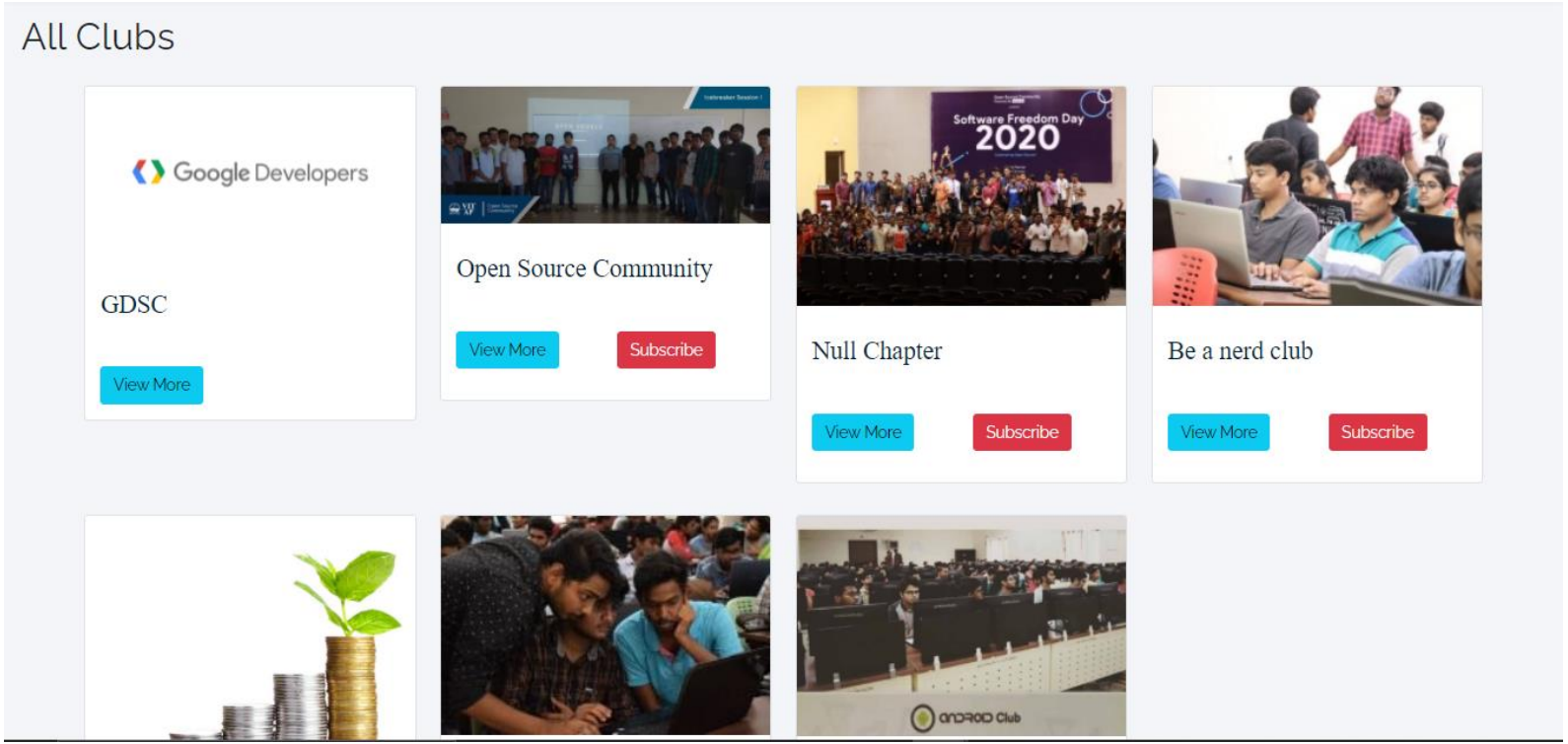
- The clubs age is only accessible to students and not club and admin users.
- The page first lists all the subscribed club and then all clubs.
- If subscribe button is not shown indicates that the user has already subscribed to the club



For un authenticated user



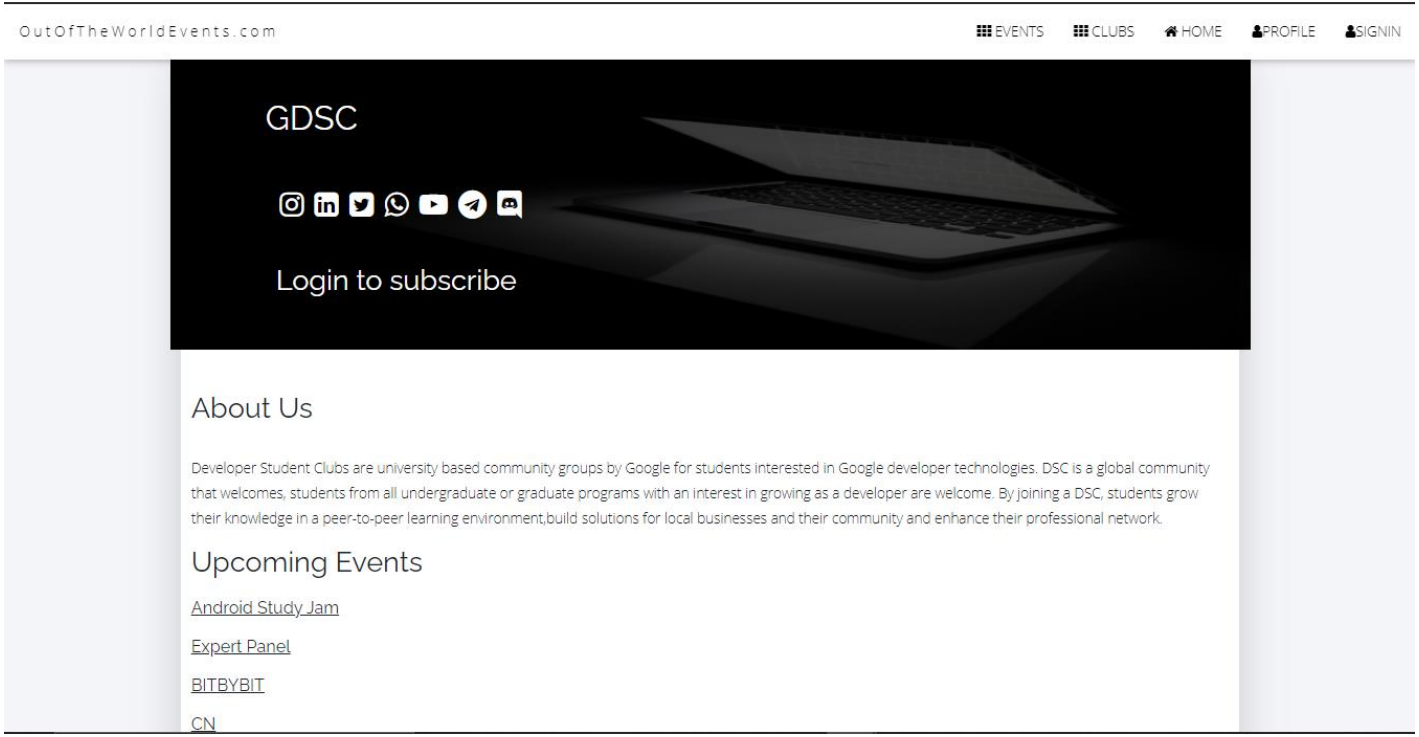
For authenticated user


All Clubs
Google Developers
GDSC
View More
Open Source Community
View More
Subscribe
Software Freedom Day
2020
Null Chapter
View More
Subscribe
Be a nerd club
View More
Subscribe
Club

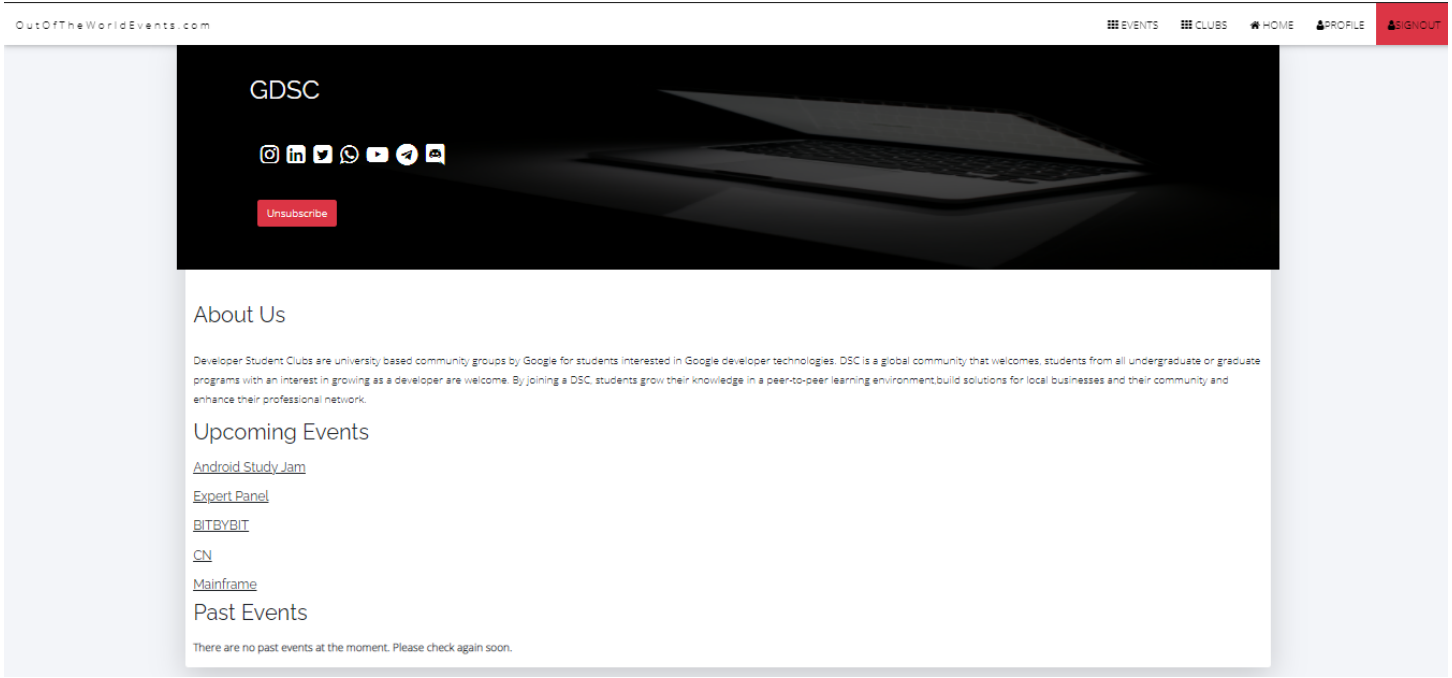
All Clubs

Club Page

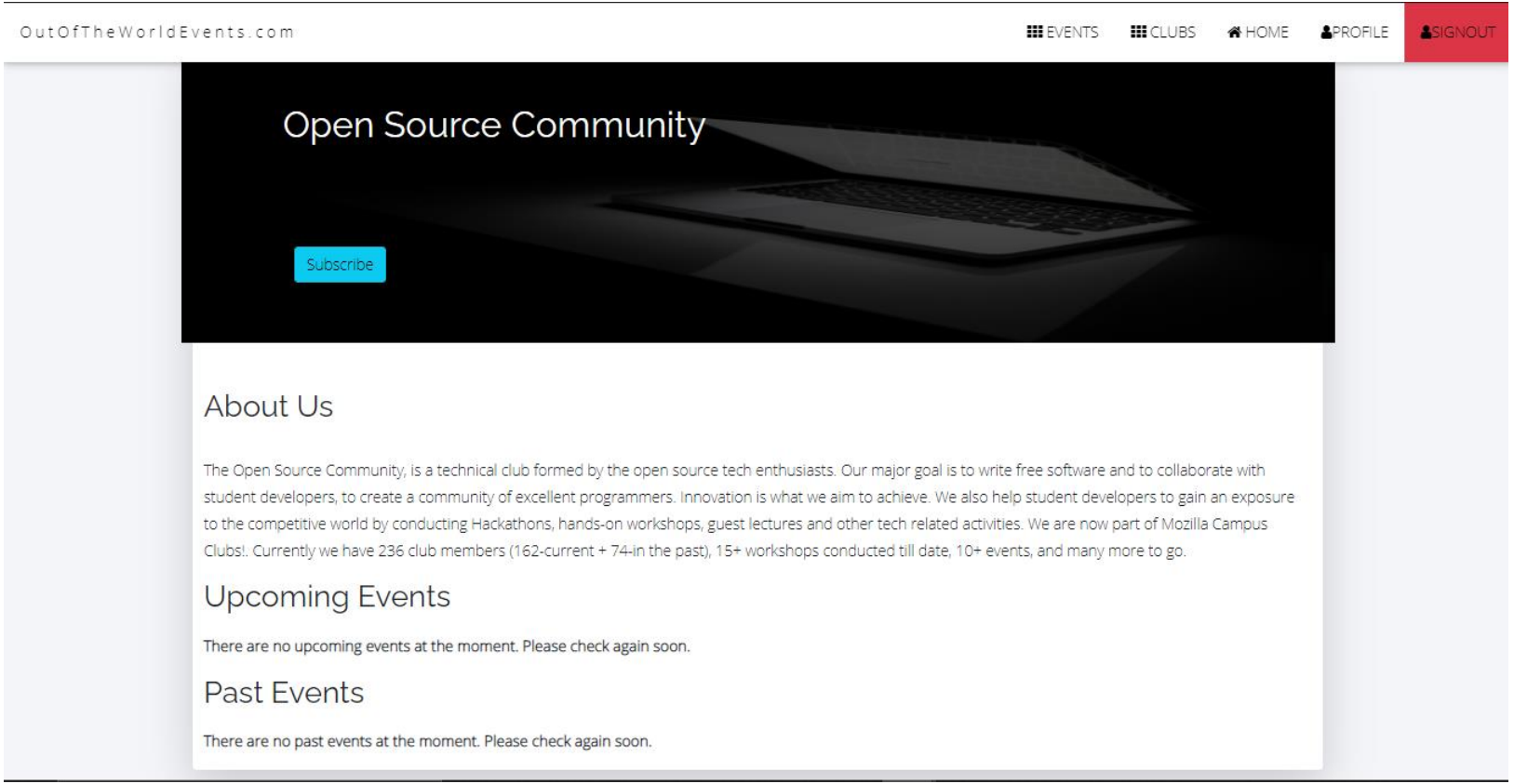
- This page provided detail information about a club.
- The content of this page can be changed by respective clubs from their profile pages.
- This page is accessible for only student users.
- For authenticated users it gives and option of subscribe and un subscribe.
- For un authenticated users it provides and option to login upon which the aforementioned options will be provided.
- Email is send whenever a user subscribes to a club on behalf of that club.
- The email template is HTML based. So clubs can design a html view of their email and make it more interactive.
- A list of all active and past events is also provided in the club summary. This list is system generated and clubs do not need to handle this part.



Club Page for Unauthenticated users



Authenticated user || Already subscribed



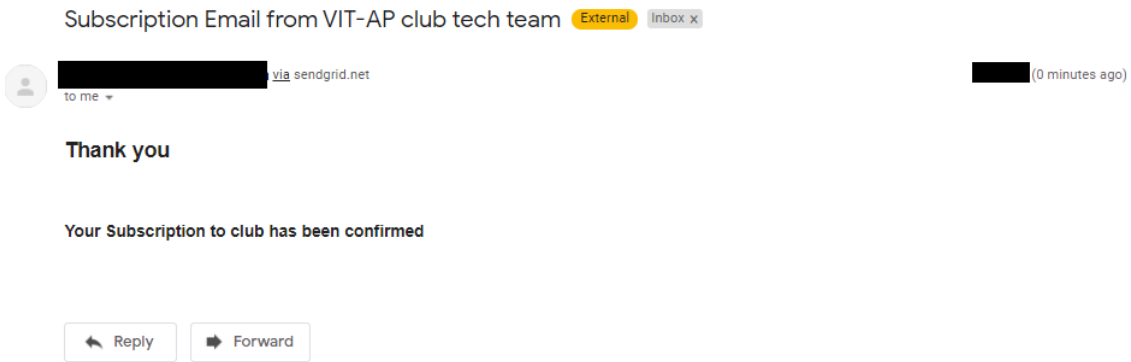
Authenticated user || Non subscribed club

```
(sandbox) C:\Users\Ak\Desktop\django_project\cloudheroku>py clubreceiver.py
 [*] Waiting for messages:
```

Queue is Empty

```
(sandbox) C:\Users\Ak\Desktop\django_project\cloudheroku>py clubreceiver.py
 [*] Waiting for messages:
 [x] Received b'rajendra.18bcd7008@vitap.ac.in'
```

After clicking subscribe



Email Received

Create Club Page

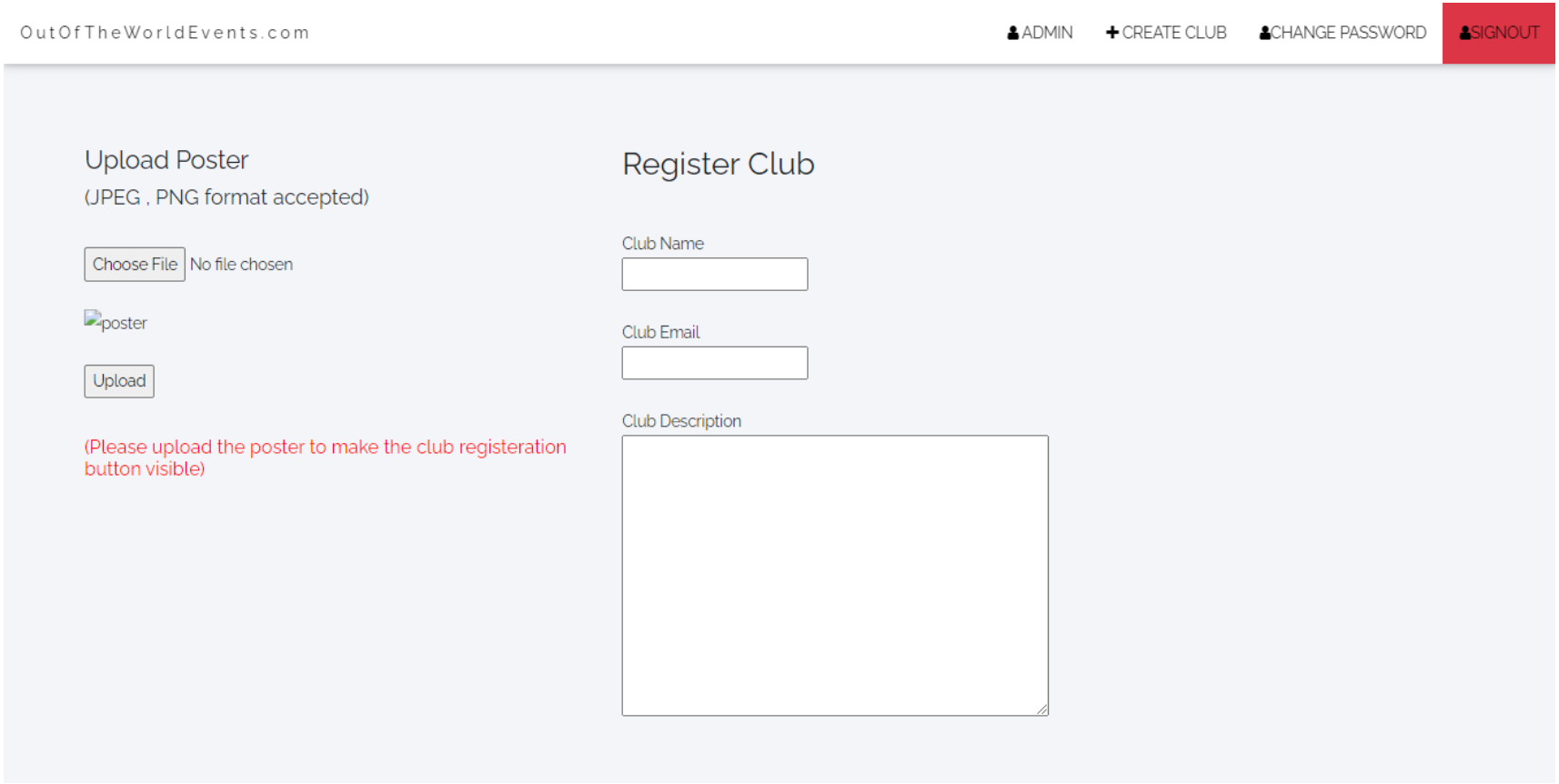
- For club creation the administration needs to provide following information :

- Club Name
- Club Email
- Club Description
- Poster Image

- A one time password is generated for the clubs to login into their newly created account and change password to their password of choice.

- This password is generated using UUID4.

- If the one time password is lost the whole needs to start all over again.



Club Registration Form

OutOfTheWorldEvents.com

cloudproject-bcd.herokuapp.com says

Image Upload Successful

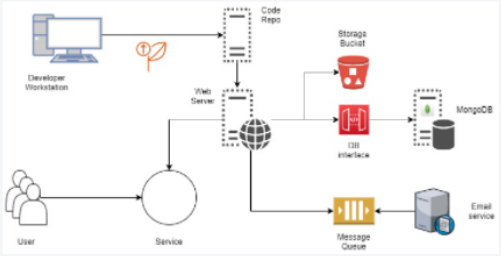
OK

MIN + CREATE CLUB CHANGE PASSWORD SIGNOUT

Upload Poster

(JPEG , PNG format accepted)

Choose File cloud.jpg



Upload

(Please upload the poster to make the club registeration button visible)

Register Club

Club Name

Dummy Club

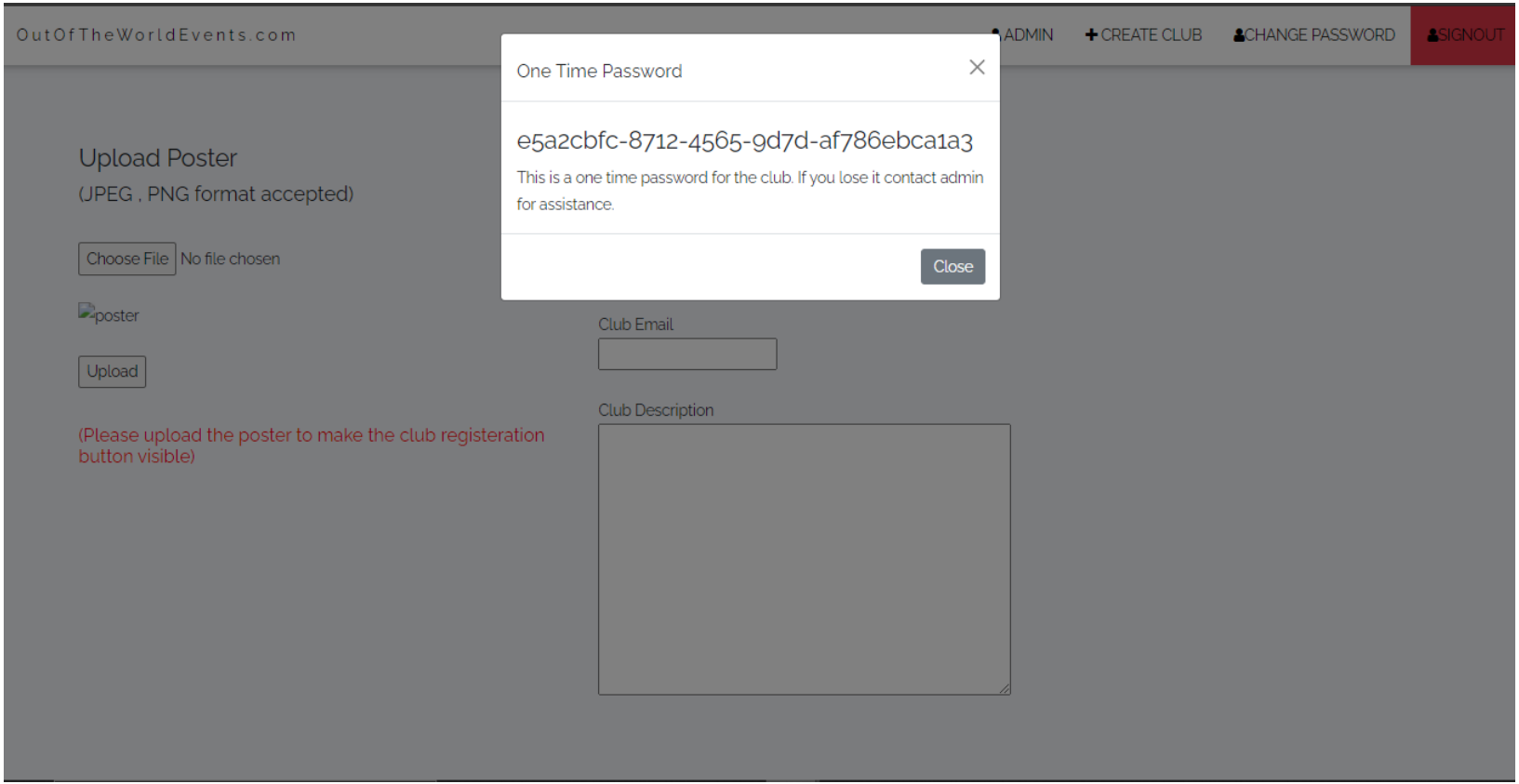
Club Email

dummy@gmail.com

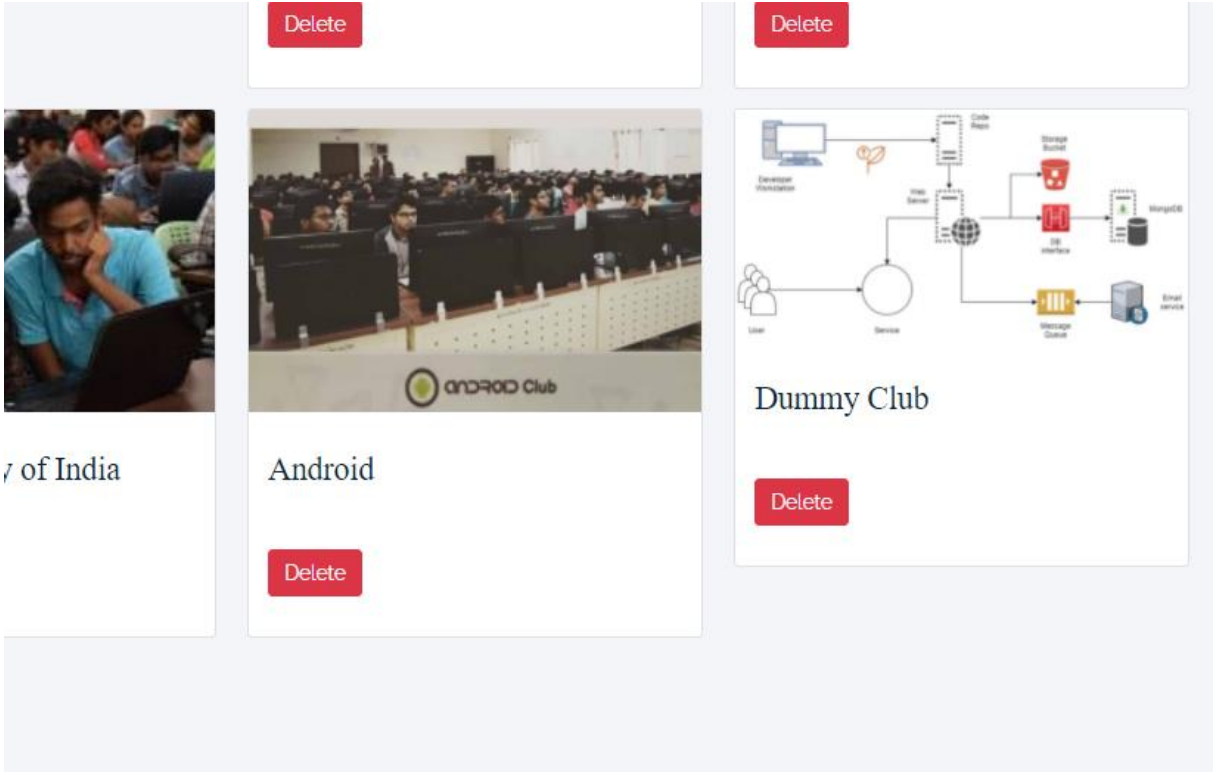
Club Description

Dummy Club generated for testing purpose

After uploading Poster



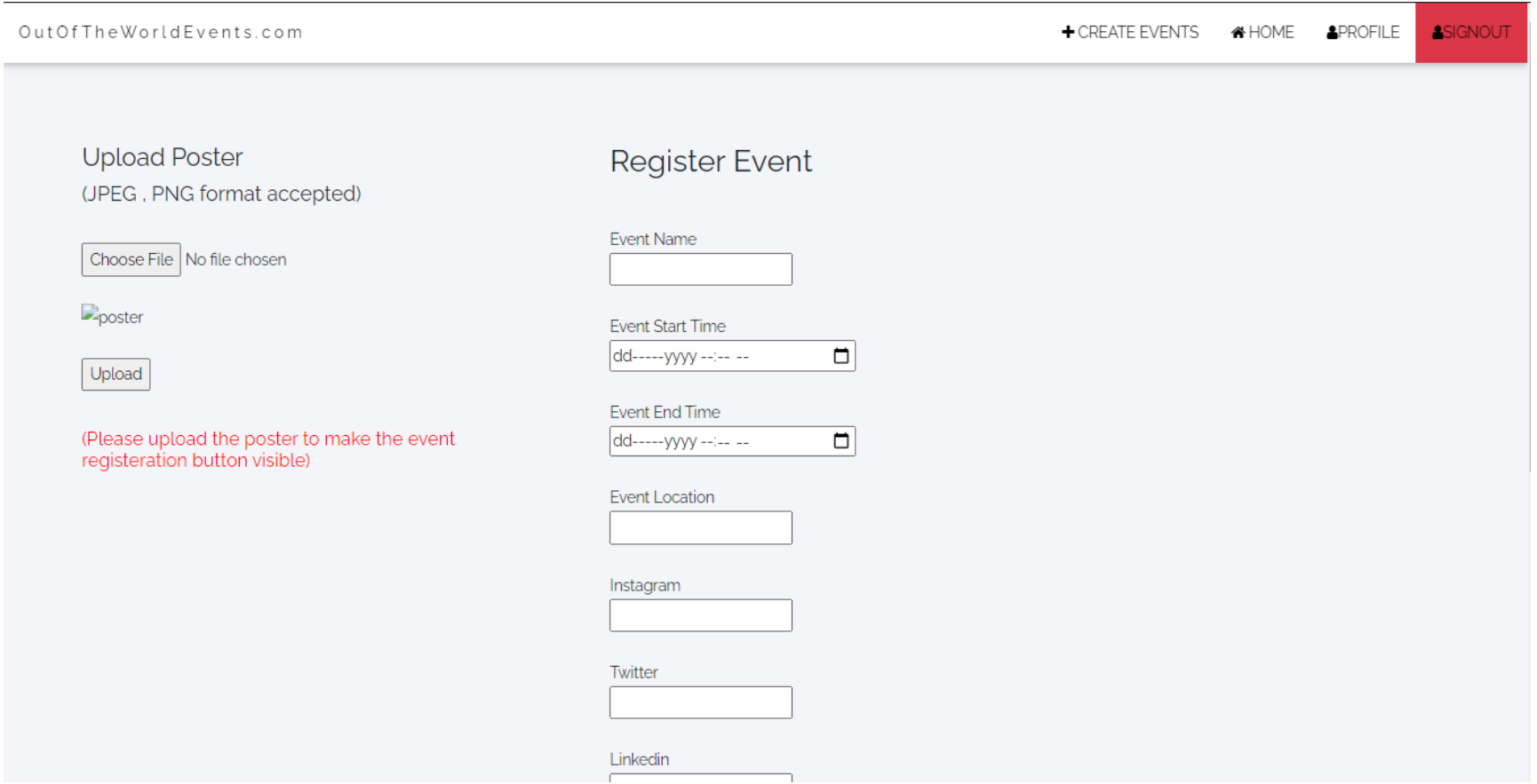
One Time Password Generation



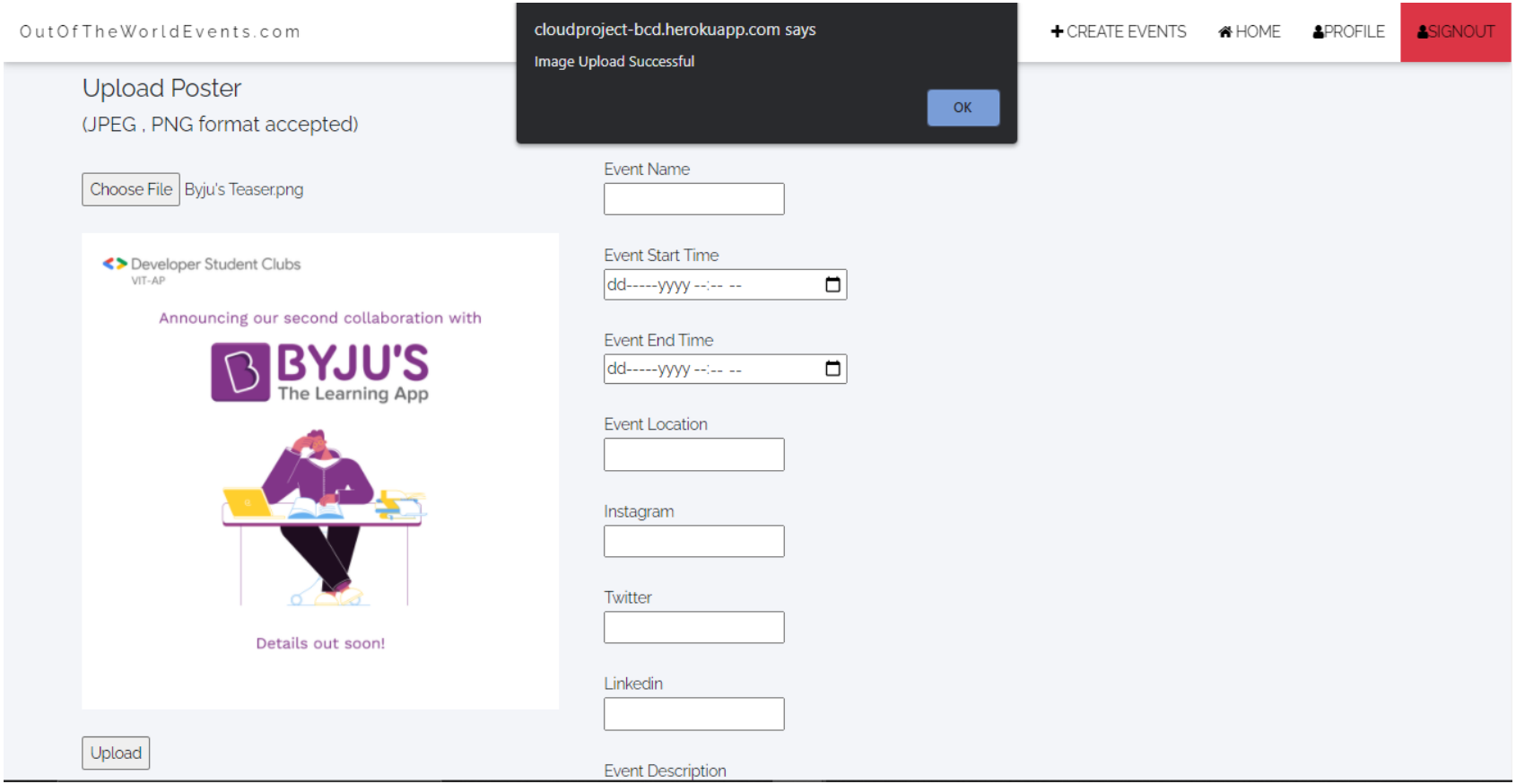
Club Visible in All clubs

Create Event Page

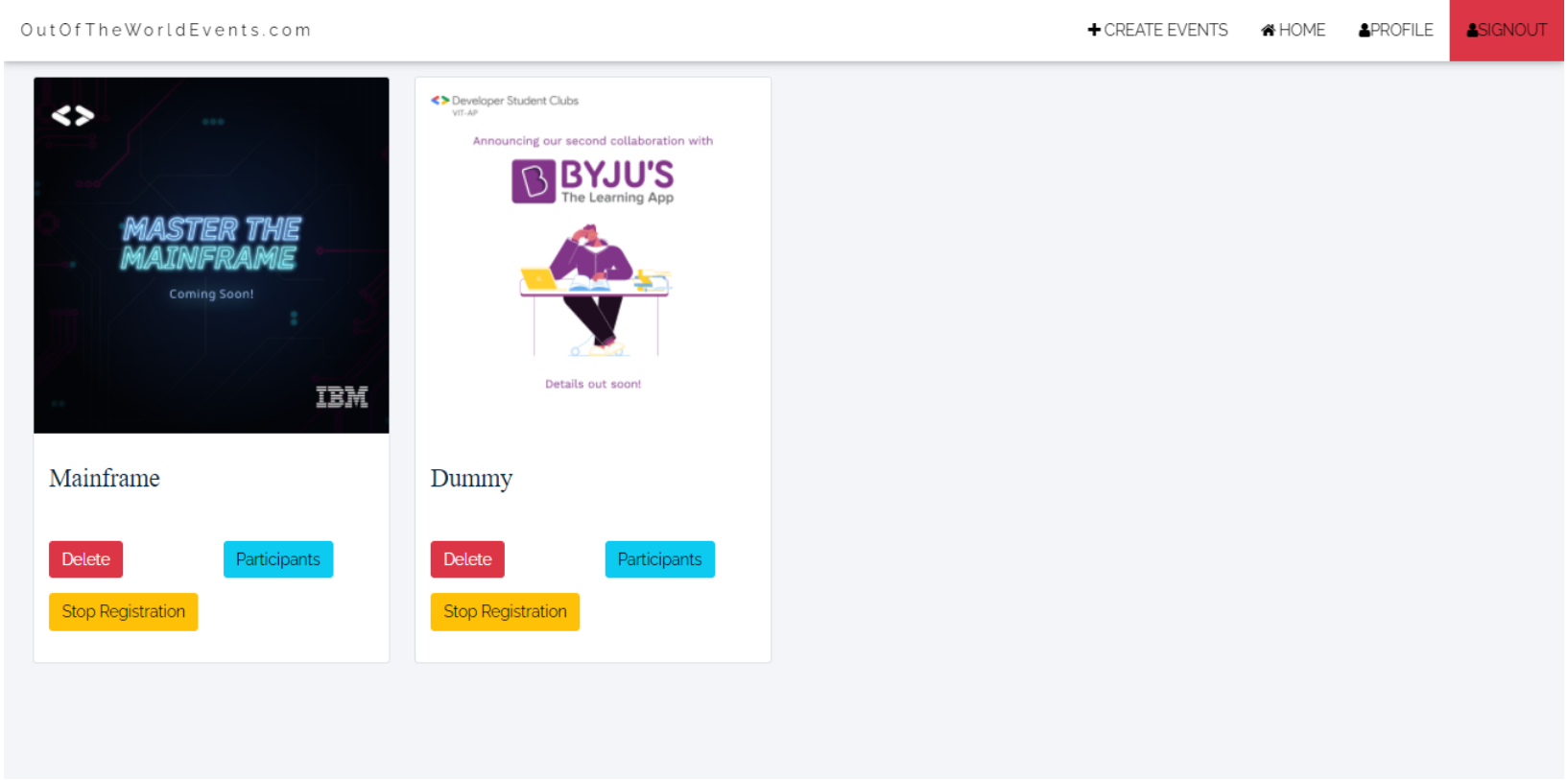
- Clubs need to fill all the required data in the form provided on the webpage.
- Event Poster needs to be uploaded first
- Then all other information can be filled and the form can be submitted.
- Event will be published instantly on all pages.



Event Registration Form



Poster Upload



Event visible to all Users of the platform

Student Home Page

OutOfTheWorldEvents.com

EVENTS CLUBS HOME Profile SIGNOUT

Registered Events

BITBYBIT

GDSC

View More

Club Home Page

- This page provides all the options that are available for a event to the club.
- It displays all the live events for that club.
- They can start and stop registration for a event
- They can delete the event
- They can download the list of participants for that event.
- All queries are handled in the same page except participation list. Here the user is redirected to a new web page that shows the list of all the participants. Also it provides to download the same in excel and csv format.

Live Events

Android Study Jam

Delete Participants

Start Registration

Expert Panel

Delete Participants

Stop Registration

BITBYBIT

Delete Participants

Stop Registration

CN

Delete Participants

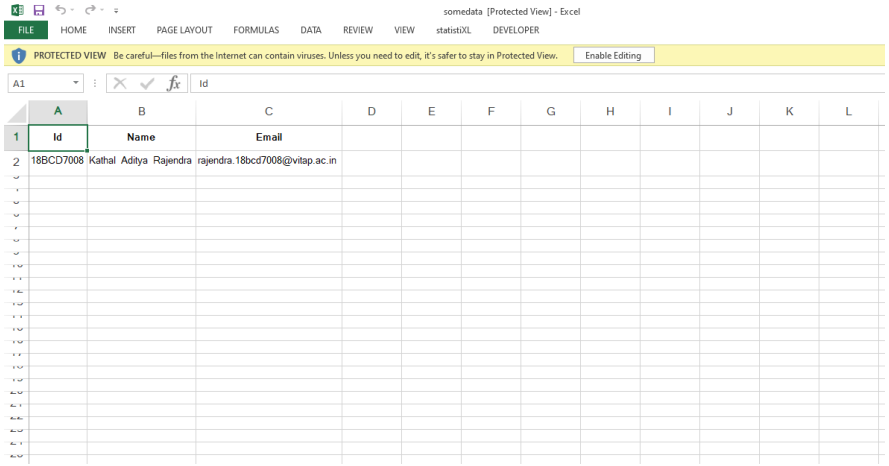
Stop Registration

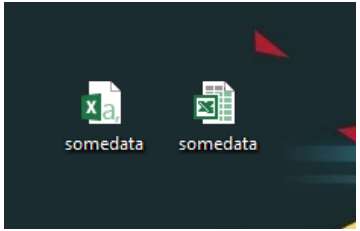
Download Participation List Page

OutOfTheWorldEvents.com

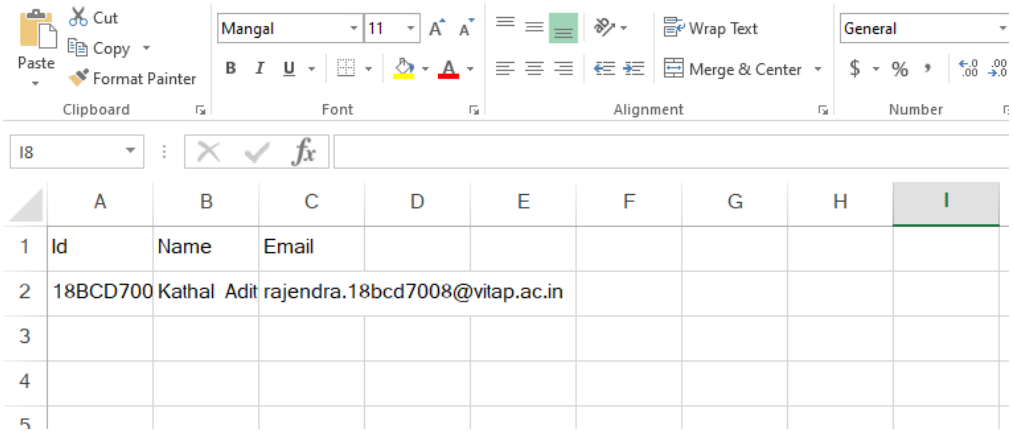
Export to Excel Export to CSV

Id	Name	Email
18BCD7008	Kathal Aditya Rajendra	rajendra.18bcd7008@vitap.ac.in





CSV File



Excel File

Admin Home Page

OutOfTheWorldEvents.com

ADMIN | CREATE CLUB | CHANGE PASSWORD | SIGNOUT

Clubs

Google Developers

GDSC

Delete

Open Source Community

Delete

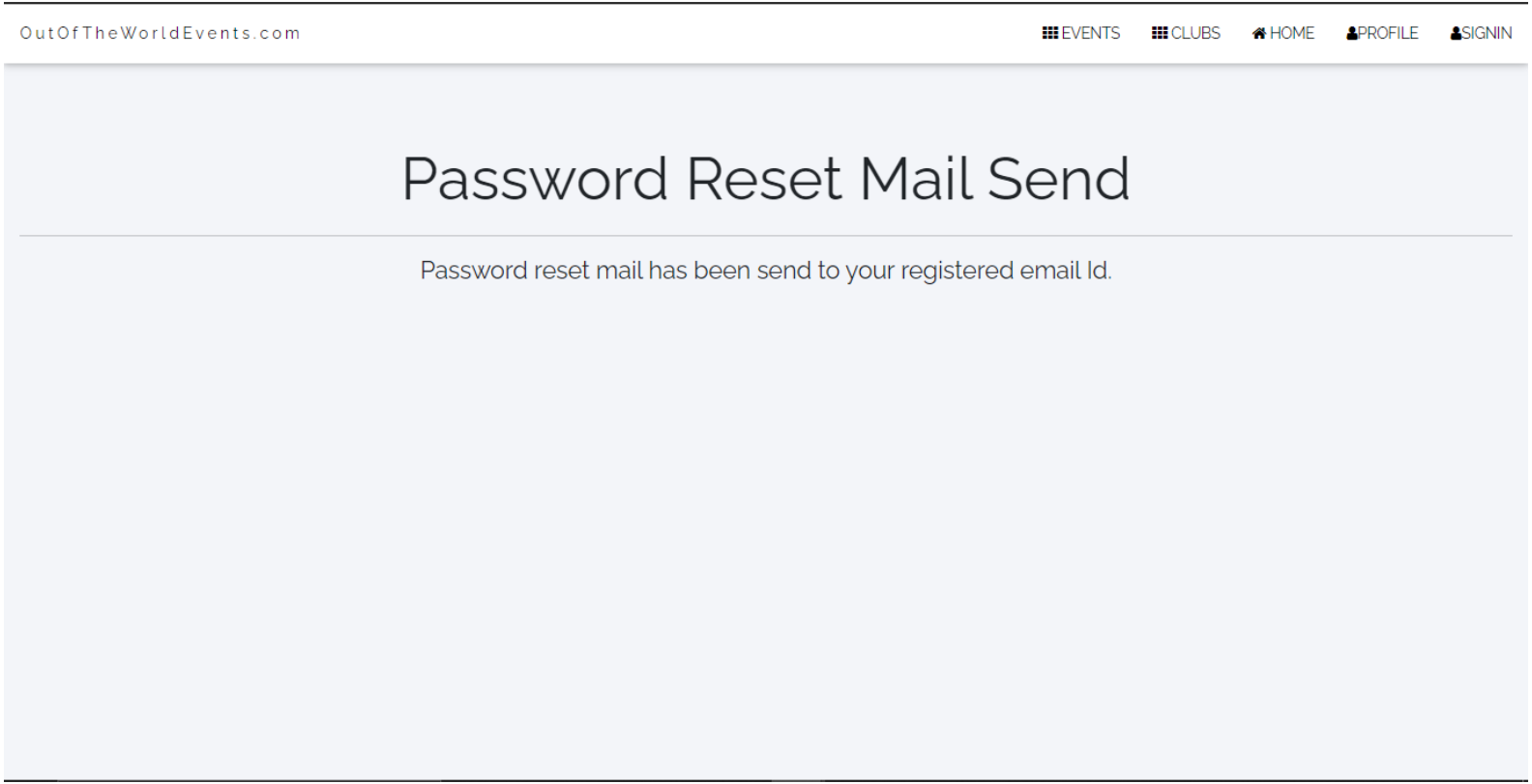
Null Chapter

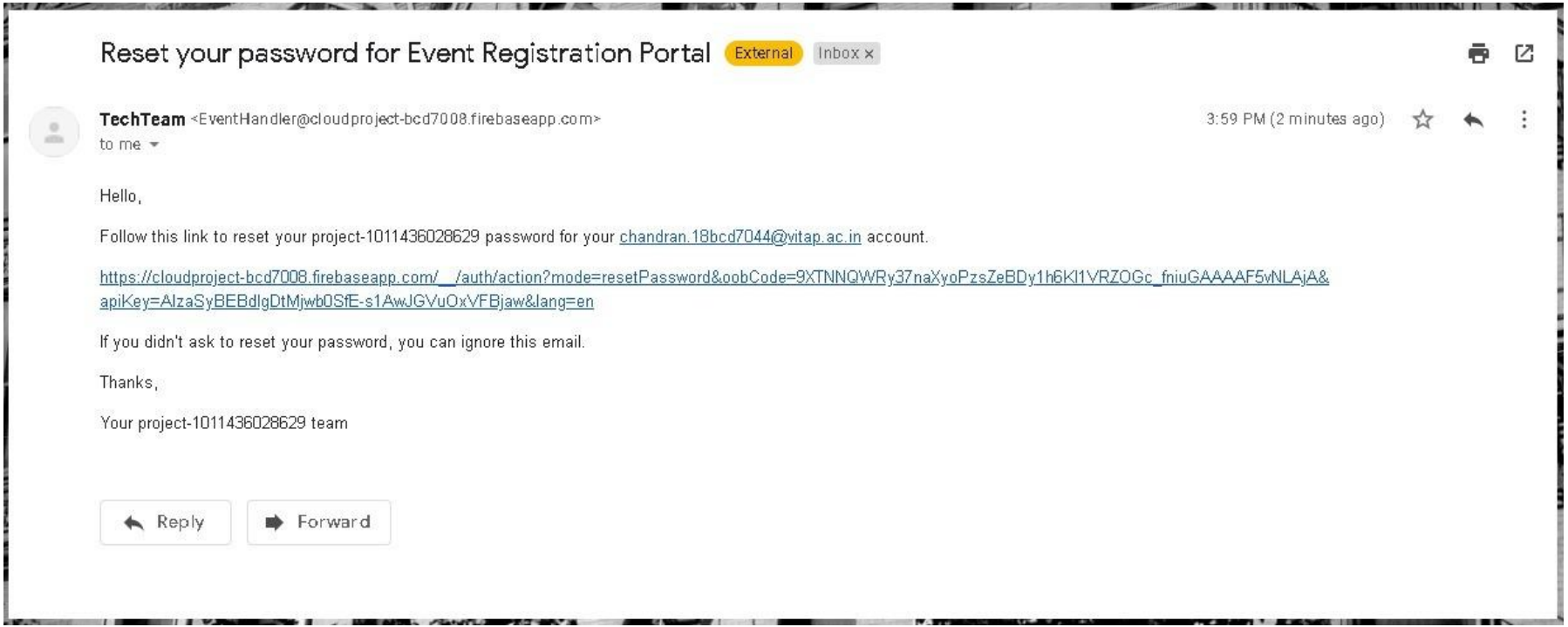
Delete

Be a nerd club

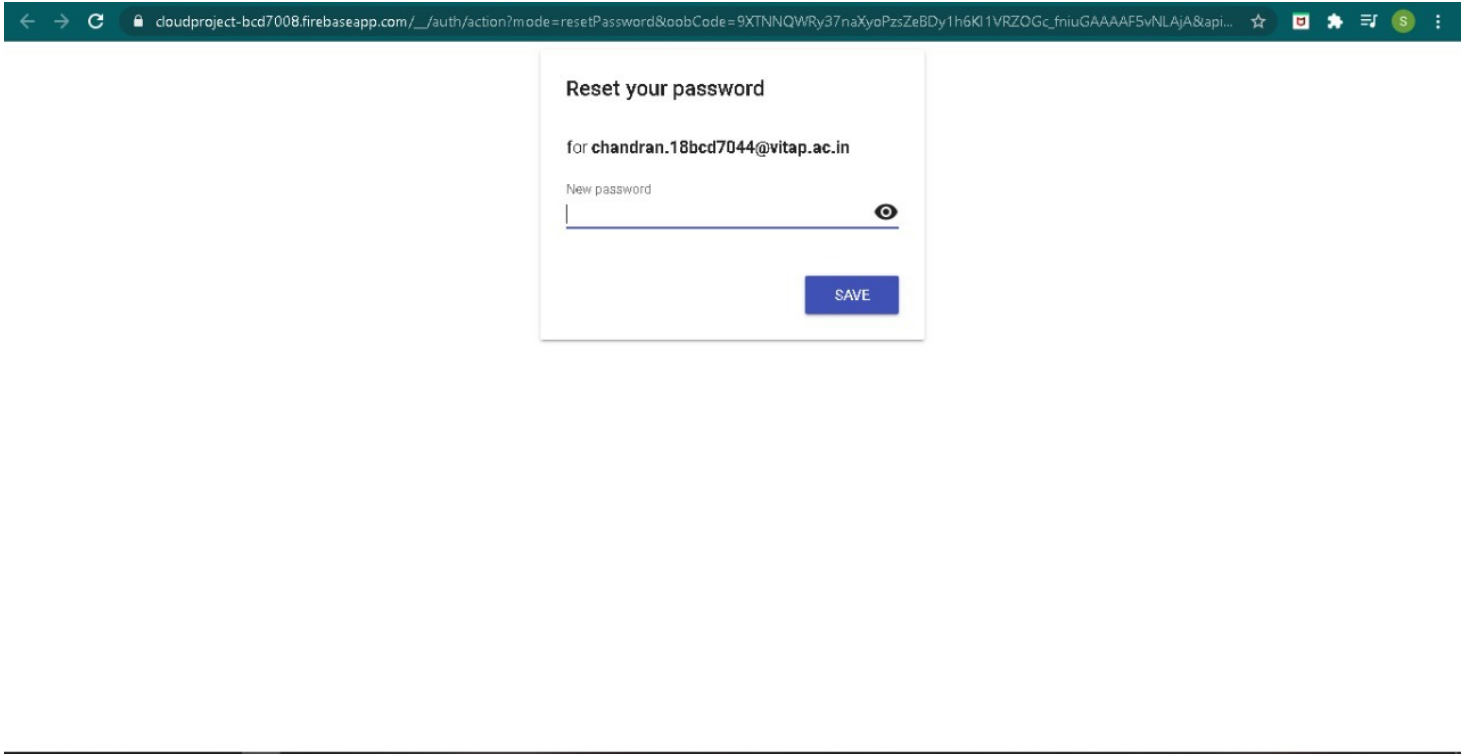
Delete

Change Password Page



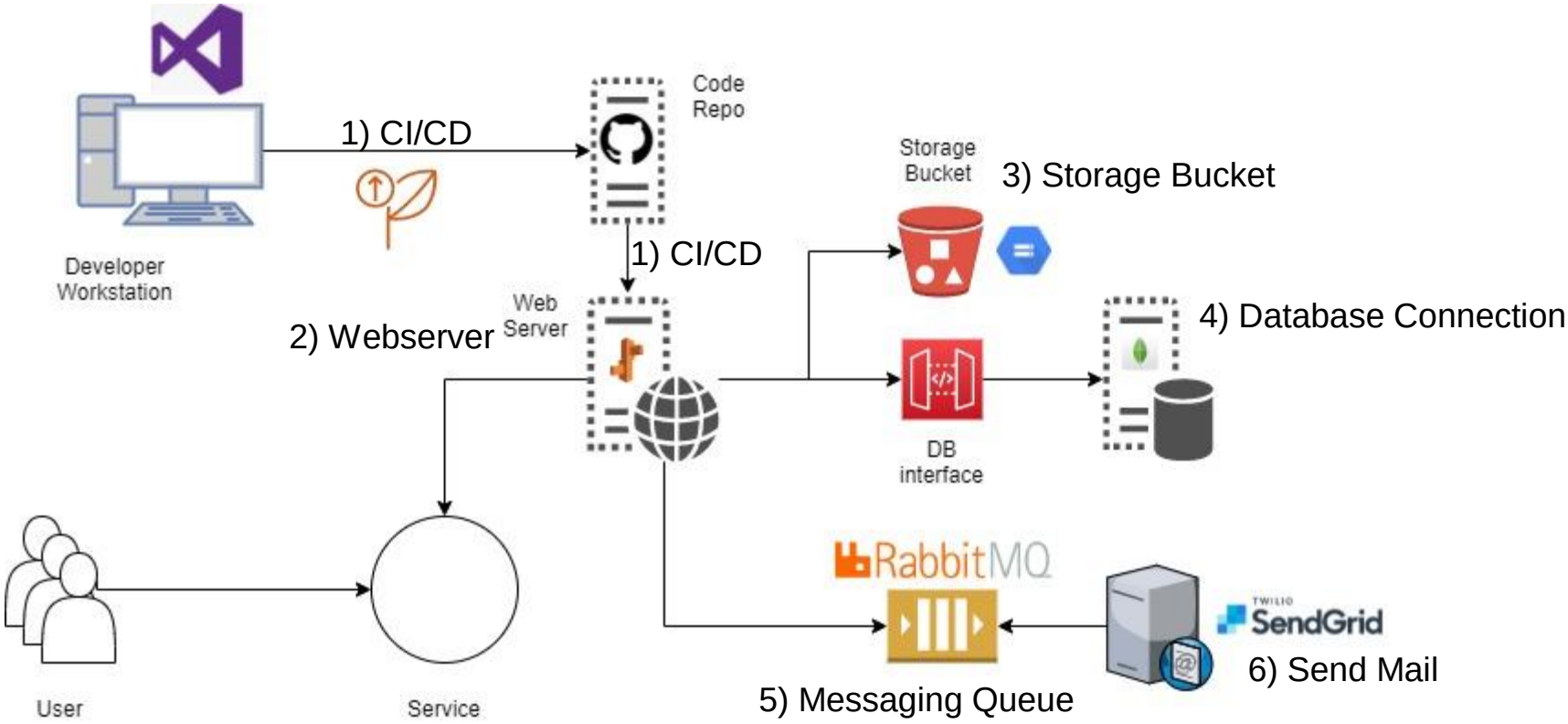


Password Reset Mail



Password Reset Window

Architecture Demo

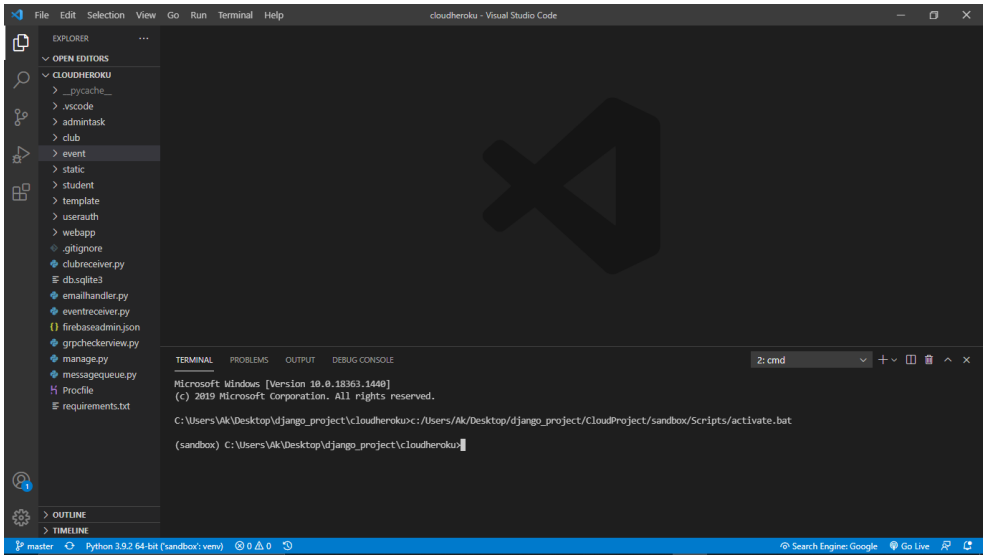

Developer
Workstation
1) CI/CD
Code
Repo
1) CI/CD
Storage
Bucket
3) Storage Bucket
2) Webserver
Web
Server
4) Database Connection
DB
interface
RabbitMQ
SendGrid
6) Send Mail
User
Service
5) Messaging Queue

1) Code Integration and Code Deployment

- For code writing we have Visual Studio Code editor. Visual Studio editor projects a rich interface to control various aspects of the project without ever leaving the IDE.

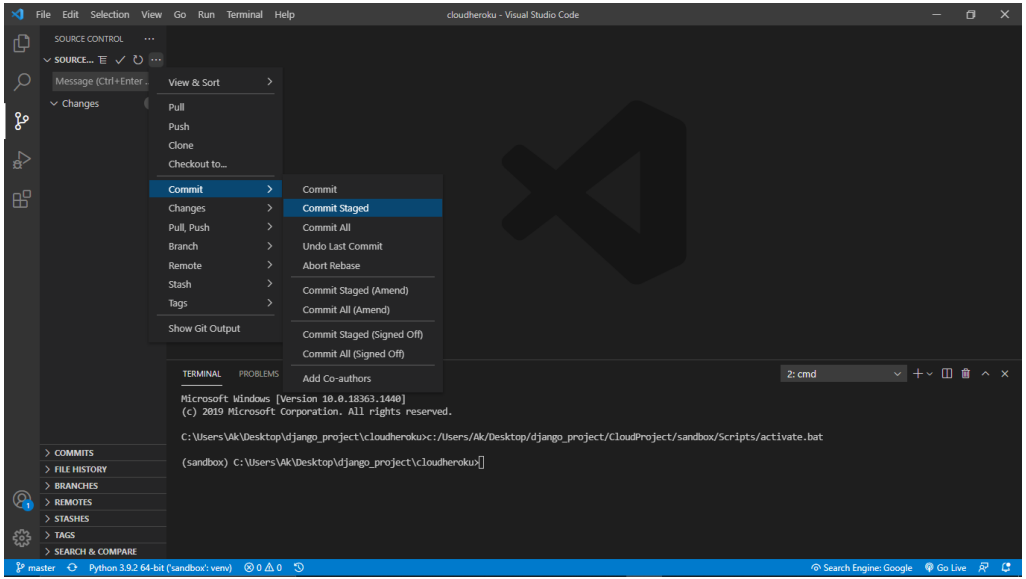
- On the left hand side we can view all the options it provides. From file explorer to GIT integration.

- By connecting the GIT repo to our webserver and visual studio after every commit the webserver restarts itself to include all the changes that have happened in the project structure.



1) Code Integration and Code Deployment

- For code integration we will use GitHub. The work can be divided among team members and different branches can be created.
- Then using the GIT UI we can control branch merge and deal with code changes.
- Since this project had only one developer only one branch was used.
- The screenshot provides a view of the GIT UI present in Visual Studio.



1) Code Integration and Code Deployment

- After all the code required for the pushed to the GitHub, the repository will look something like this.

2.) Deployment on Heroku

- First we will see how to deploy a Django web-app on Heroku. We explore this option of deployment because it is free of cost and great for low traffic application.

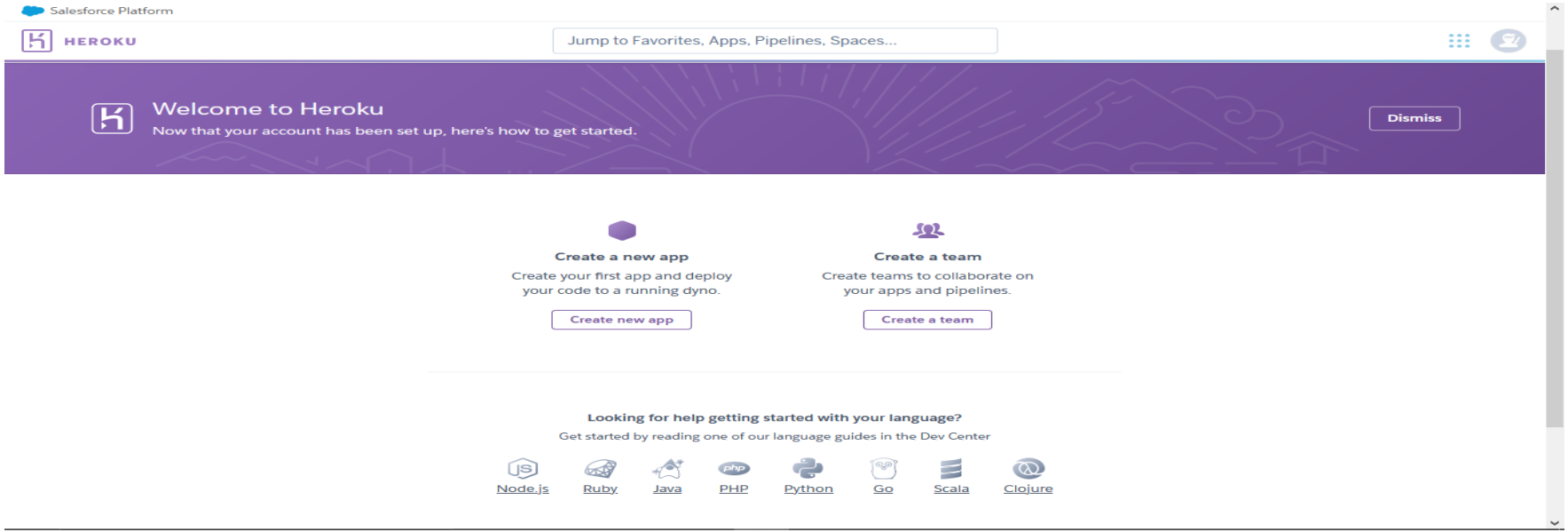
- Heroku follows container model and provides containers called dynos upon which our webapp is installed and executed.

- Free dynos are good for low traffic website. But they have a significant disadvantage when it comes to speed. After 29 mins of inactivity dynos go in sleep mode. So when a new user visits the website after 29 minutes of ideal time the load time of website is very slow.

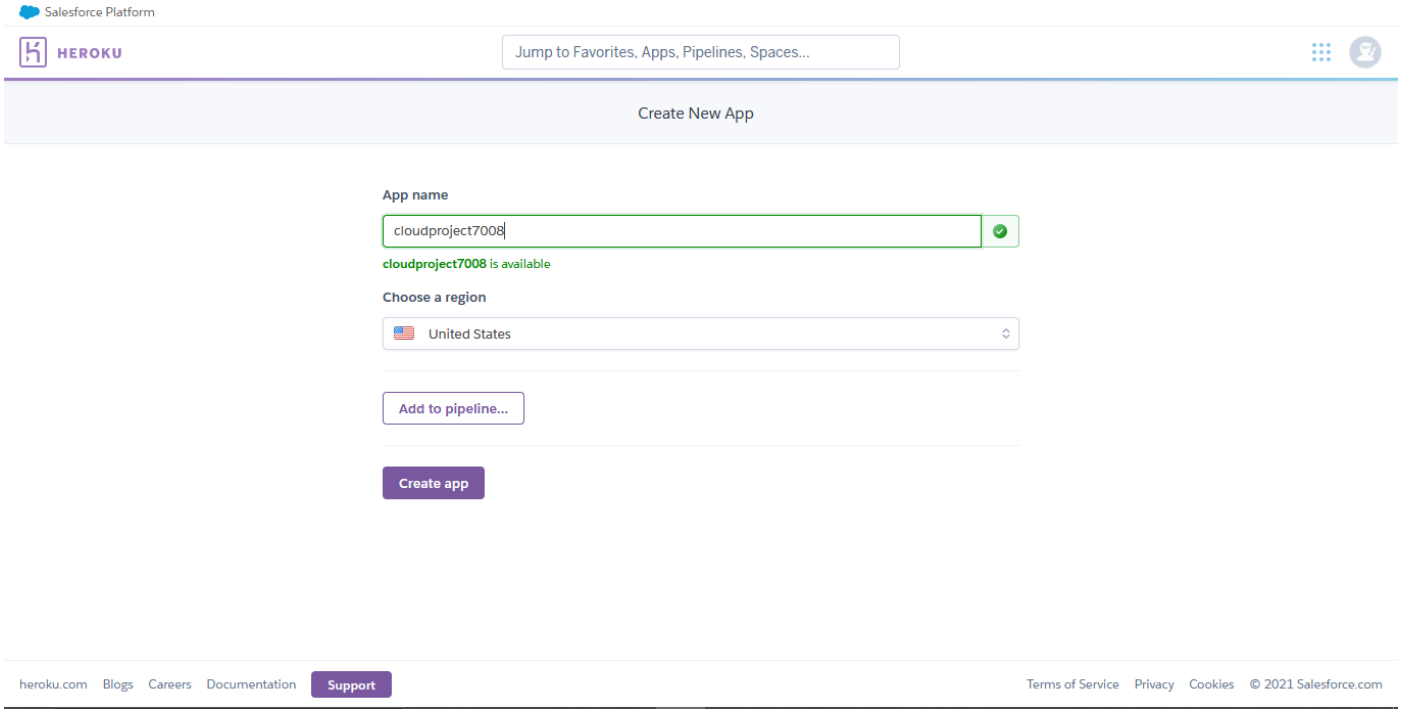
- Another problem that data present is a dyno is not persistent. For low traffic website a developer might use SQLite to store and retrieve data. But every time a dyno restarts it starts from the state in which the code was pushed. Thus the data stored in SQLite is lost. Any kind of changes made to the files from the user end will not be reflected once the dyno restarts.

2.) Deployment on Heroku

- First create a free account on Heroku using a valid email id as you will be receving a verification link on the same email id.
- After login you will be welcomed with this screen.

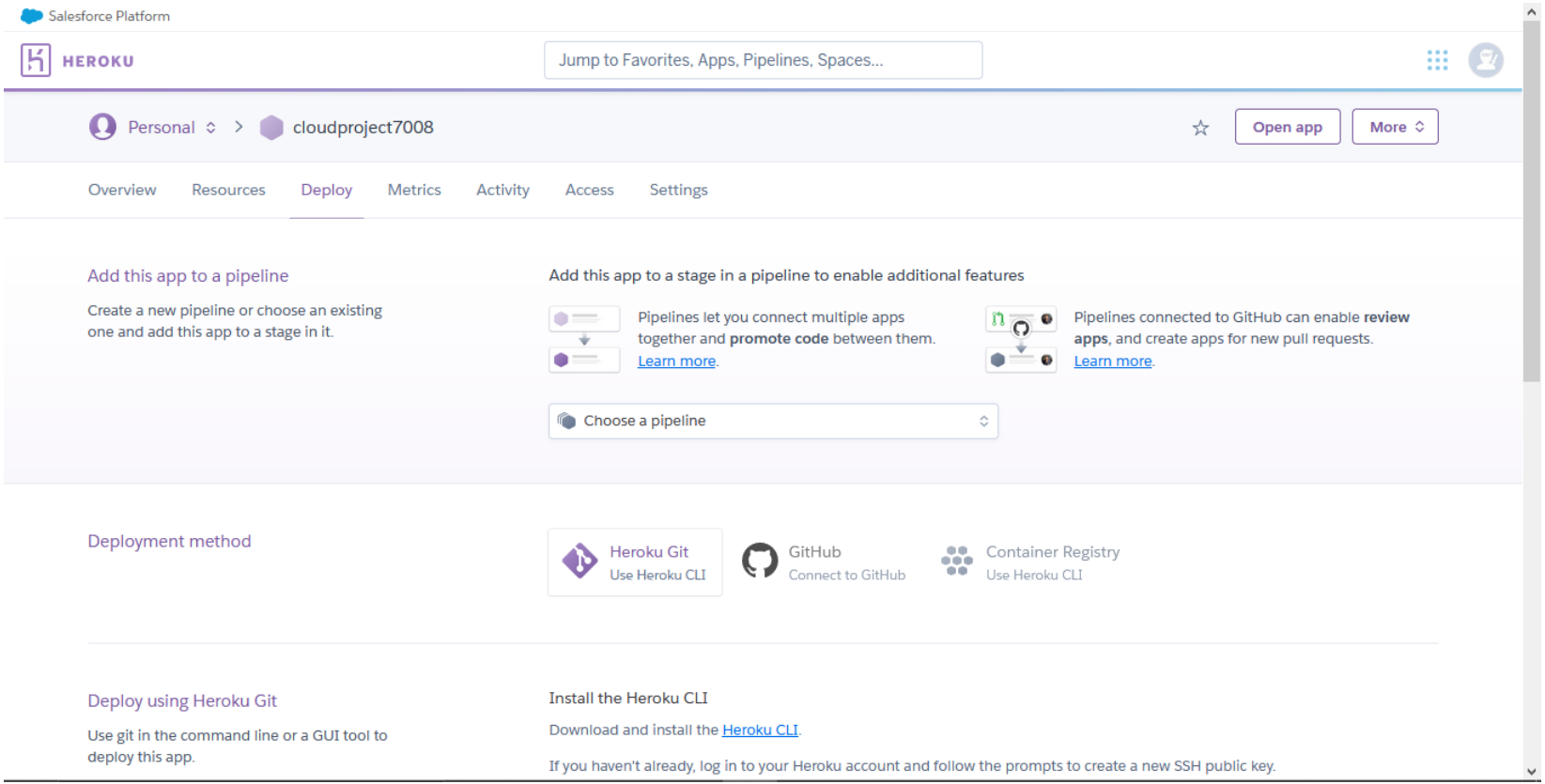


- Choose start a new app option on the previous screen. Then this screen will be loaded.

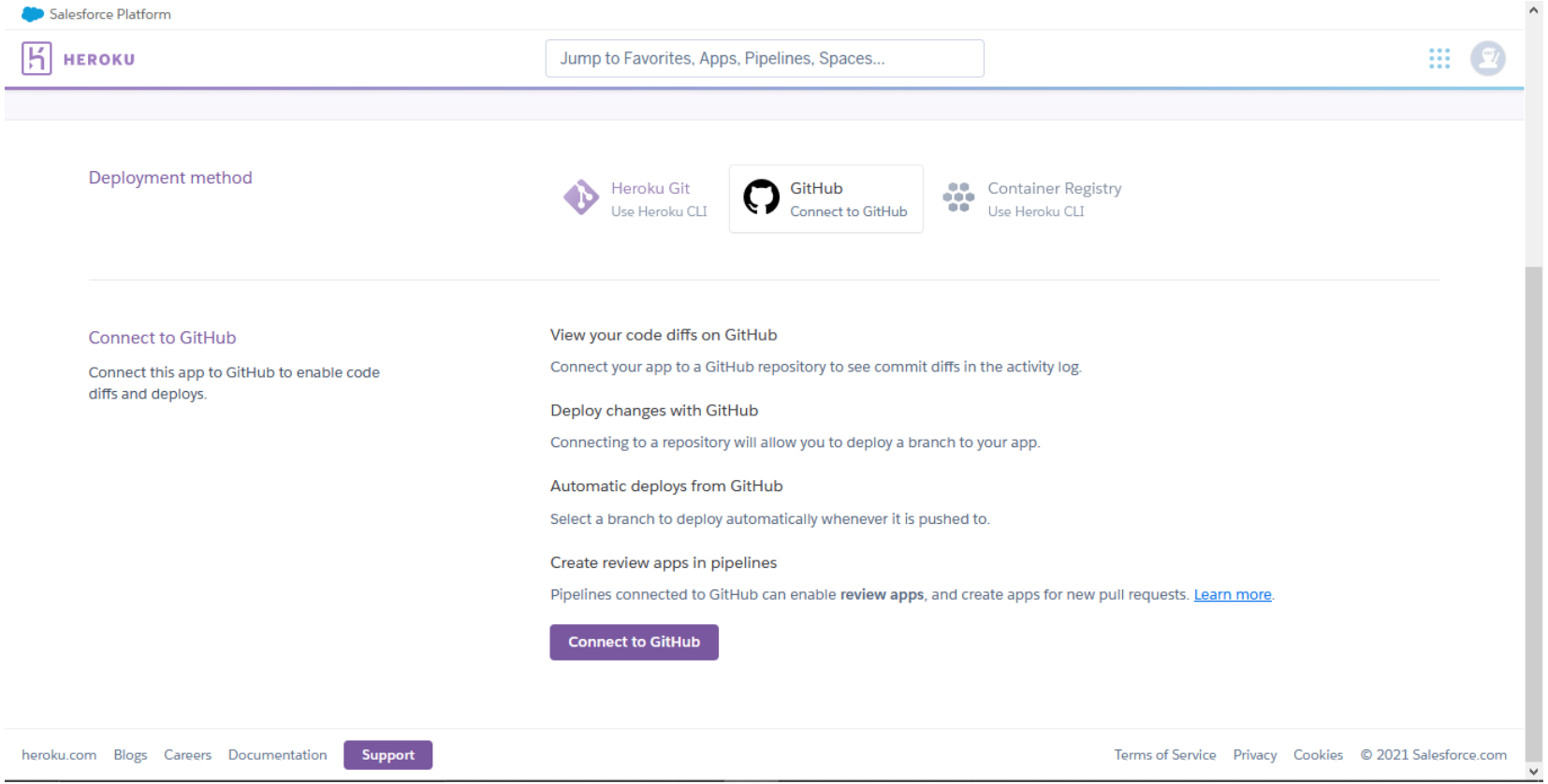


- Enter an app name that is unique all over the world. This will be your dyno name also.

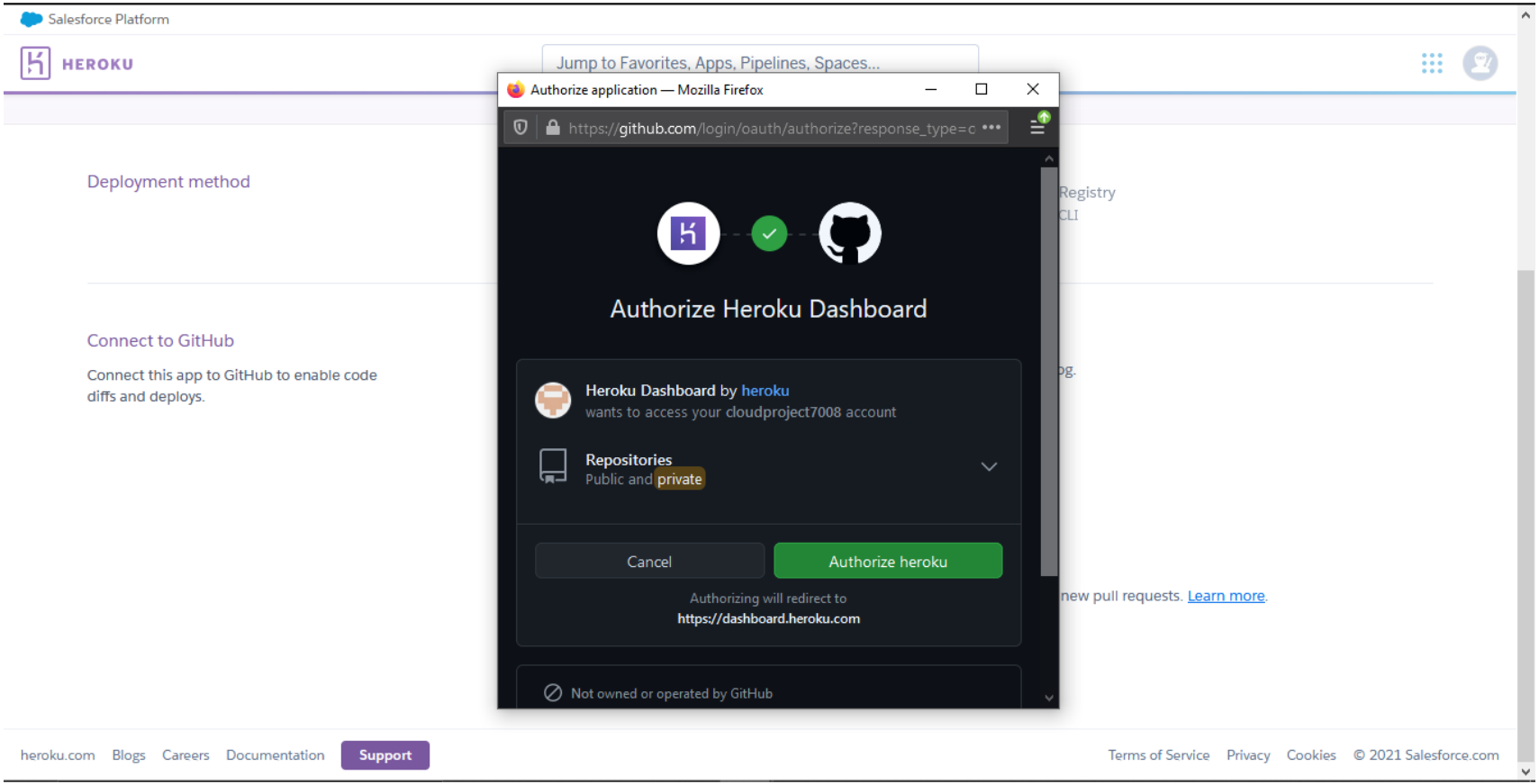
- After clicking “Create App” in the previous window. This window will appear. On the deploy window we need to make few changes to connect to the GitHub repository and start automatic deployment.



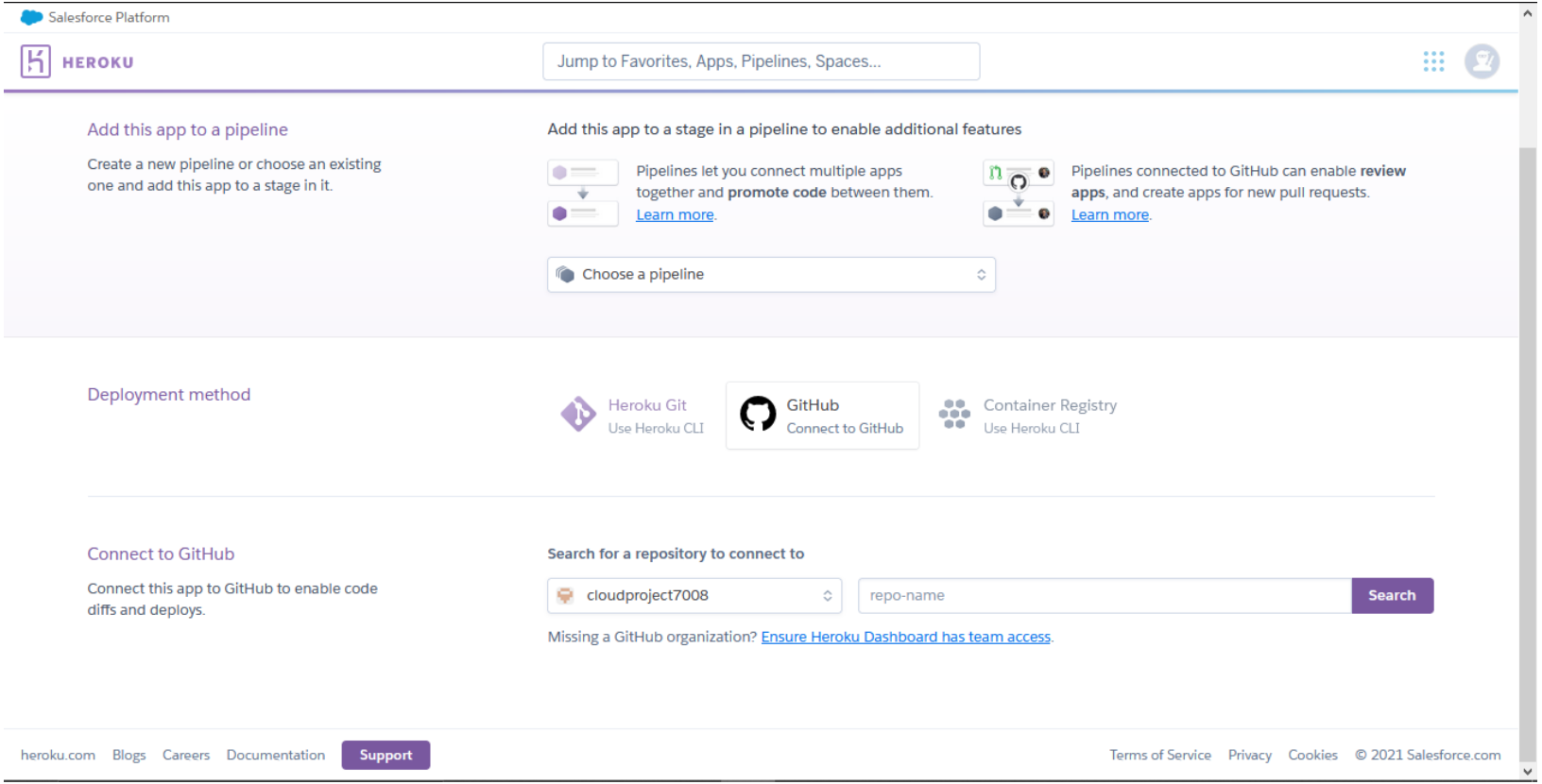
- Under deployment method choose “GitHub” the following window will open. Choose “Connect to GitHub” option. This will open a OAuth page where github will ask permission to let heroku connect to your repository.



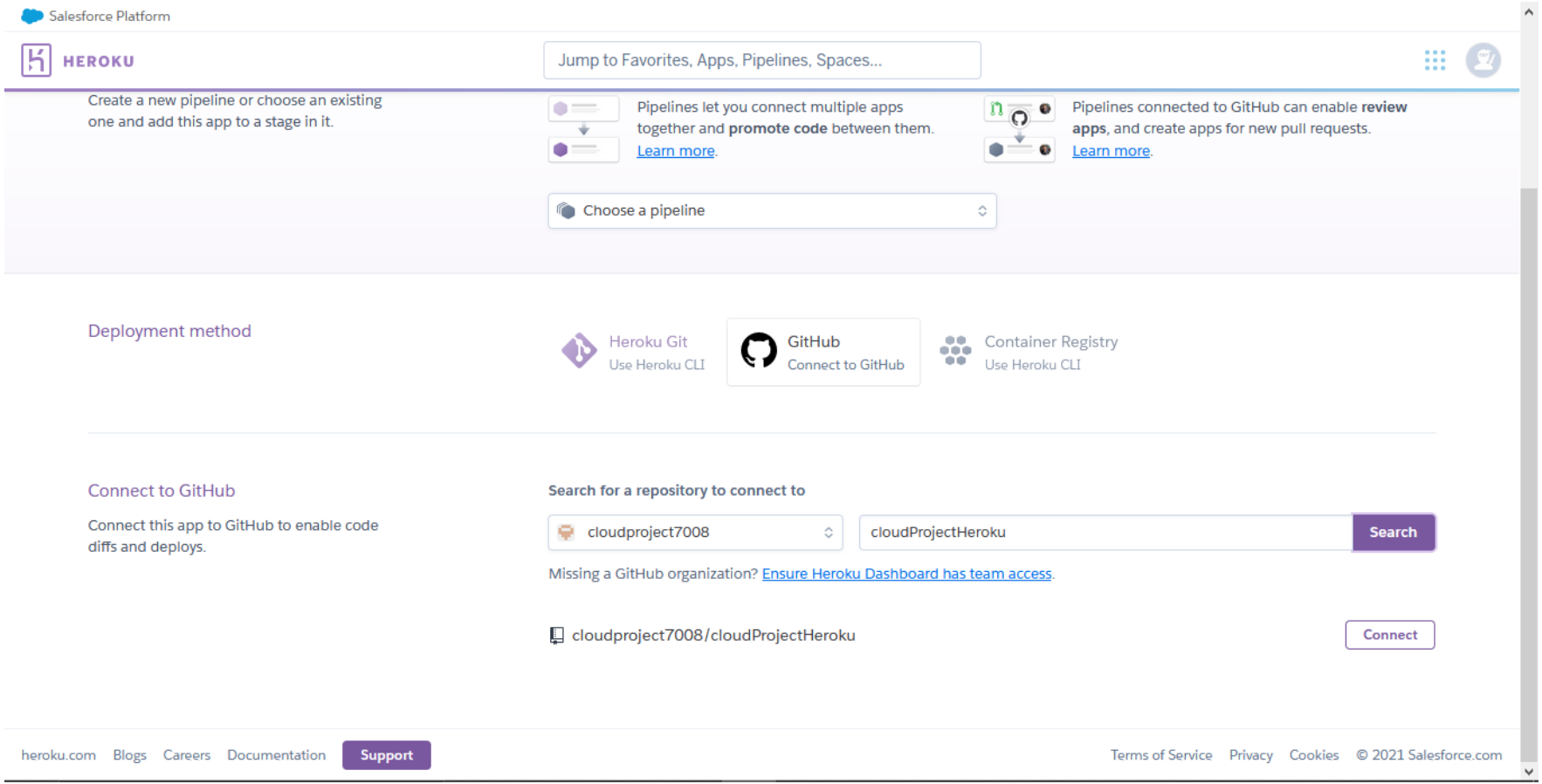
- Click “Authorize Heroku”



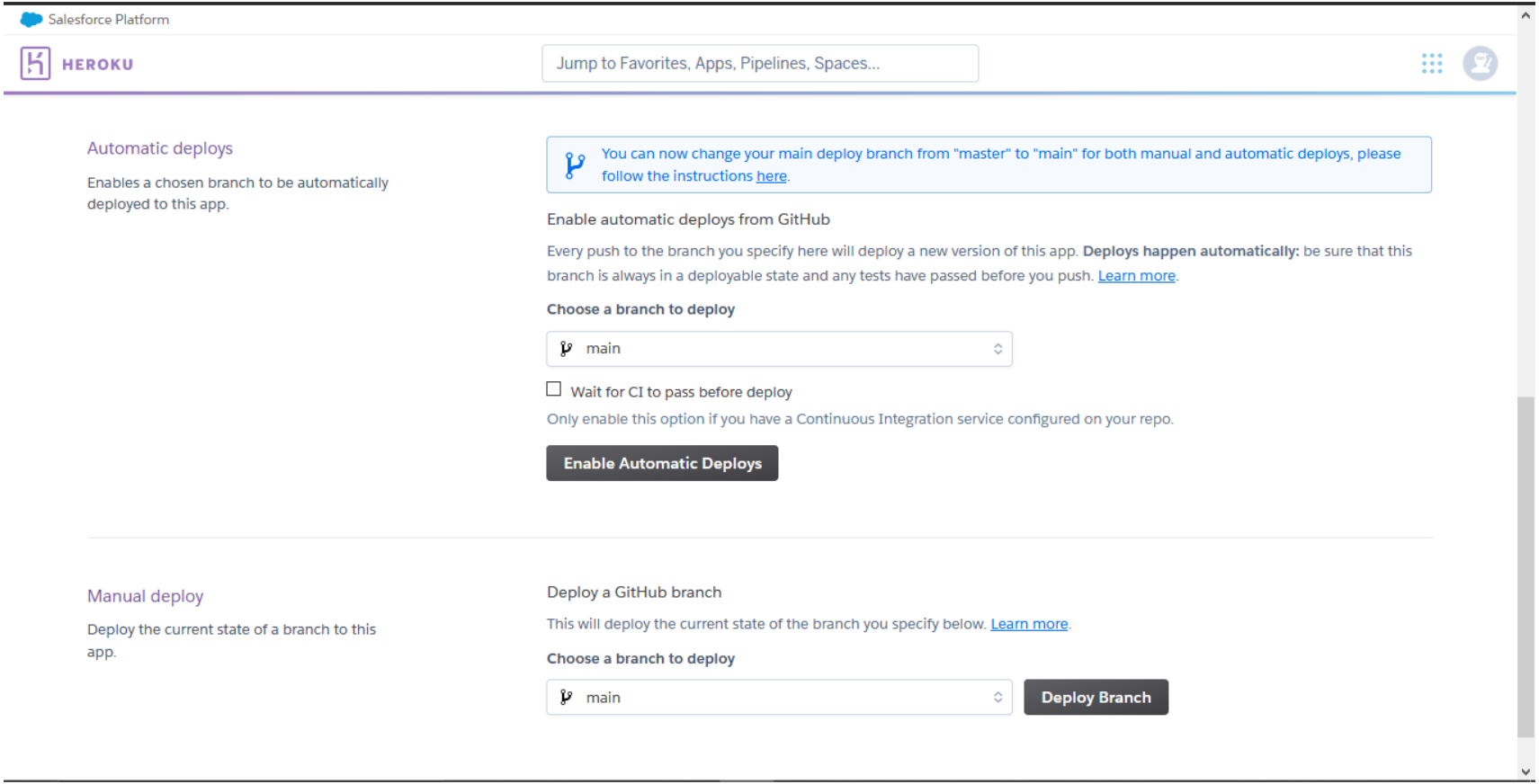
- Under “Connect to Github” Enter the repository name and click search.



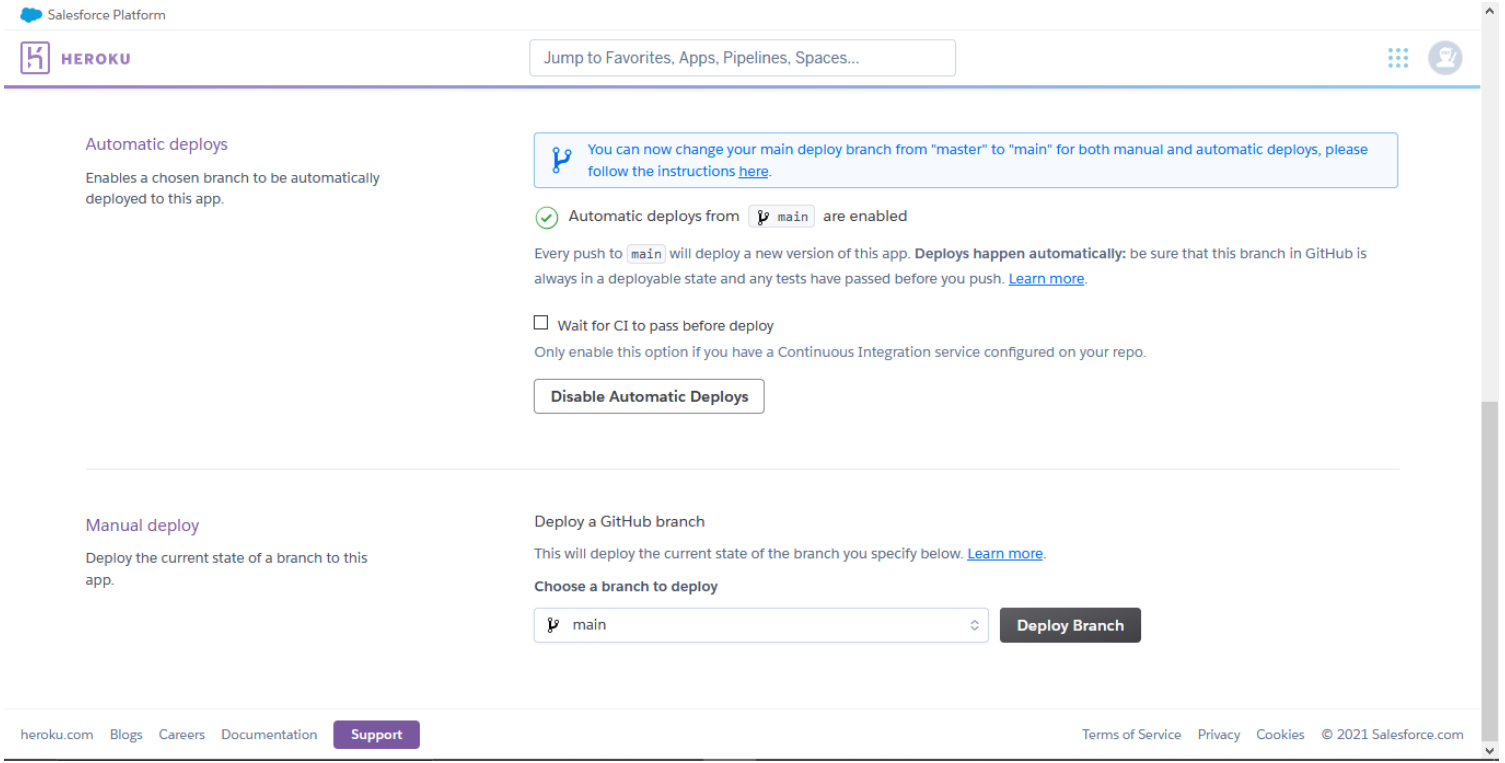
- Click on “connect”



- After successful connection. These options will be visible. Click on “Enable Automatic Deploys”.



- Now the GitHub and Heroku dyno are linked successfully. Whenever we will commit a code on the given branch the dyno will restart to include all changes.



2) Preparation of code for Heroku dyno

- In the current state the dyno will not be able to run our djnago application. This is because we have not defined the Procfile that heroku dyno use to identify the location of WSGI file. This file is used to forward web requests from web servers to web application.

- All a requirements.txt file is all required to tell the dyno which packages to install. Otherwise the code will throw package not found exception.

- To load static files we need a package called “Whitenoise” provided by heroku itself. We need to install this package and all add it to the setting.py file along with static file paths.

- Allowed Host list also needed to be changed to allow the domain provided by heroku. Otherwise error page will be loaded.

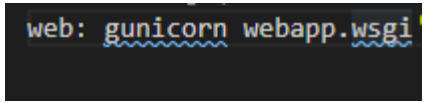
Add requirements.txt

- Run “pip freeze” command. This will give list of all the packages installed in that virtual env.
- Create a txt file named requirements.txt.
- Copy all these packages name and paste in requiements.txt
- Save the file in the same directory as manage.py.

```
(sandbox) C:\Users\Ak\Desktop\django_project\CloudProject>pip freeze
asgiref==3.3.4
awsebcli==3.19.4
botocore==1.20.75
CacheControl==0.12.6
cachetools==4.2.1
cement==2.8.2
certifi==2020.12.5
cffi==1.14.5
chardet==4.0.0
colorama==0.4.3
Django==3.2
dnspython==2.1.0
ebcli==4.0.18
enum34==1.1.10
firebase-admin==4.5.3
future==0.16.0
gcloud==0.18.3
google-api-core==1.26.3
google-api-python-client==2.1.0
google-auth==1.28.0
google-auth-httplib2==0.1.0
google-cloud-core==1.6.0
google-cloud-firestore==2.1.0
google-cloud-storage==1.37.1
google-crc32c==1.1.2
google-resumable-media==1.2.0
googleapis-common-protos==1.53.0
grpcio==1.37.0
gunicorn==20.1.0
httplib2==0.19.1
idna==2.10
Jinja2==3.0.1
jmespath==0.10.0
jws==0.1.3
keyring==9.0
```

Adding Procfile

- First run “pip install gunicorn”
- Create a file named Procfile. Do not give any dot extension to this file. It should be named only.
- Write the code written in the screenshot in the Procfile.
- It is the location to the wsgi.py file. Gunicorn is required because it is a WSGI HTTP server.



web defines the application type , guicorn is the WSGI HTTP server and webapp.wsgi is the Django application wsgi settings file location.

Adding WhiteNoise

- First run “pip install whitenoise”
- Add the following lines in the middleware dictionary of the settings.py file. This instructs the Django application to use whitenoise while loading static file.
- Add these lines of code also.
- Staticfiles dirs tells the code where the static files are located and the static root is the directory that will be created in the dyno when it runs collectstatic.

```
(sandbox) C:\Users\Ak\Desktop\django_project\CloudProject>pip install whitenoise
Requirement already satisfied: whitenoise in c:\users\ak\desktop\django_project\cl
5.2.0)
```

```
MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
    'whitenoise.middleware.WhiteNoiseMiddleware',
]
```

```
STATICFILES_DIRS = [ os.path.join(BASE_DIR , "static"),]
STATIC_ROOT = os.path.join(BASE_DIR , "staticfiles")
STATICFILES_STORAGE = 'whitenoise.storage.CompressedManifestStaticFilesStorage'
```

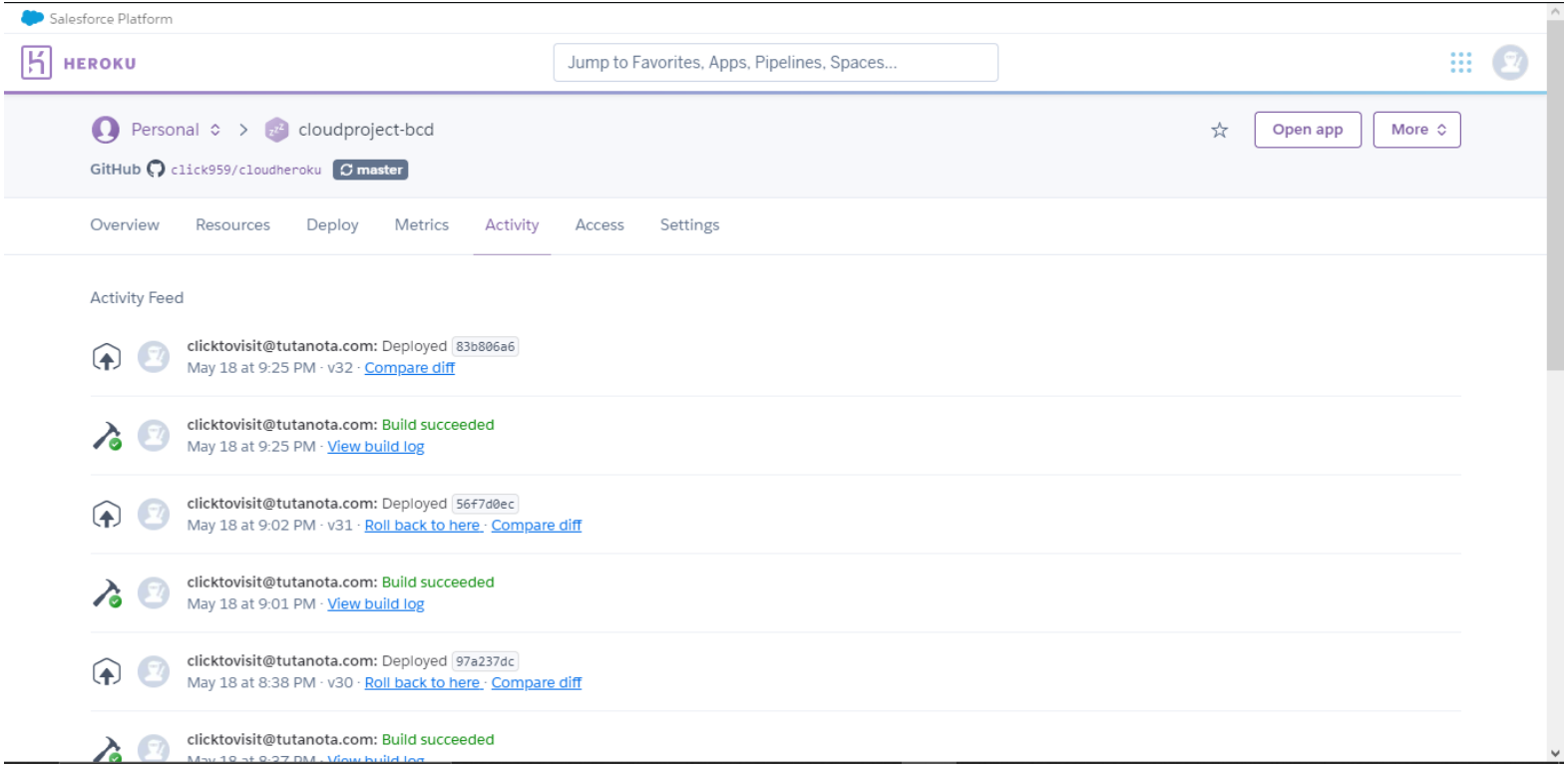
Allowed host

- In settings.py file a list is present at the top called ALLOWED_HOSTS. Add your heroku domain to that list or for general purpose add ‘*’ to the list.

```
ALLOWED_HOSTS = ['cloudproject-bcd.herokuapp.com']
```

Now the code is ready to be deployed on Heroku. Just commit the code , as we have already turned on the automatic deploys as soon as you push the code , dyno starts to load the webapp.

This is how the activity tab will look when you push the code.



2) Deploy the application on AWS Beanstalk

- We can deploy our application to AWS using EC2 , but that takes a lot of time. We need to configure the network , add path to the files etc.
- Using AWS Beanstalk one can directly deploy their web app using CLI
- The steps involved in deploying a Django webapp on beanstalk are discussed in the next slides.

Steps

- Create a virtual environment in python using the command virtualenv <env_name>
- Activate the virtual environment by running the activate.bat present in Scripts folder.
- Using the requirements.txt we created for heroku run the command:
    - pip install –r requirements.txt

```
(sandbox) C:\Users\Ak\Desktop\django_project\CloudAWS\webapp>pip install -r requirements.txt
Requirement already satisfied: asgiref==3.3.4 in c:\users\ak\desktop\django_project\cloudaws\sandbox\lib\site-packages (from -r requirements.txt (li
ne 1)) (3.3.4)
Requirement already satisfied: botocore==1.20.75 in c:\users\ak\desktop\django_project\cloudaws\sandbox\lib\site-packages (from -r requirements.txt
(line 2)) (1.20.75)
Requirement already satisfied: CacheControl==0.12.6 in c:\users\ak\desktop\django_project\cloudaws\sandbox\lib\site-packages (from -r requirements.t
xt (line 3)) (0.12.6)
Requirement already satisfied: cachetools==4.2.1 in c:\users\ak\desktop\django_project\cloudaws\sandbox\lib\site-packages (from -r requirements.txt
(line 4)) (4.2.1)
Requirement already satisfied: cement==2.8.2 in c:\users\ak\desktop\django_project\cloudaws\sandbox\lib\site-packages (from -r requirements.txt (lin
e 5)) (2.8.2)
Requirement already satisfied: certifi==2020.12.5 in c:\users\ak\desktop\django_project\cloudaws\sandbox\lib\site-packages (from -r requirements.txt
 (line 6)) (2020.12.5)
Requirement already satisfied: cffi==1.14.5 in c:\users\ak\desktop\django_project\cloudaws\sandbox\lib\site-packages (from -r requirements.txt (line
 7)) (1.14.5)
```

Steps

- Install AWS EB CLI. This can be done using the pip command
    - pip install awsebcl

```
(sandbox) C:\Users\Ak\Desktop\django_project\CloudAWS\webapp>pip install awsebcli
Collecting awsebcli
  Using cached awsebcli-3.19.4-py3-none-any.whl
Requirement already satisfied: colorama<0.4.4,>=0.2.5 in c:\users\ak\desktop\django_project\cloudaws\sandbox\lib\site-packages (from awsebcli) (0.4.
3)
Requirement already satisfied: semantic-version==2.8.5 in c:\users\ak\desktop\django_project\cloudaws\sandbox\lib\site-packages (from awsebcli) (2.8
.5)
Requirement already satisfied: six<1.15.0,>=1.11.0 in c:\users\ak\desktop\django_project\cloudaws\sandbox\lib\site-packages (from awsebcli) (1.14.0)
Requirement already satisfied: wcwidth<0.2.0,>=0.1.7 in c:\users\ak\desktop\django_project\cloudaws\sandbox\lib\site-packages (from awsebcli) (0.1.9
)
```

```
(sandbox) C:\Users\Ak\Desktop\django_project\CloudAWS\webapp>eb --version
EB CLI 3.19.4 (Python 3.9.2)

(sandbox) C:\Users\Ak\Desktop\django_project\CloudAWS\webapp>
```

Steps

- Now we need to create a folder called .ebextensions. Usually the beanstalk program searches for an application file. But Django has no such file. So this extension contains the django.config file. This setting, WSGIPath, specifies the location of the WSGI script that Elastic Beanstalk uses to start your application.

- Run “mkdir .ebextensions”

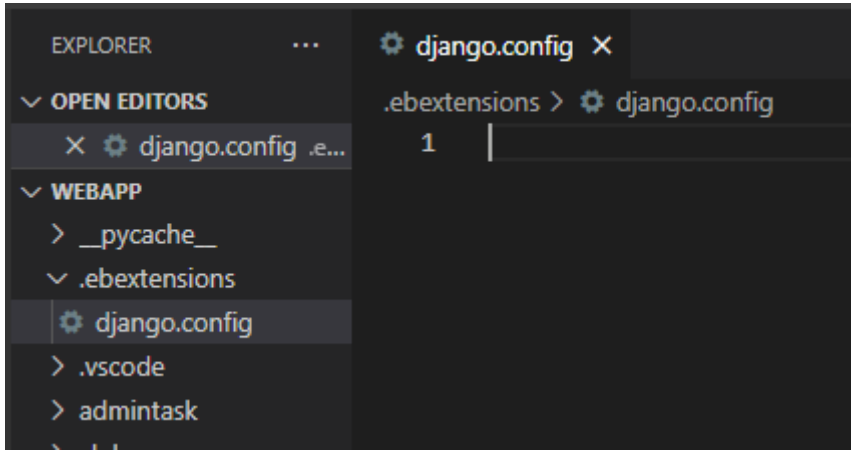
```
(sandbox) C:\Users\Ak\Desktop\django_project\CloudAWS\webapp>mkdir .ebextensions

(sandbox) C:\Users\Ak\Desktop\django_project\CloudAWS\webapp>
```

- Then run “cd .ebextensions” to change to that folder.

```
(sandbox) C:\Users\Ak\Desktop\django_project\CloudAWS\webapp>cd .ebextensions

(sandbox) C:\Users\Ak\Desktop\django_project\CloudAWS\webapp\.ebextensions>
```

Steps

- Create a django.config file using file explorer



- Give the location to your project WSGI.py file

```
option_settings:
  aws:elasticbeanstalk:container:python:
    WSGIPath: webapp/wsgi.py
```

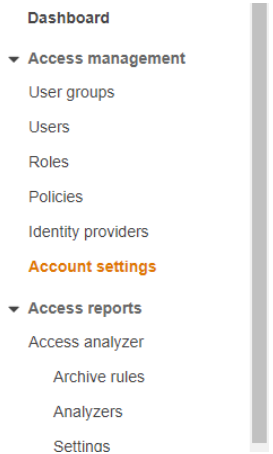
Steps

```
C:\Users\Ak\Desktop\django_project\CloudAWS\webapp\.ebextensions>pip3 --version
pip 21.0.1 from c:\users\ak\appdata\local\programs\python\python39\lib\site-packages\pip (python 3.

C:\Users\Ak\Desktop\django_project\CloudAWS\webapp\.ebextensions>python --version
Python 3.9.2
```

- Deactivate the virtual environment using “deactivate” command.
- Check pip and python version

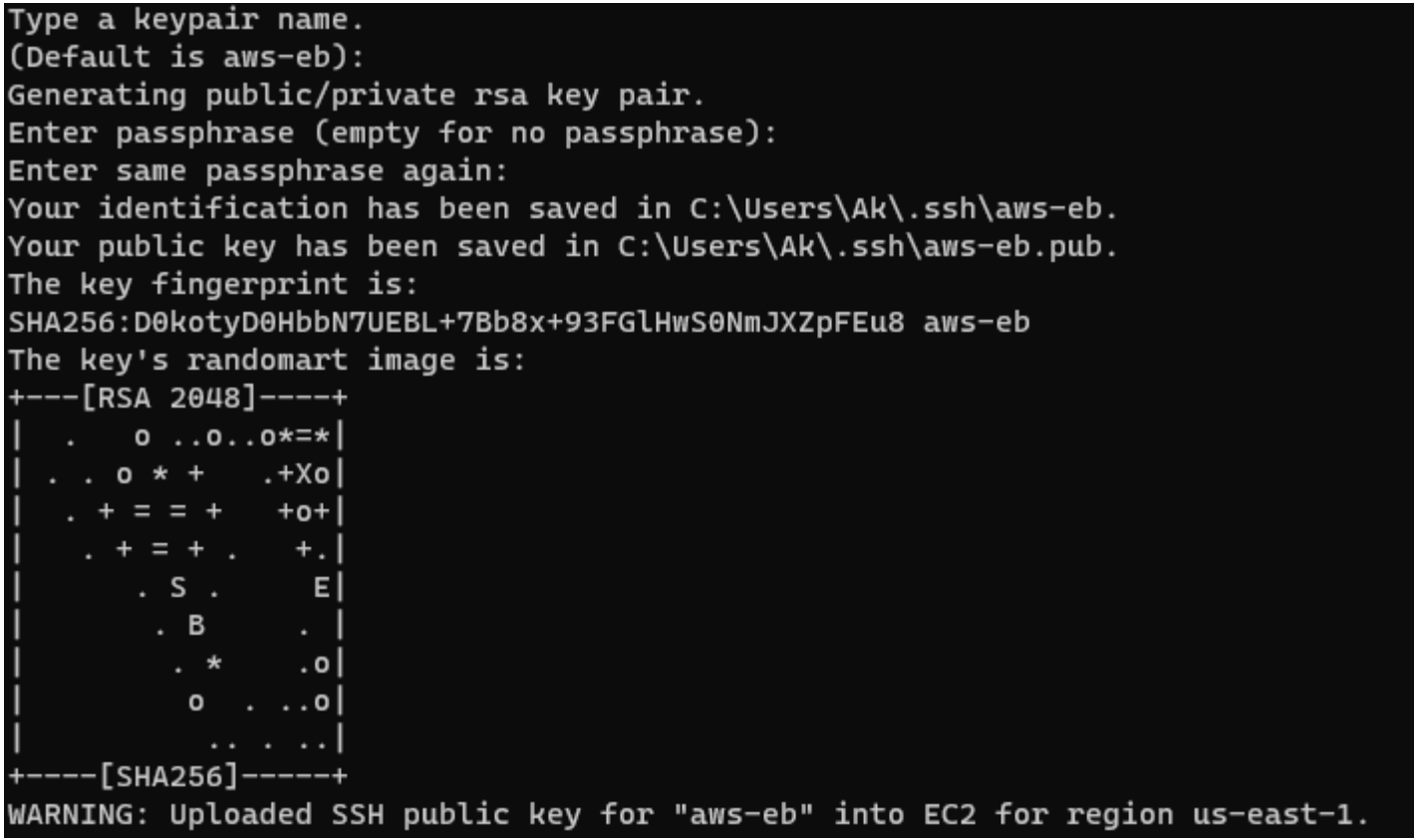
```
C:\Users\Ak\Desktop\django_project\CloudAWS\webapp\.ebextensions>pip install awsebcli
Collecting awsebcli
  Using cached awsebcli-3.19.4-py3-none-any.whl
Collecting termcolor==1.1.0
  Using cached termcolor-1.1.0-py3-none-any.whl
```

- Install awsebcli using “pip install awsebcli”
- Login into your AWS account
- Search IAM.
- In IAM window you will see Account Setting.
- Here you will find your CLI connection key



```
Enter Application Name
(default is "webapp"): cloudProject
Application cloudProject has been created.

It appears you are using Python. Is this correct?
(Y/n): Y
Select a platform branch.
1) Python 3.8 running on 64bit Amazon Linux 2
2) Python 3.7 running on 64bit Amazon Linux 2
3) Python 3.6 running on 64bit Amazon Linux
(default is 1): 1

Do you wish to continue with CodeCommit? (Y/n): n
Do you want to set up SSH for your instances?
(Y/n): y
```

Give required information



Key Pair for SSH

```
C:\Users\Ak\Desktop\django_project\CloudAWS\webapp>eb create cloudProjectenv
warning: LF will be replaced by CRLF in club/firebaseadmin.json.
The file will have its original line endings in your working directory
```

```
Uploading cloudProject/app-1002-210521_141643.zip to S3. This may take a while.
Upload Complete.
Environment details for: cloudProjectenv
  Application name: cloudProject
  Region: us-east-1
  Deployed Version: app-1002-210521_141643
  Environment ID: e-fvrgercymy
  Platform: arn:aws:elasticbeanstalk:us-east-1::platform/Python 3.8 running on 64bit Amazon Linux 2/3.2.2
  Tier: WebServer-Standard-1.0
  CNAME: UNKNOWN
  Updated: 2021-05-21 08:47:36.196000+00:00
Printing Status:
2021-05-21 08:47:34    INFO    createEnvironment is starting.
2021-05-21 08:47:36    INFO    Using elasticbeanstalk-us-east-1-174378555117 as Amazon S3 storage bucket for environment data.
2021-05-21 08:48:03    INFO    Created security group named: sg-0e0454e44aea278d7
2021-05-21 08:48:18    INFO    Created load balancer named: awseb-e-f-AWSEBLoa-1FN4Z96K4OG1G
2021-05-21 08:48:18    INFO    Created security group named: awseb-e-fvrgercymy-stack-AWSEBSecurityGroup-5SHHD1Q02RH6
2021-05-21 08:48:19    INFO    Created Auto Scaling launch configuration named: awseb-e-fvrgercymy-stack-AWSEBAutoScalingLaunchConfiguration-1CKW2CF
HELHQS
2021-05-21 08:49:22    INFO    Created Auto Scaling group named: awseb-e-fvrgercymy-stack-AWSEBAutoScalingGroup-1SN0AWZLNRNQR
2021-05-21 08:49:22    INFO    Waiting for EC2 instances to launch. This may take a few minutes.
2021-05-21 08:49:22    INFO    Created Auto Scaling group policy named: arn:aws:autoscaling:us-east-1:174378555117:scalingPolicy:a7a26049-583b-4dd2-
985c-c249e4830632:autoScalingGroupName/awseb-e-fvrgercymy-stack-AWSEBAutoScalingGroup-1SN0AWZLNRNQR:policyName/awseb-e-fvrgercymy-stack-AWSEBAutoSca
lingScaleDownPolicy-1IIDL37CXJDXJ
2021-05-21 08:49:22    INFO    Created Auto Scaling group policy named: arn:aws:autoscaling:us-east-1:174378555117:scalingPolicy:4356ddef-5bed-47f8-
85a1-42f9cc286e4f:autoScalingGroupName/awseb-e-fvrgercymy-stack-AWSEBAutoScalingGroup-1SN0AWZLNRNQR:policyName/awseb-e-fvrgercymy-stack-AWSEBAutoSca
lingScaleUpPolicy-178BESA7P2CDG
2021-05-21 08:49:22    INFO    Created CloudWatch alarm named: awseb-e-fvrgercymy-stack-AWSEBCloudwatchAlarmLow-1IAJRGSVWNVQD
2021-05-21 08:49:23    INFO    Created CloudWatch alarm named: awseb-e-fvrgercymy-stack-AWSEBCloudwatchAlarmHigh-191PF5W9HOO1T
2021-05-21 08:49:42    INFO    Instance deployment successfully generated a 'Procfile'.
2021-05-21 08:49:44    INFO    Instance deployment completed successfully.
2021-05-21 08:50:48    INFO    Successfully launched environment: cloudProjectenv
```

Uploading of code source and initialization of all services

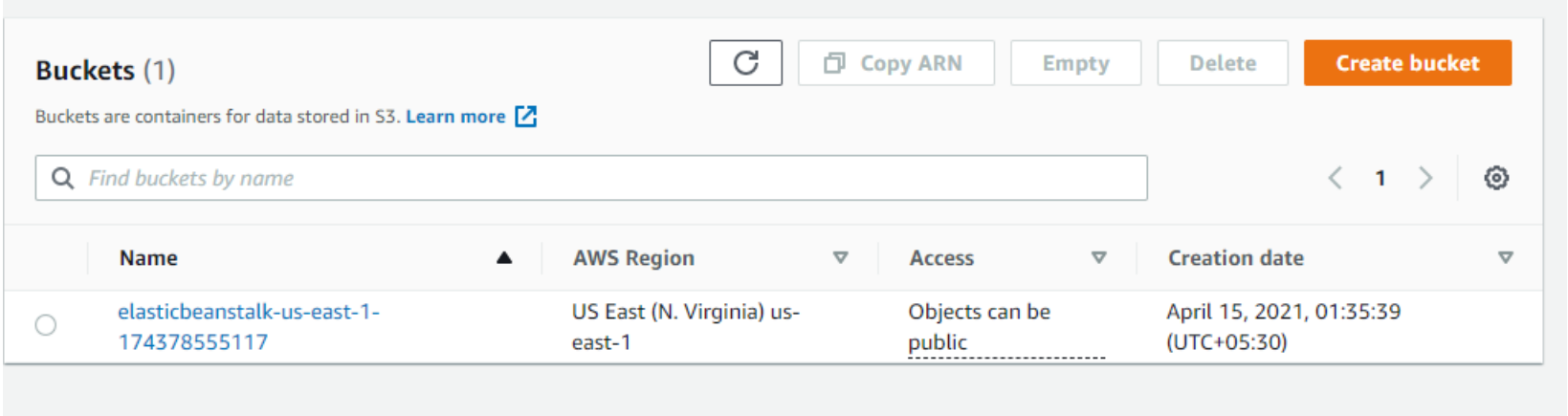
```
C:\Users\Ak\Desktop\django_project\CloudAWS\webapp>eb deploy
Creating application version archive "app-4cd4-210521_155207".
Uploading cloudProject/app-4cd4-210521_155207.zip to S3. This may take a while.
Upload Complete.
2021-05-21 10:27:00    INFO    Environment update is starting.
2021-05-21 10:27:43    INFO    Deploying new version to instance(s).
2021-05-21 10:27:50    INFO    Instance deployment successfully generated a 'Procfile'.
2021-05-21 10:27:58    INFO    Instance deployment completed successfully.
2021-05-21 10:28:22    INFO    New application version was deployed to running EC2 instances.
2021-05-21 10:28:22    INFO    Environment update completed successfully.


C:\Users\Ak\Desktop\django_project\CloudAWS\webapp>eb open
```

Type	Details
INFO	Environment update completed successfully.
INFO	New application version was deployed to running EC2 instances.
INFO	Instance deployment completed successfully.
INFO	Instance deployment successfully generated a 'Procfile'.
INFO	Deploying new version to instance(s).

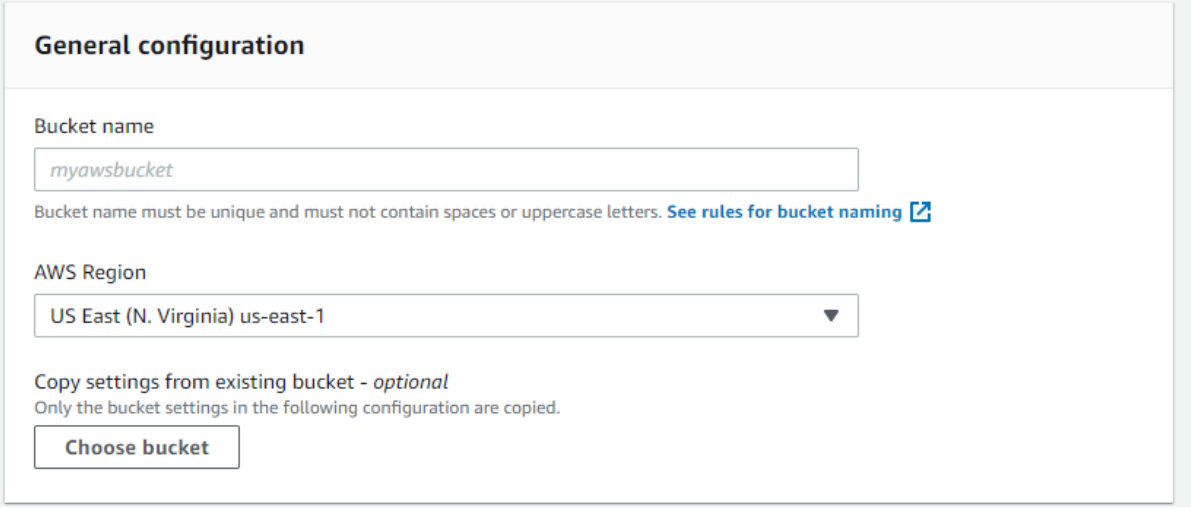
Upon successful deployment of source code

Steps using UI:

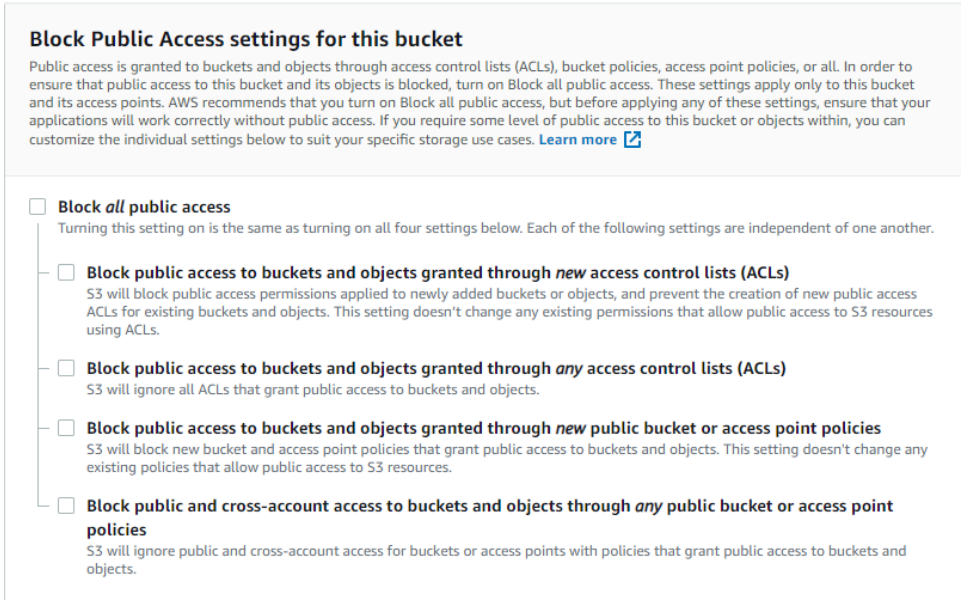
- Open AWS beanstalk
- Provide your App name
- Provide the environment
- In our case it is Python
- Choose a relevant python version
- Create a S3 bucket to upload the source code – cloudproject7008
- Copy its URL
- Click on create application



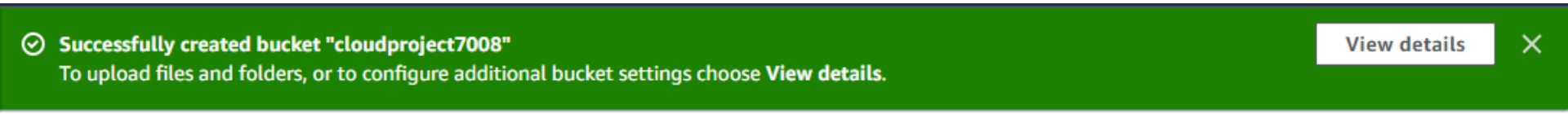
Click on create bucket

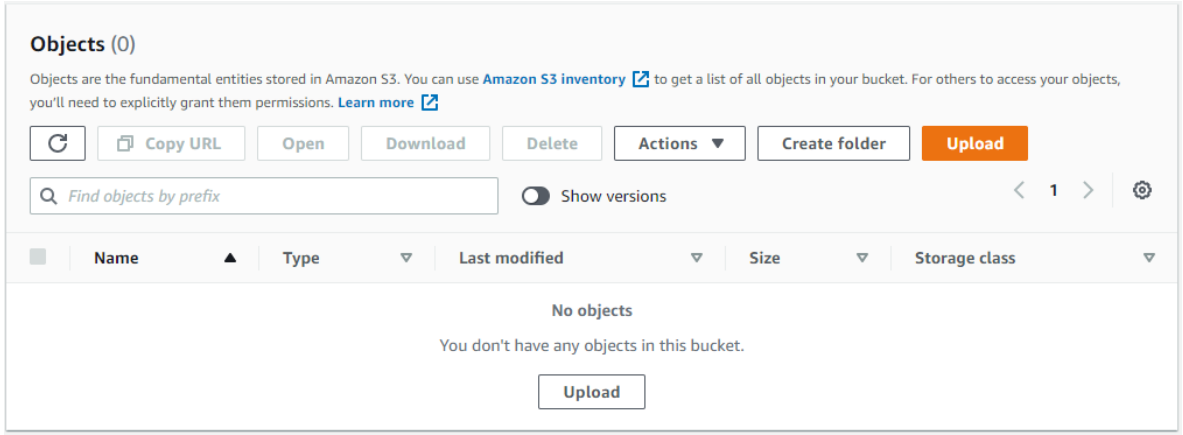


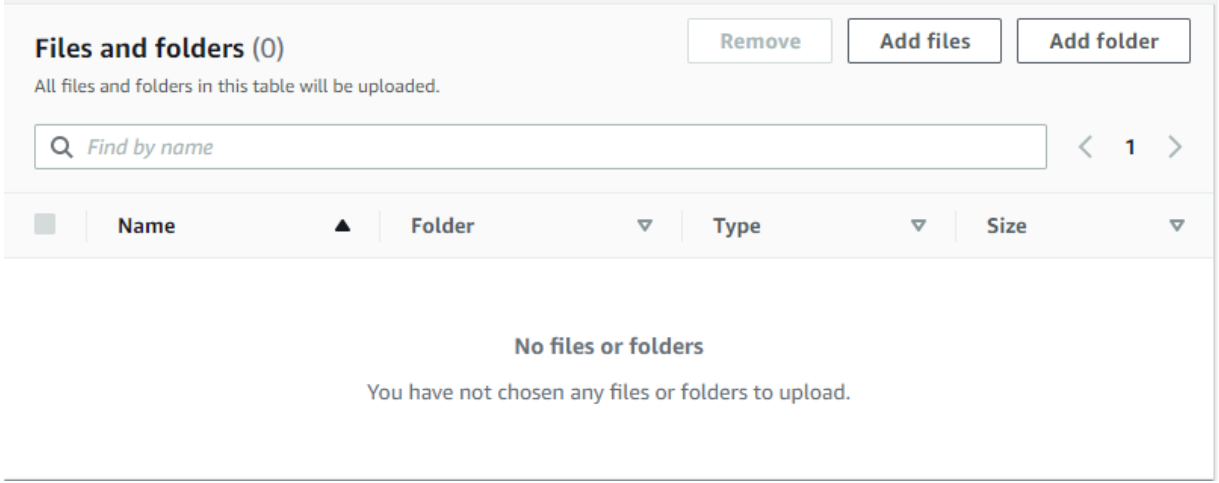
Provide Bucket name and keep the AWS region as default.



Uncheck the option “block all public access”







Click on upload and upload your source code

Uploading

2%

Cancel

Total remaining: 992 files: 8.3 MB(97.86%)
Estimated time remaining: 22 minutes
Transfer rate: 6.4 KB/s

Edit bucket policy

Bucket policy

The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. Learn more

Policy examples | Policy generator

Bucket ARN

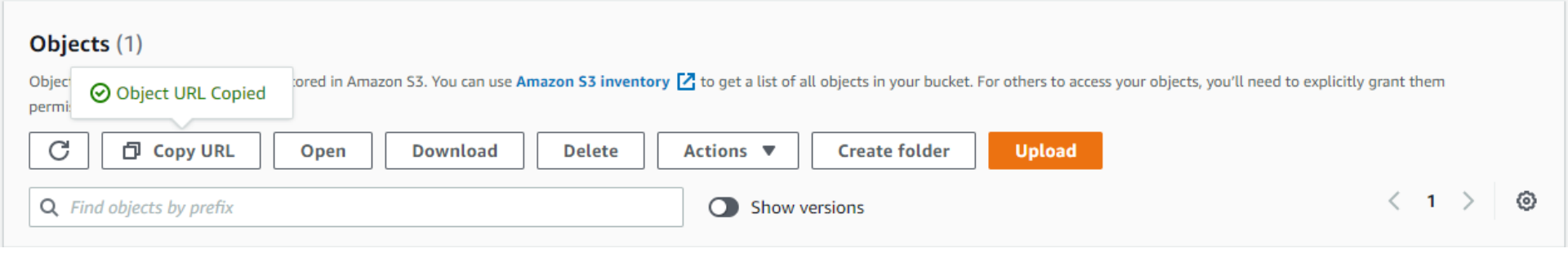
arn:aws:s3:::cloudproject7008

Policy

```
1
```

```
{
  "Version": "2008-10-17",
  "Statement": [
    {
      "Sid": "AllowPublicRead",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::cloudproject7008/*"
    }
  ]
}
```

Copy the bucket URL and start filling the beanstalk form



Application information

Application name

CloudProject

Up to 100 Unicode characters, not including forward slash (/).

Platform

Platform

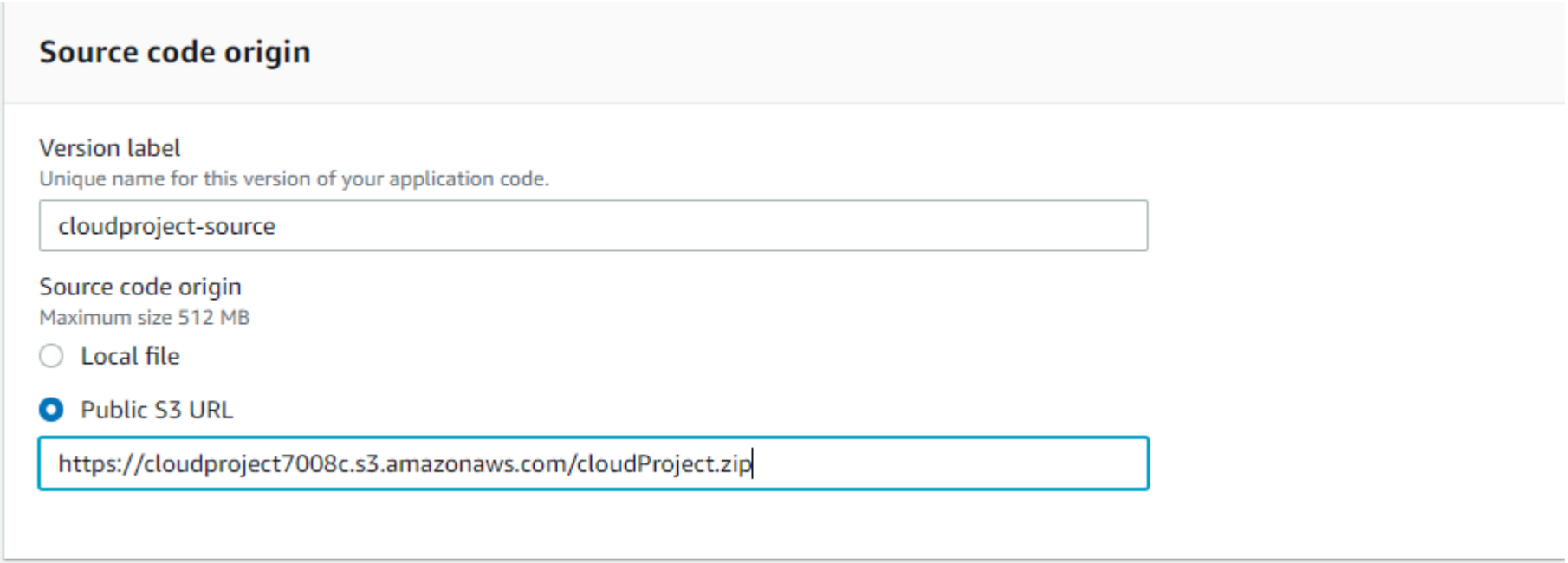
Python

Platform branch

Python 3.8 running on 64bit Amazon Linux 2

Platform version

3.2.2 (Recommended)



Click on create application

Creating Cloudproject-env-1
This will take a few minutes. ..

1:05pm Created CloudWatch alarm named:
awseb-e-vj6es3pm93-stack-AWSEBCloudwatchAlarmLow-1IDOXT00HFSWR

1:05pm Created CloudWatch alarm named:
awseb-e-vj6es3pm93-stack-AWSEBCloudwatchAlarmHigh-6DZM0O7702X4

1:05pm Created Auto Scaling group policy named:
arn:aws:autoscaling:us-east-1:174378555117:scalingPolicy:32e93802-65a2-4115-8904-f39e53c54b10:autoScalingGroupName/awseb-e-vj6es3pm93-stack-AWSEBAutoScalingGroup-1NJZPYZ0M23YV:policyName/awseb-e-vj6es3pm93-stack-AWSEBAutoScalingScaleDownPolicy-9D554HT7N0LG

1:05pm Created Auto Scaling group policy named:
arn:aws:autoscaling:us-east-1:174378555117:scalingPolicy:f03c9fa5-d920-476c-a53d-c6c213d0d885:autoScalingGroupName/awseb-e-vj6es3pm93-stack-AWSEBAutoScalingGroup-1NJZPYZ0M23YV:policyName/awseb-e-vj6es3pm93-stack-AWSEBAutoScalingScaleUpPolicy-1VQDK326JX3N4

1:05pm Created Load Balancer listener named:
arn:aws:elasticloadbalancing:us-east-1:174378555117:listener/app/awseb-AWSEB-G4XUXEIXFECZ/2761edcc0d205289/3e86db4fa39afc12

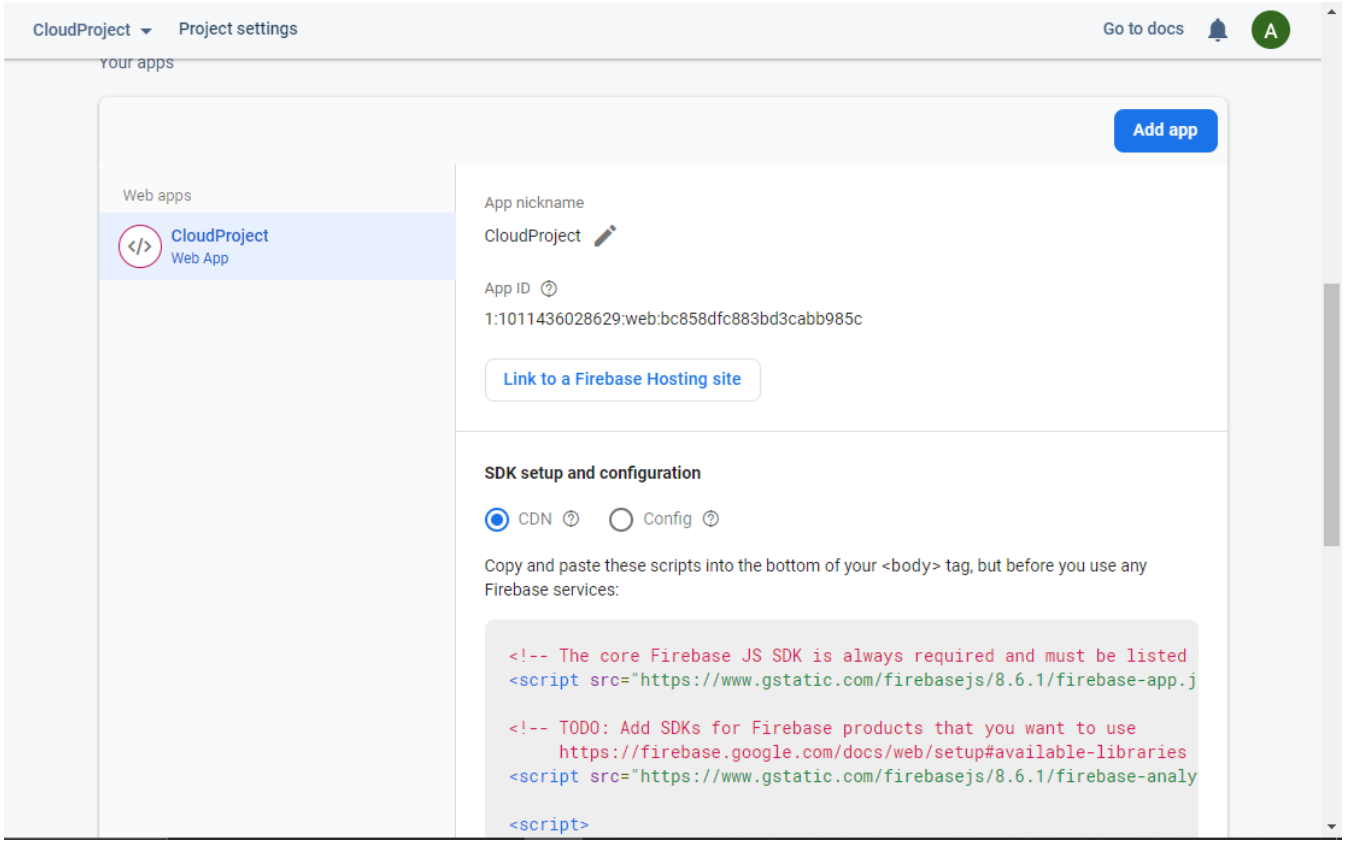
1:05pm Waiting for EC2 instances to launch. This may take a few minutes.

1:05pm Created Auto Scaling group named:
awseb-e-vj6es3pm93-stack-AWSEBAutoScalingGroup-1NJZPYZ0M23YV

1:05pm Created load balancer named:
arn:aws:elasticloadbalancing:us-east-1:174378555117:loadbalancer/app/awseb-AWSEB-G4XUXEIXFECZ/2761edcc0d205289

3) Storage Bucket

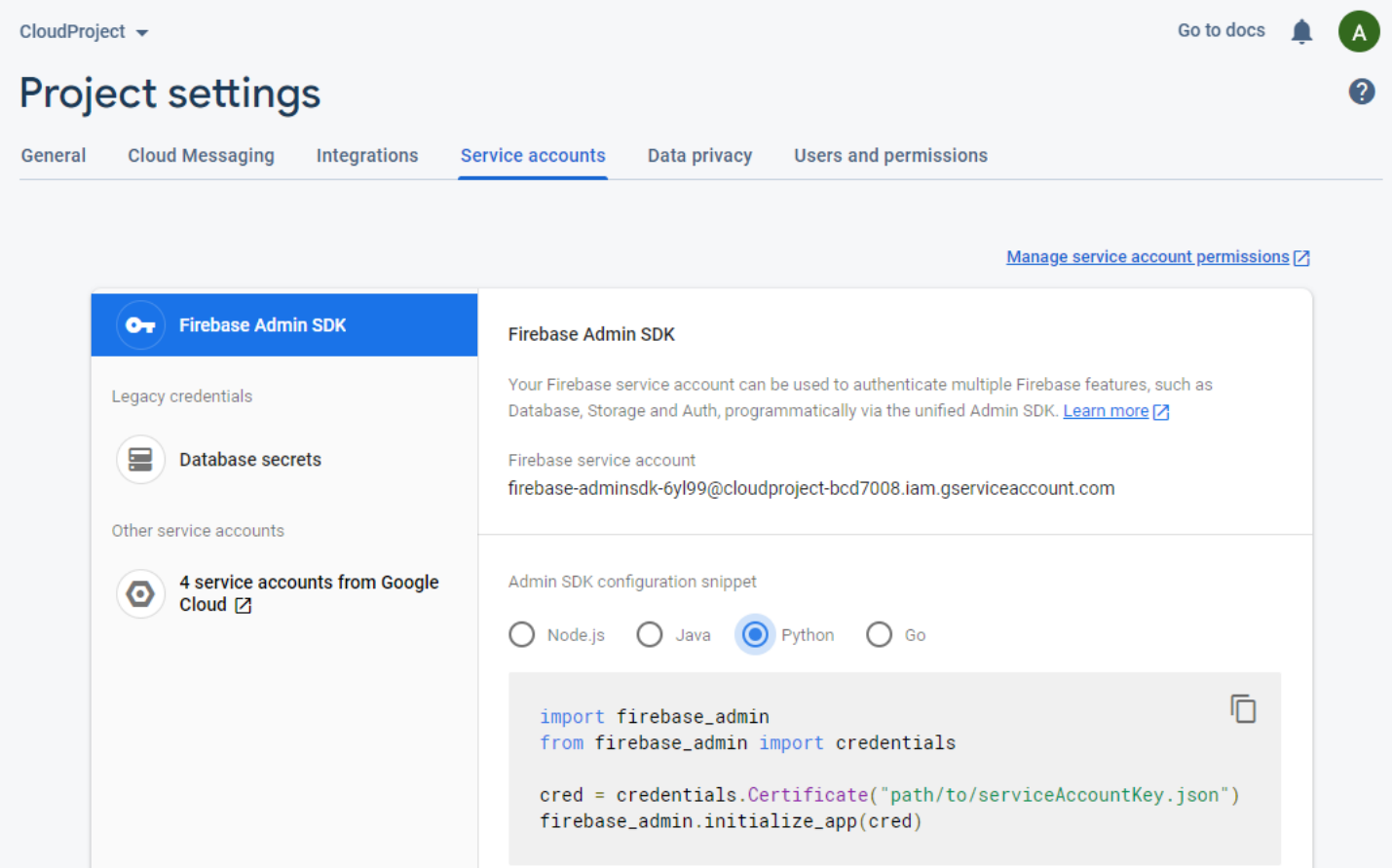
- Google Cloud Storage Bucket was used to store images generated by Admin and Clubs.
- Two folders were created. One for events images and one for club images.
- The information required to connect the webpage to the bucket can be found as follows:
    - Register a webapp with the Google Cloud Account
    - Go to Project Setting
    - Go to Service Account
    - Generate a new key
    - Download the JSON file which contains all the information required to connect
    - Connect to Google Cloud Storage



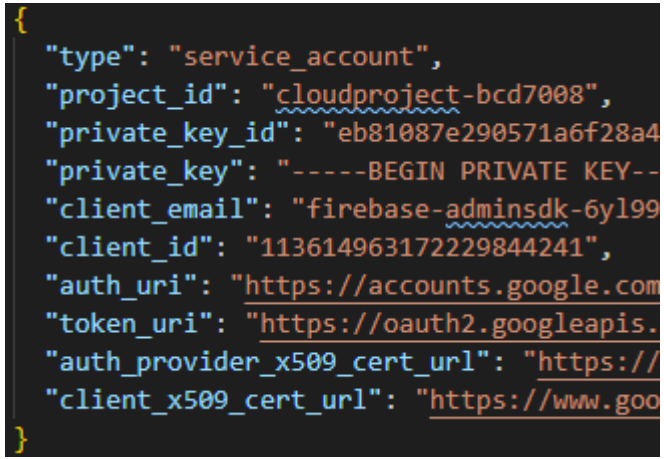
Register the webapp in Google Cloud

```
<!-- TODO: Add SDKs for Firebase products that you want to use
     https://firebase.google.com/docs/web/setup#available-libraries
<script src="https://www.gstatic.com/firebasejs/8.6.1/firebase-analy

<script>
  // Your web app's Firebase configuration
  // For Firebase JS SDK v7.20.0 and later, measurementId is optiona
  var firebaseConfig = {
    apiKey: "AIzaSyBEBdlgDtMjwb0SfE-s1AwJGVuOxVFBjaw",
    authDomain: "cloudproject-bcd7008.firebaseapp.com",
    databaseURL: "https://cloudproject-bcd7008-default-rtdb.firebase
    projectId: "cloudproject-bcd7008",
    storageBucket: "cloudproject-bcd7008.appspot.com",
    messagingSenderId: "1011436028629",
    appId: "1:1011436028629:web:bc858dfc883bd3cabb985c",
    measurementId: "G-EJYZXZDTQV"
  };
```

Configuration file for connecting using web



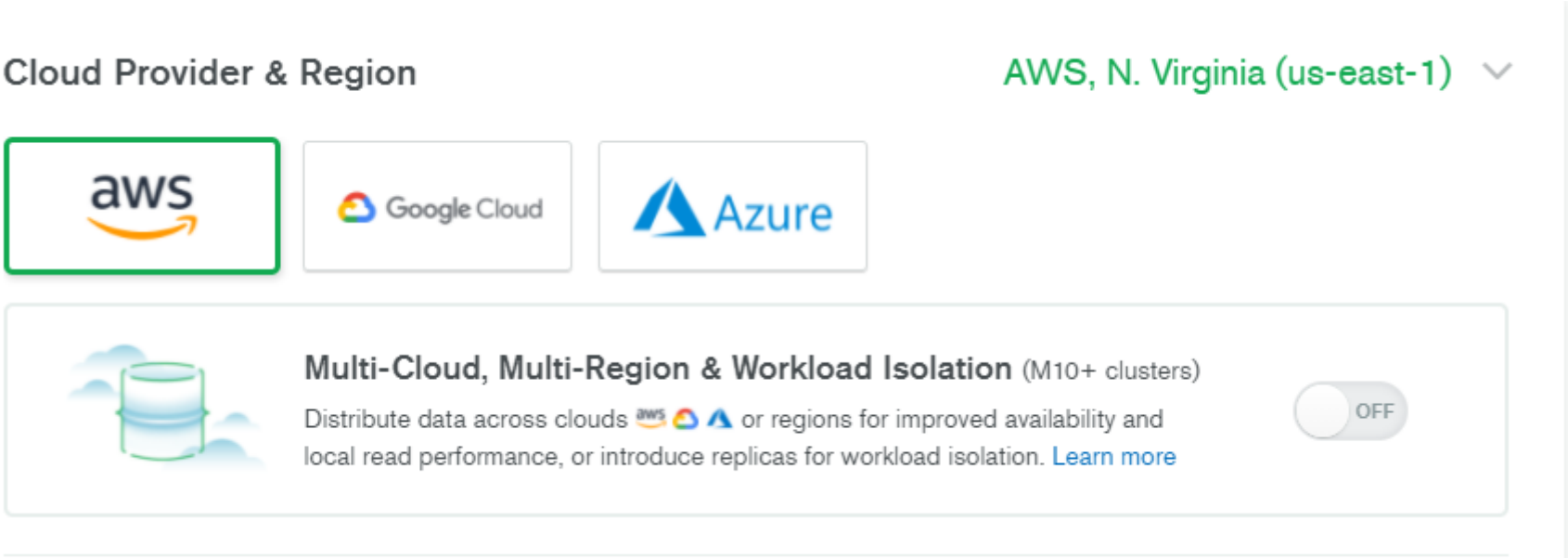
Project Setting / Service Account. Download the JSON file



JSON file content. Only half image to protect the account

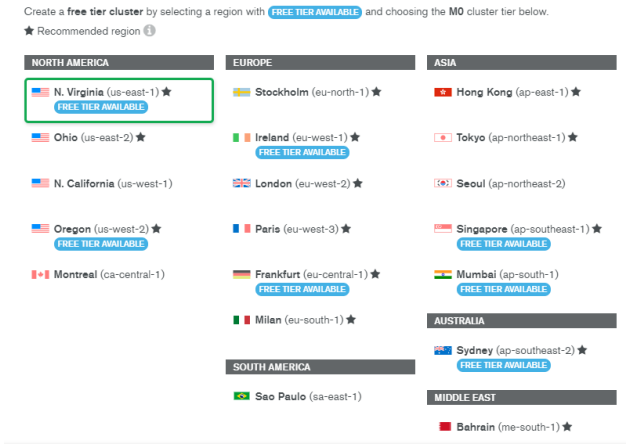
4) Database Connection

- Create account on cloud atlas and create a free cluster. First choose a cloud provider



4) Database Connection

- Choose the region on the node
- After selecting the region use the “Free Forever” sandbox
- Choose the cluster name.
- Click on create cluster



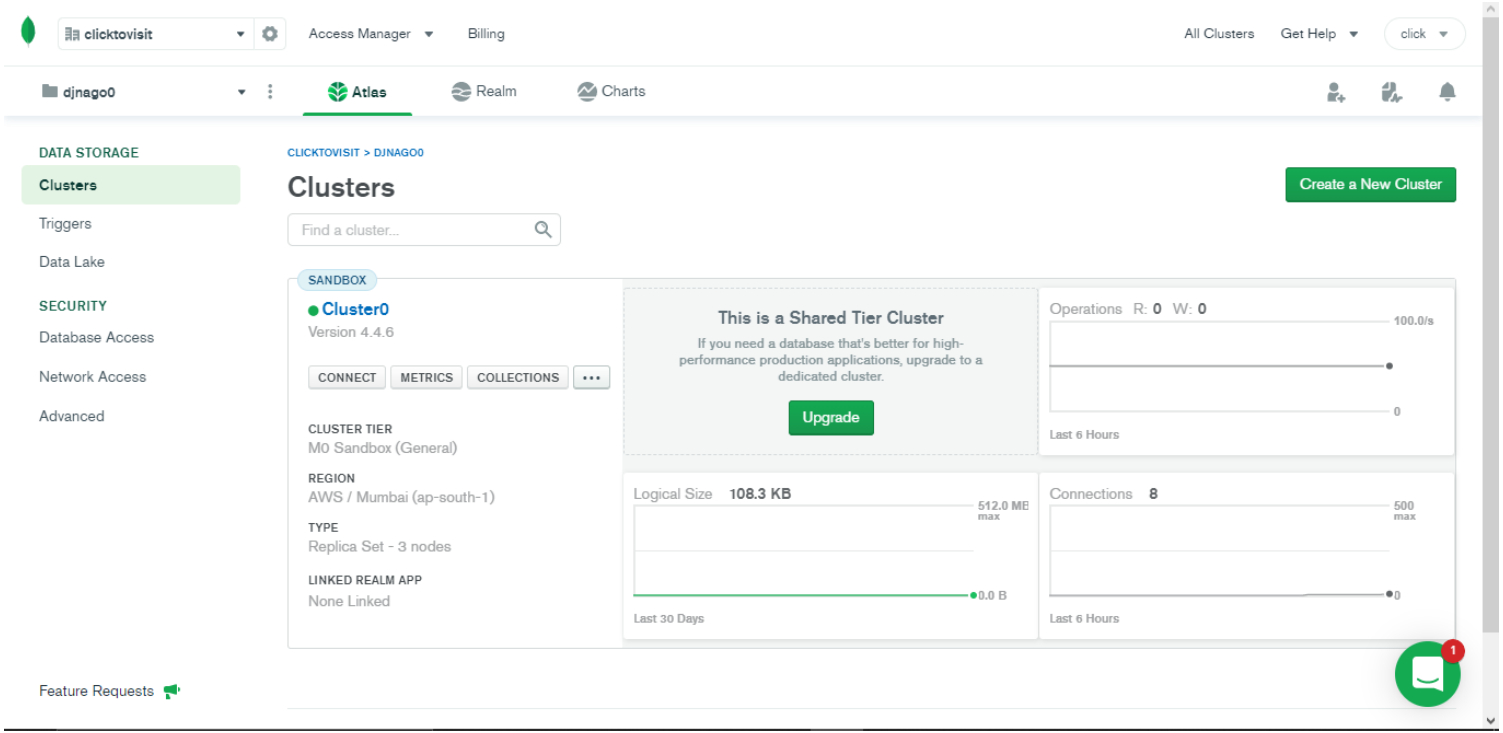
Shared Clusters for development environments and low-traffic applications

Tier	RAM	Storage	vCPU	Base Price
M0 Sandbox	There is a limit of one M0 cluster per project.			
M2	Shared	2 GB	Shared	$9 / MONTH
M5	Shared	5 GB	Shared	$25 / MONTH

As we have already created a cluster it shows M0 sandbox limit reached.

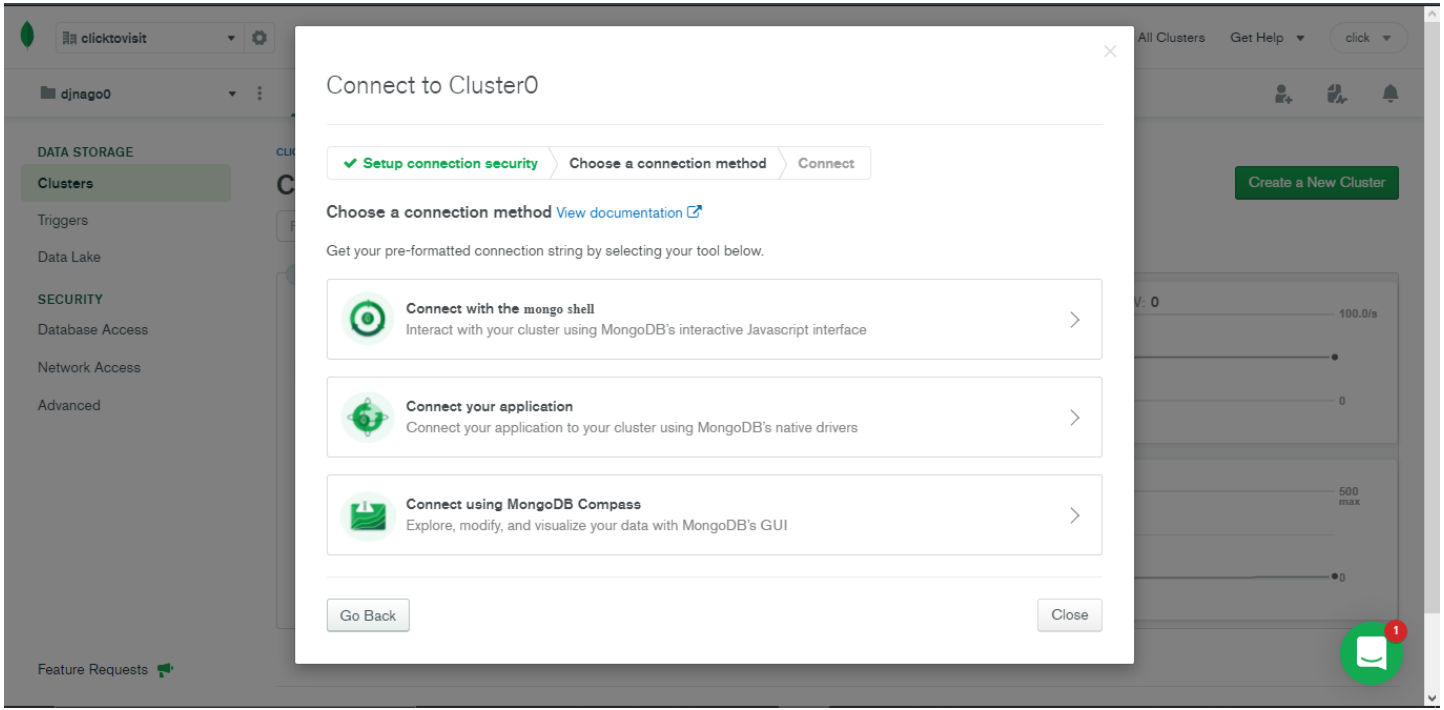
4) Database Connection

- After the cluster is created and connect button is available. Click on the connect button.



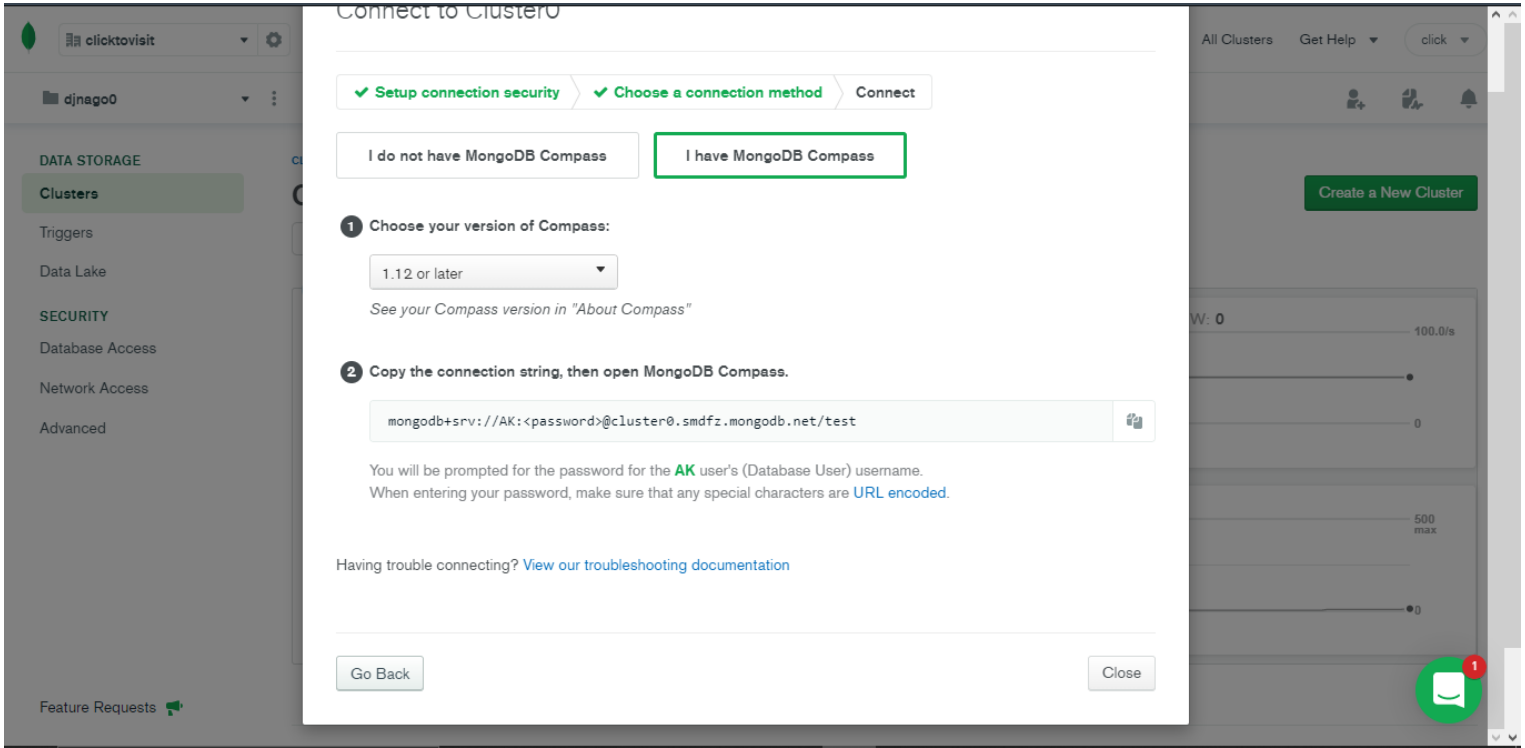
4) Connecting to MongoDB compass

- In choose a connection method choose MongoDB compass.



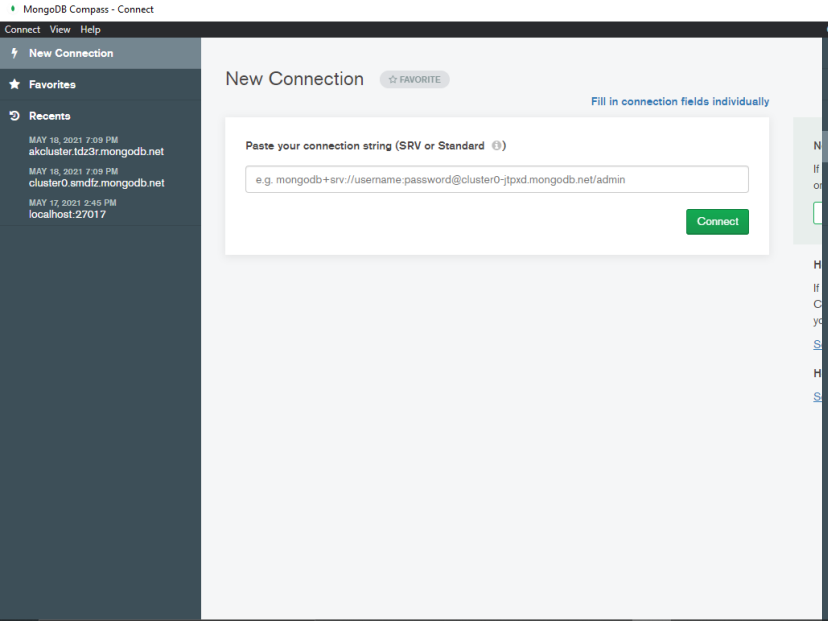
4) Connecting to MongoDB compass

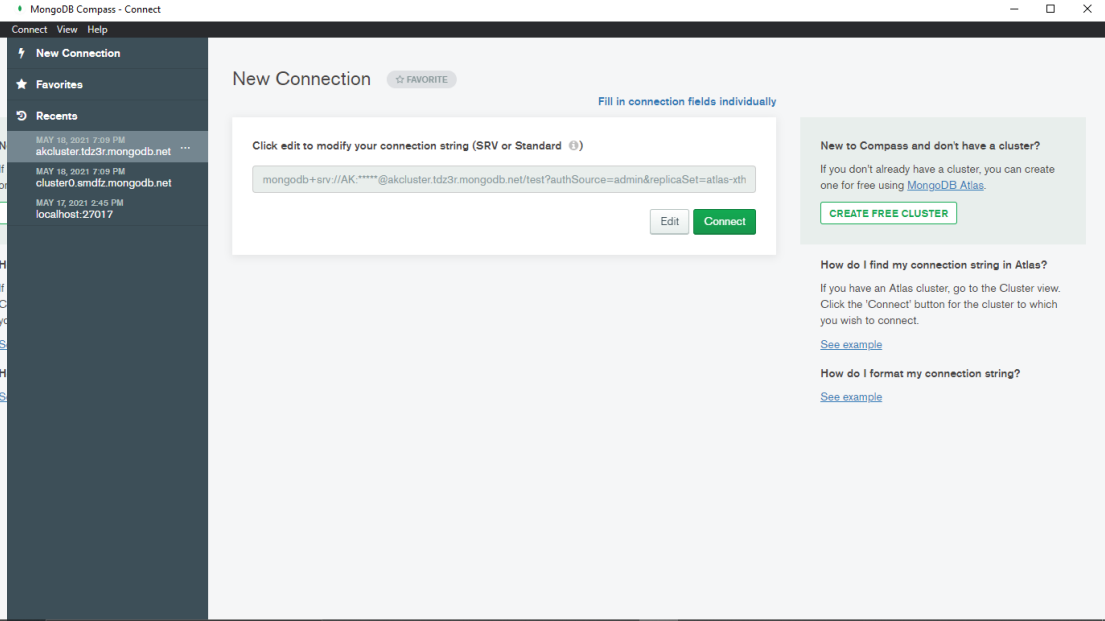
- Choose a proper version of the mongodb compass you have and copy the connection string.



4) Connecting to MongoDB compass

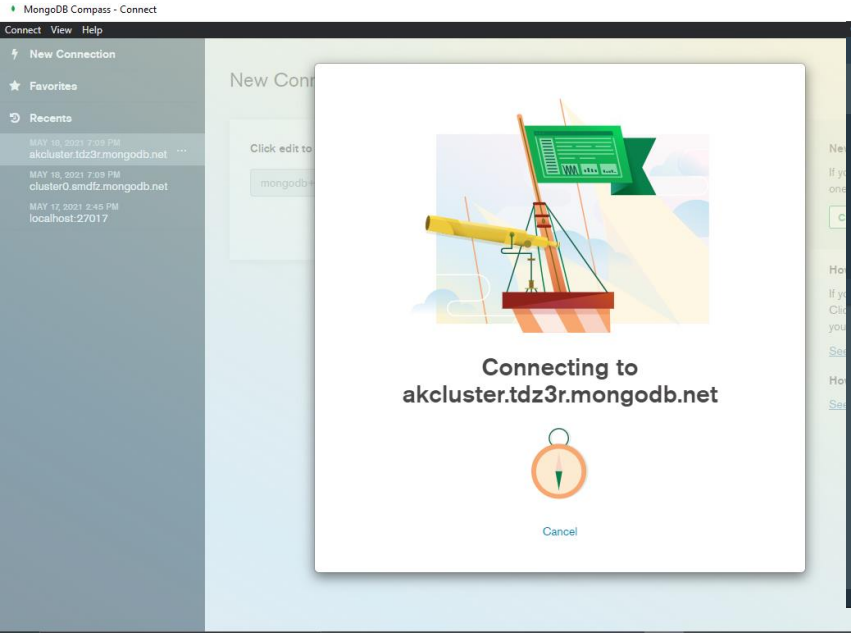
- The first window shows the MongoDB compass window. Paste the connection string in the box provided and replace the “password” with your password for the cluster in the string. Click connect.

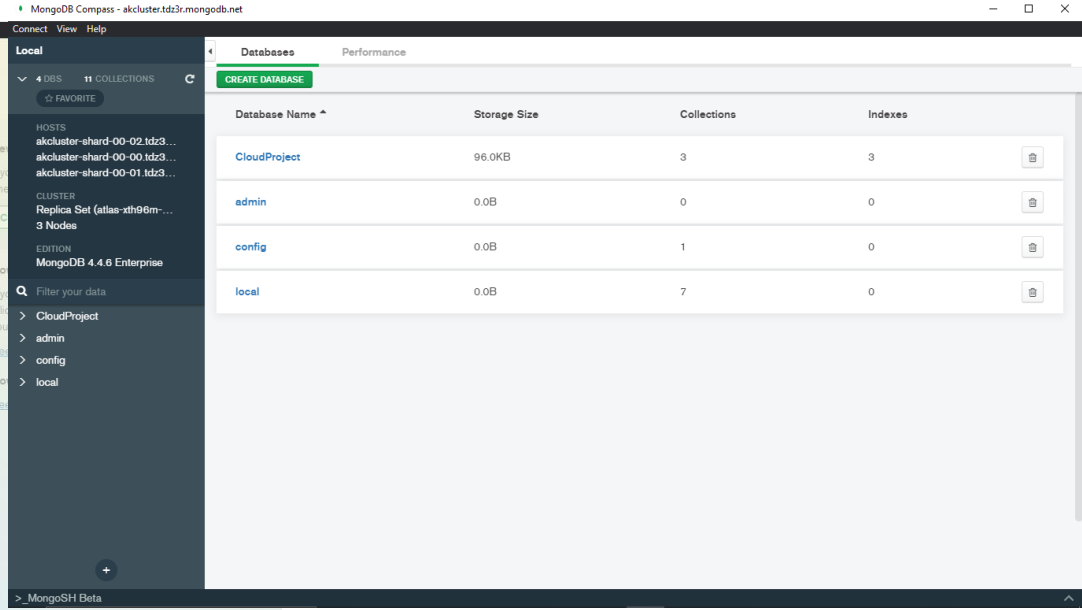




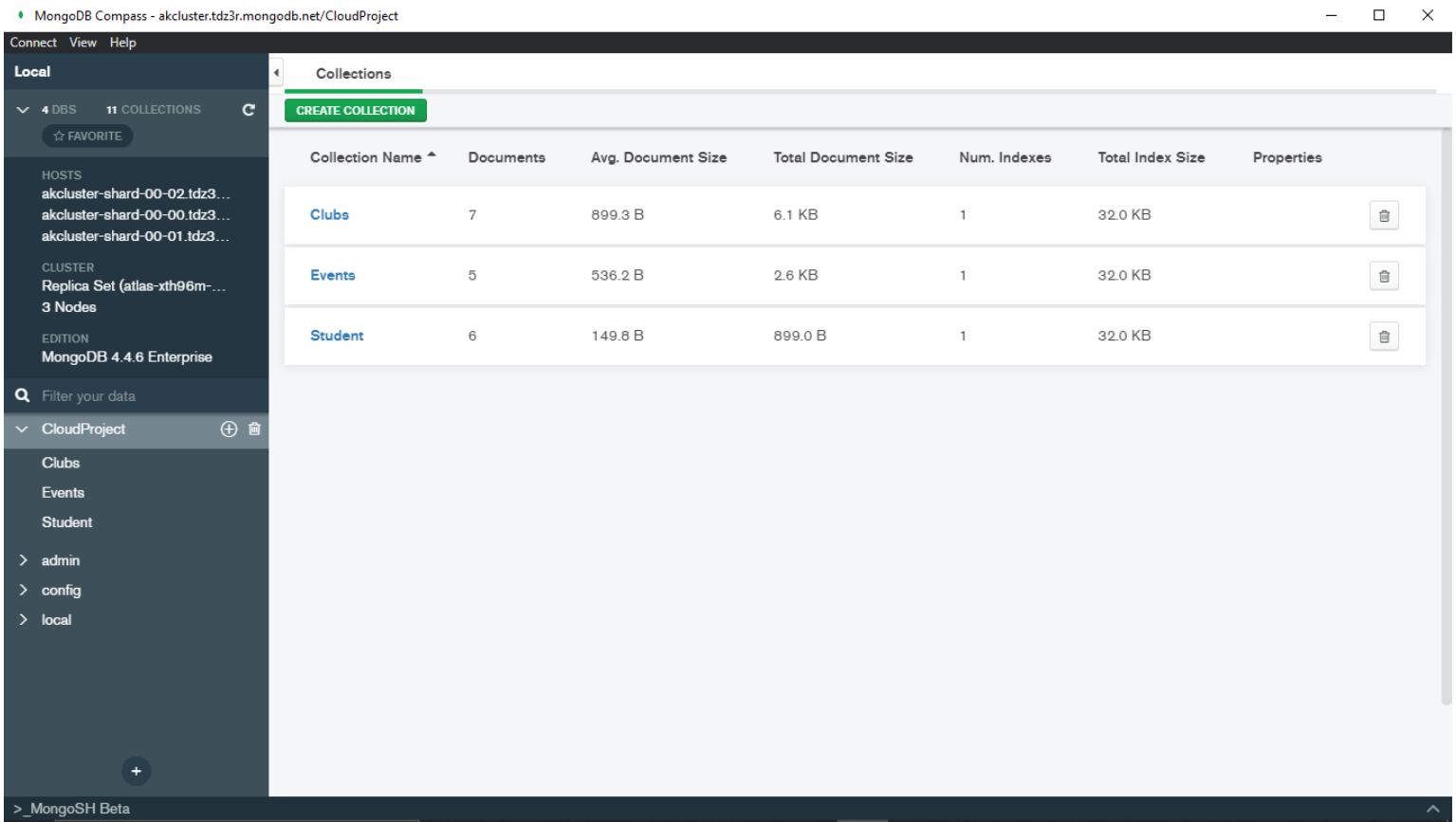
4) Connecting to MongoDB compass

- The first window should appear and it will take a few second to connect to the database. After that the second window will appear. Click on “Create Database” option and give database name and collection name. Do not select any check box. We have created a database named CloudProject.

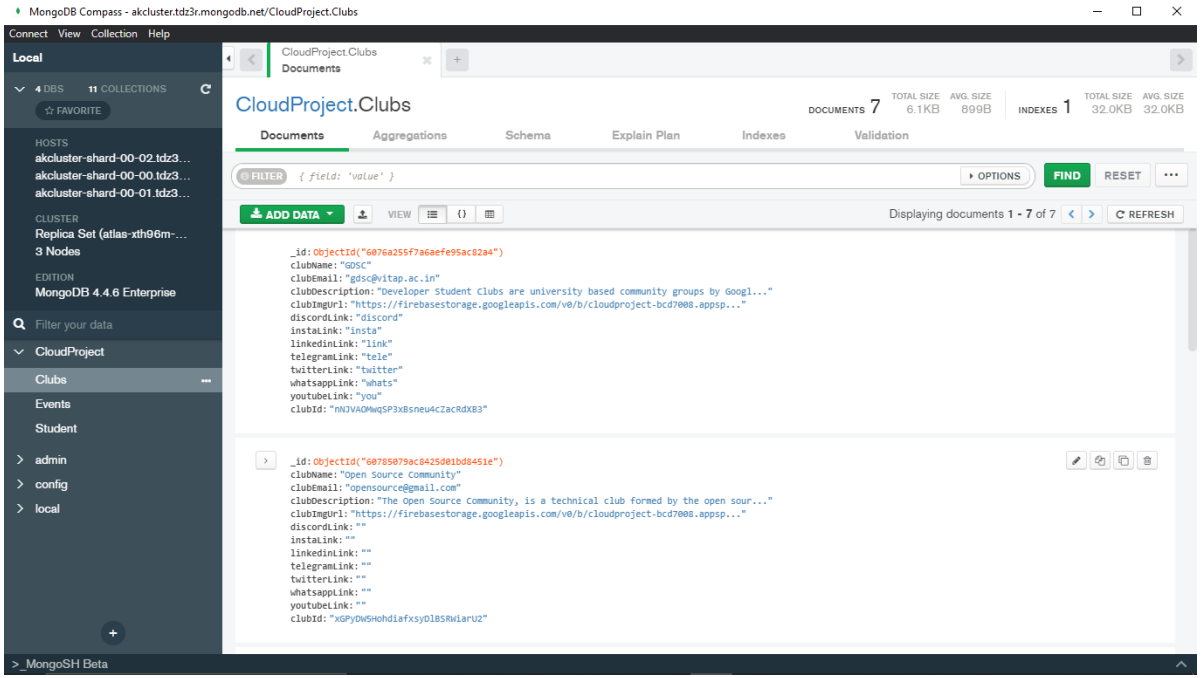


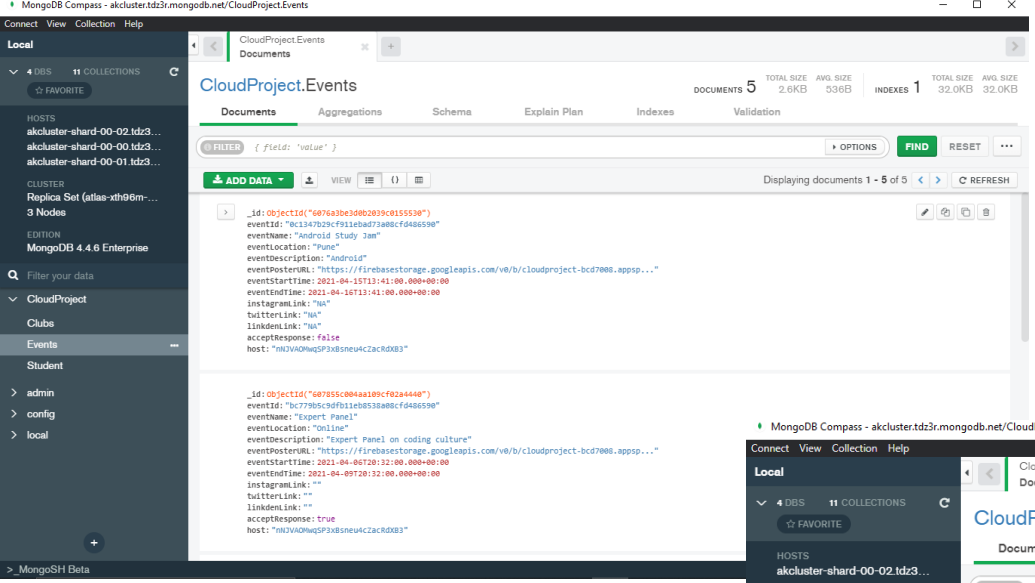


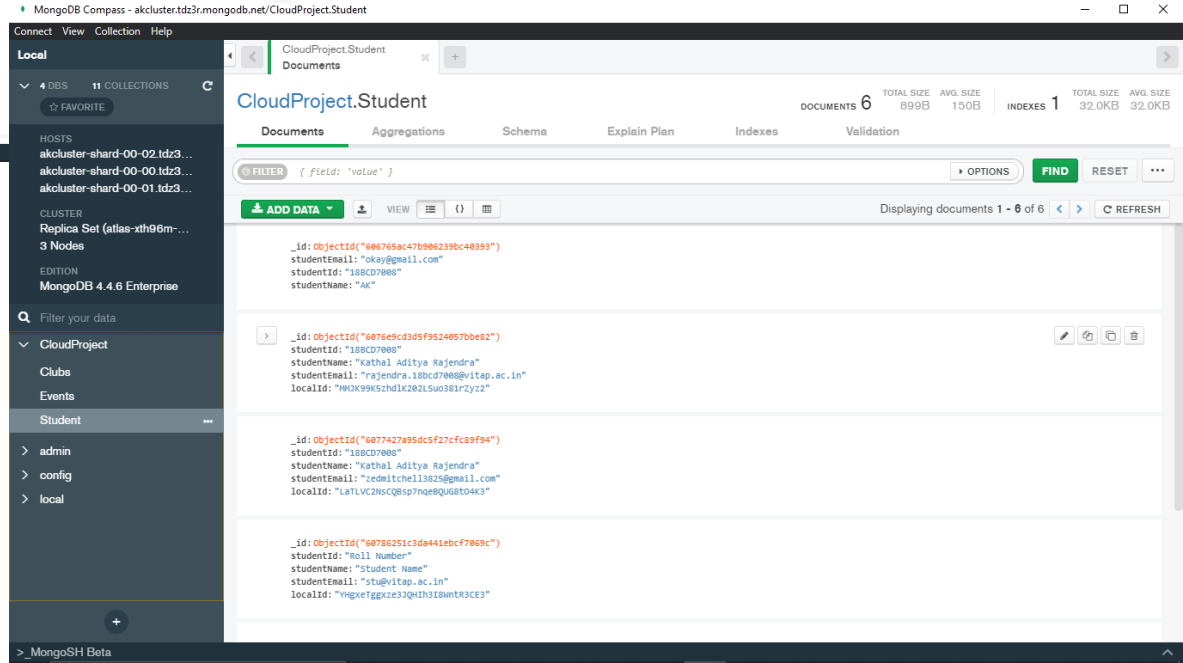
- To create different collections that will store data about different topics , click in create collection button. Give the information required and click on create.



- Data can be inserted into the collection using “Add Data” option or by writing query in mongo shell.

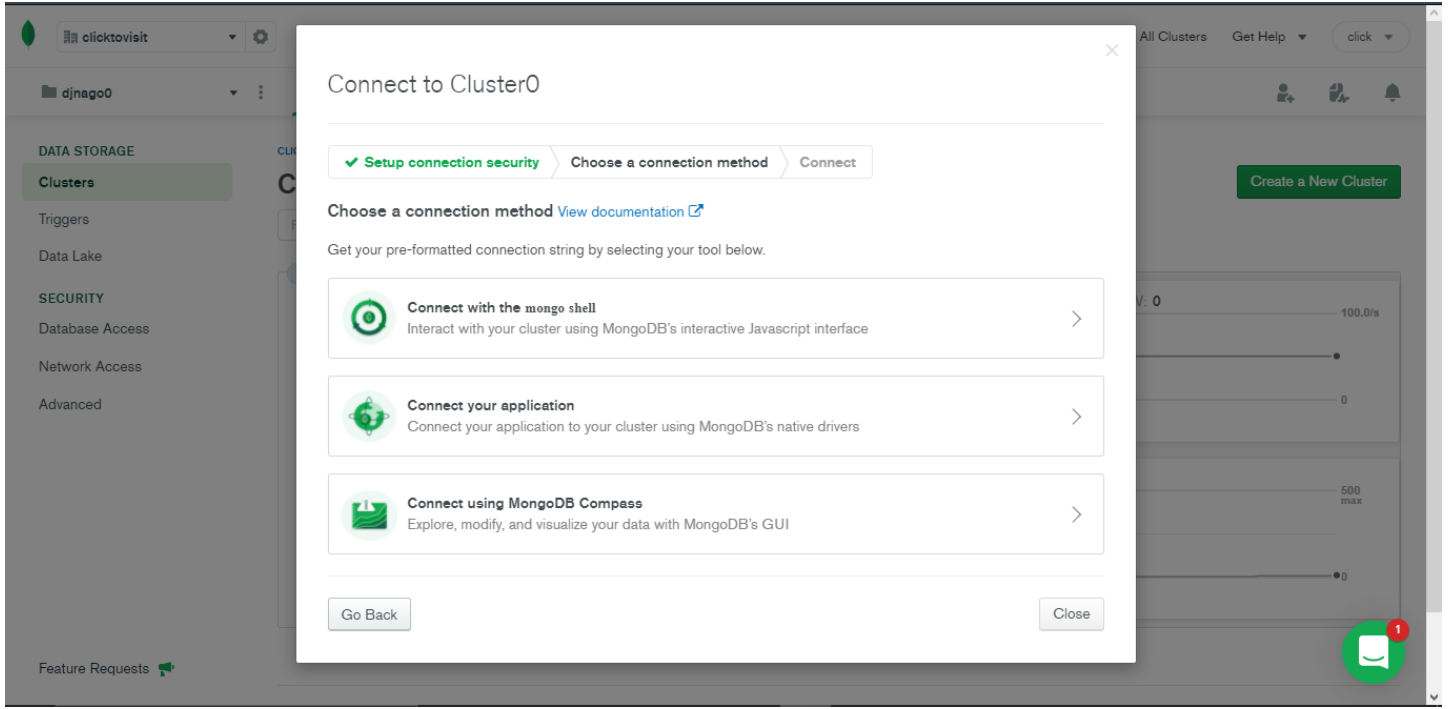



MongoDB Compass - akcluster.tdz3r.mongodb.net/CloudProject.Events
Connect View Collection Help
Local
4 DBS 11 COLLECTIONS
FAVORITE
HOSTS
akcluster-shard-00-02.tdz3...
akcluster-shard-00-00.tdz3...
akcluster-shard-00-01.tdz3...
CLUSTER
Replica Set (atlas-xth96m-...
3 Nodes
EDITION
MongoDB 4.4.6 Enterprise
Filter your data
CloudProject
Clubs
Events
Student
admin
config
local
CloudProject.Events
Documents
CloudProject.Events
DOCUMENTS 5 TOTAL SIZE 2.6KB AVG. SIZE 536B
INDEXES 1 TOTAL SIZE 32.0KB AVG. SIZE 32.0KB
Documents Aggregations Schema Explain Plan Indexes Validation
FILTER { field: 'value' } OPTIONS FIND RESET
ADD DATA VIEW
Displaying documents 1 - 5 of 5 REFRESH
_id: ObjectId("6076a3be3d0b2039c0155530")
eventId: "0c1347b29cf911ebad73a08cfd486590"
eventName: "Android Study Jam"
eventLocation: "Pune"
eventDescription: "Android"
eventPosterURL: "https://firebasestorage.googleapis.com/v0/b/cloudproject-bcd7008.appsp..."
eventStartTime: 2021-04-15T13:41:00.000+00:00
eventEndTime: 2021-04-16T13:41:00.000+00:00
instagramLink: "NA"
twitterLink: "NA"
linkdenLink: "NA"
acceptResponse: false
host: "nNJVAOMwqSP3xBsneu4CZacRdXB3"
_id: ObjectId("607855c004aa109cf02a4440")
eventId: "bc779b5c9df911ebb538a08cfd486590"
eventName: "Expert Panel"
eventLocation: "Online"
eventDescription: "Expert Panel on coding culture"
eventPosterURL: "https://firebasestorage.googleapis.com/v0/b/cloudproject-bcd7008.appsp..."
eventStartTime: 2021-04-06T20:32:00.000+00:00
eventEndTime: 2021-04-05T20:32:00.000+00:00
instagramLink: ""
twitterLink: ""
linkdenLink: ""
acceptResponse: true
host: "nNJVAOMwqSP3xBsneu4CZacRdXB3"
>_MongoSH Beta


MongoDB Compass - akcluster.tdz3r.mongodb.net/CloudProject.Student
Connect View Collection Help
Local
4 DBS 11 COLLECTIONS
FAVORITE
HOSTS
akcluster-shard-00-02.tdz3...
akcluster-shard-00-00.tdz3...
akcluster-shard-00-01.tdz3...
CLUSTER
Replica Set (atlas-xth96m-...
3 Nodes
EDITION
MongoDB 4.4.6 Enterprise
Filter your data
CloudProject
Clubs
Events
Student
admin
config
local
CloudProject.Student
Documents
CloudProject.Student
DOCUMENTS 6 TOTAL SIZE 899B AVG. SIZE 150B
INDEXES 1 TOTAL SIZE 32.0KB AVG. SIZE 32.0KB
Documents Aggregations Schema Explain Plan Indexes Validation
FILTER { field: 'value' } OPTIONS FIND RESET
ADD DATA VIEW
Displaying documents 1 - 6 of 6 REFRESH
_id: ObjectId("606765ac47b906239bc40393")
studentEmail: "okay@gmail.com"
studentId: "18BCD7008"
studentName: "AK"
_id: ObjectId("6076e9cd3d5f9524057bbe82")
studentId: "18BCD7008"
studentName: "Kathal Aditya Rajendra"
studentEmail: "rajendra.18bcd7008@vitap.ac.in"
localId: "MMJK99K5zhdlK202LSuo381rZyz2"
_id: ObjectId("6077427a95dc5f27cfc89f94")
studentId: "18BCD7008"
studentName: "Kathal Aditya Rajendra"
studentEmail: "zedmitchell3825@gmail.com"
localId: "LaTLVC2NsCQ8sp7nqe8QUG8tO4k3"
_id: ObjectId("60786251c3da441ebcf7069c")
studentId: "Roll Number"
studentName: "Student Name"
studentEmail: "stu@vitap.ac.in"
localId: "YHgxeTggxze3JQHIh3I8WntR3CE3"
>_MongoSH Beta

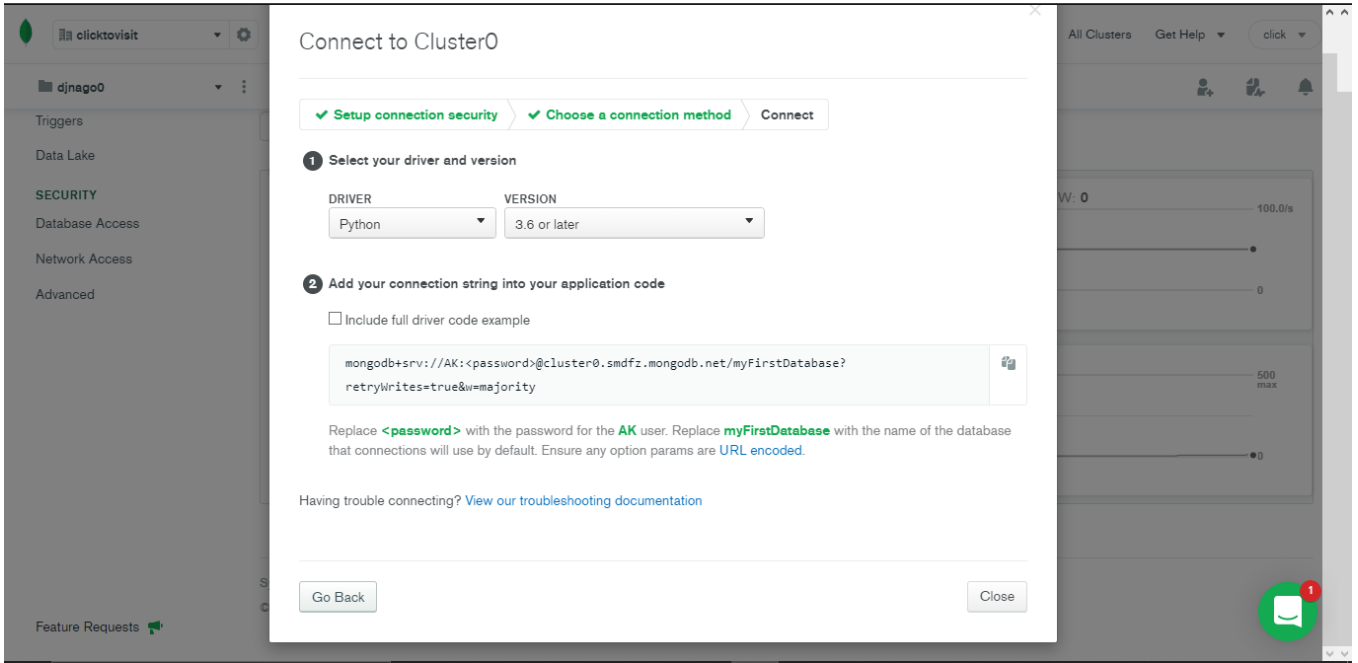
4) Connecting to cluster using python

- Choose the “connect your application” option.



4) Connecting to cluster using python

- Choose a proper python version and copy the connection string.



4) Connecting to cluster using python

- Run “pip install pymongo”
- Use the connection string from before.
- Replace <password> with your cluster password.
- Now we can run any query on the database using the db variable reference.

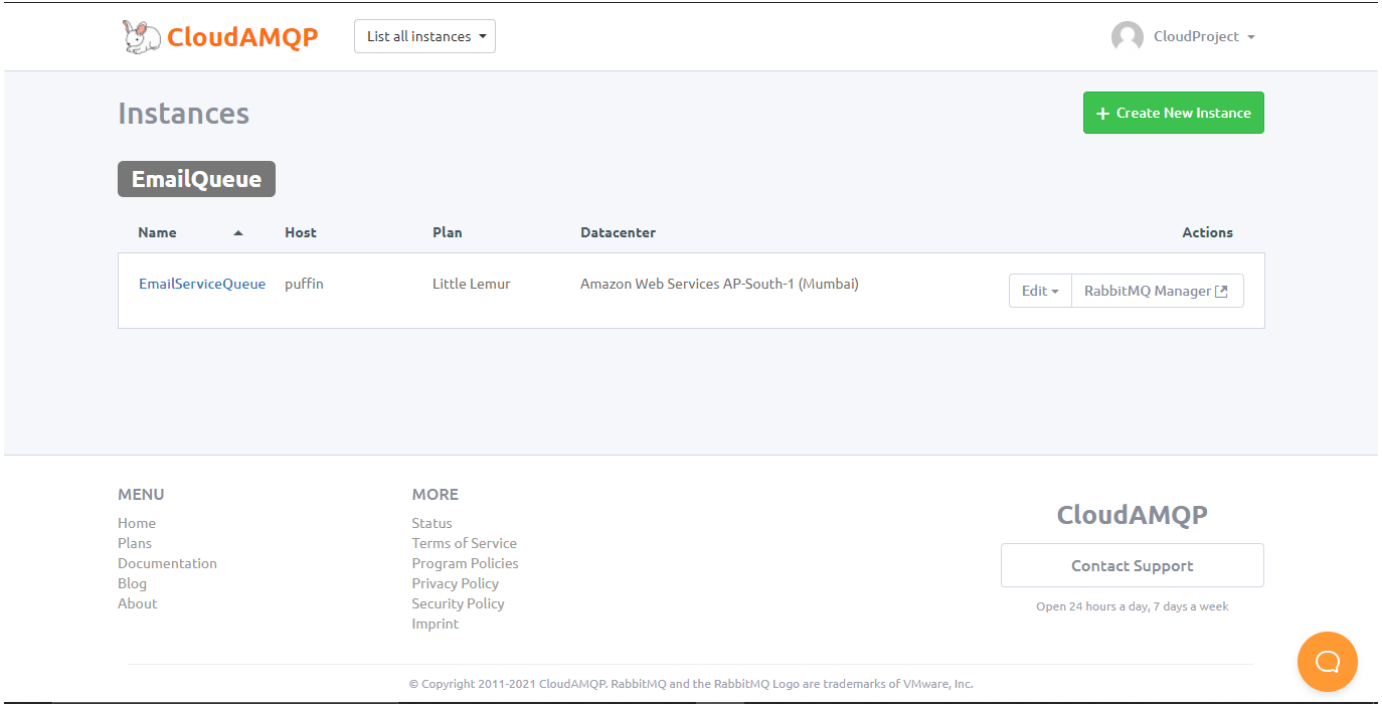
```
client = MongoClient("mongodb+srv://AK:c2V
db = client.get_database("CloudProject")
```

5) Messaging Queue

- Messing Queue is implemented to reduce load off the webserver while sending mails.
- If messaging queue would not have been used the webserver will have to send mail itself to the user and thus it will not be able to respond to a longer time. This would have reduced the efficiency of the webserver.
- Also in case of error webserver would have to itself resolve the error and thus eating much more of its time.
- We have implemented the messaging queue using the CloudAMQP which provides RabbitMQ has a service.
- Thus now a different program listens to the changes in the queue and responds accordingly saving the webserver’s time.
- The first step is to create a free account in the CloudAMQP website.

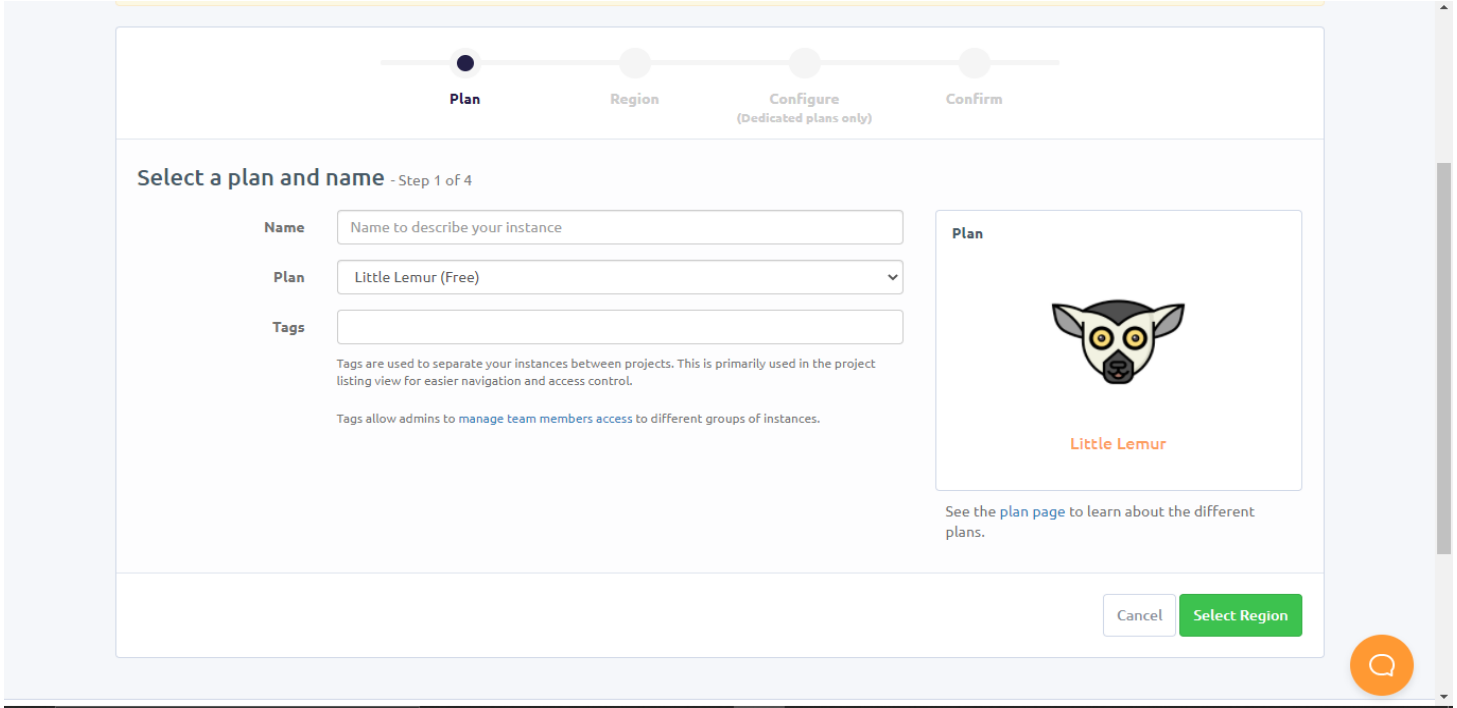
5) Messaging Queue

- Click on create new Instance.



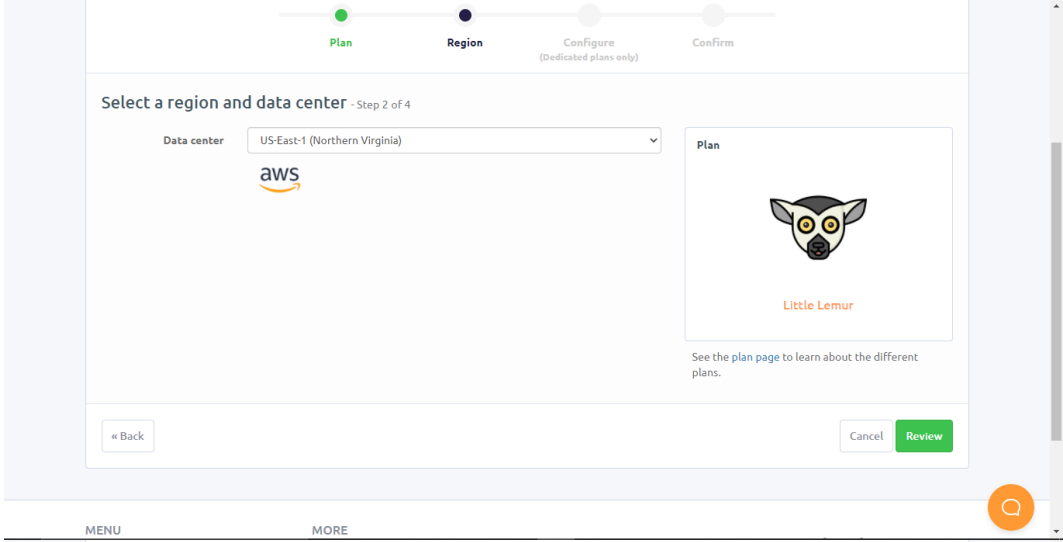
5) Messaging Queue

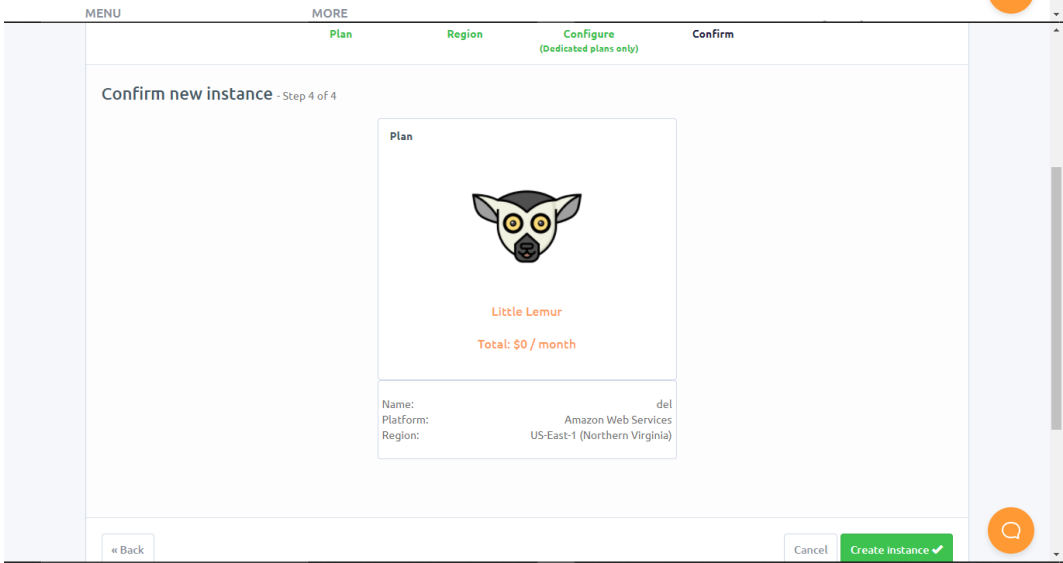
- Fill the details required. Remember this is a instance not the queue. This instance will contain the queue. So name accordingly.



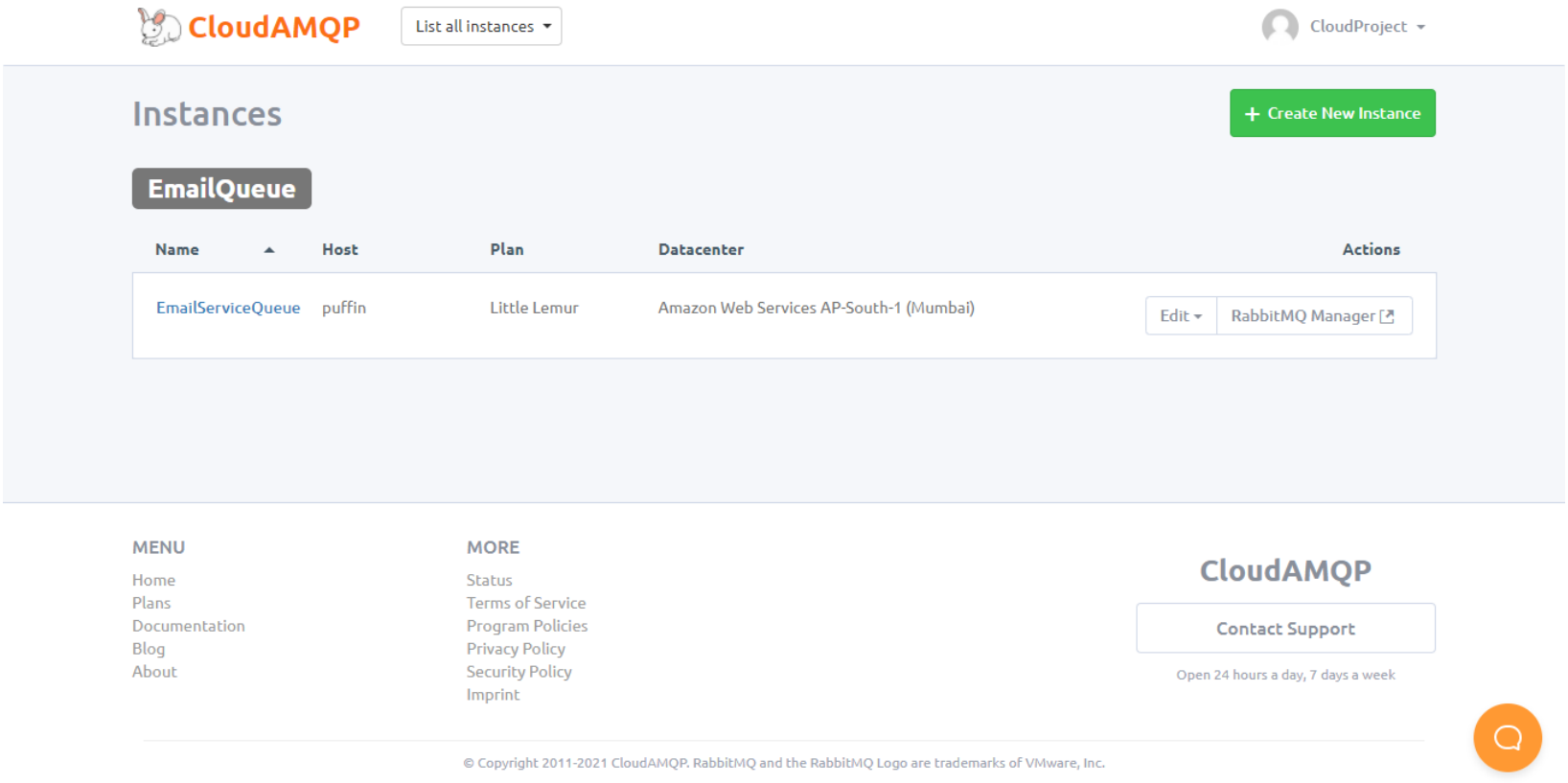
5) Messaging Queue

- Select the data center and its respective cloud provider.
- Since we are in India , South –East is a good location with low latency. If Mumbai is available it is even more good.
- Click on “create instace” after review.

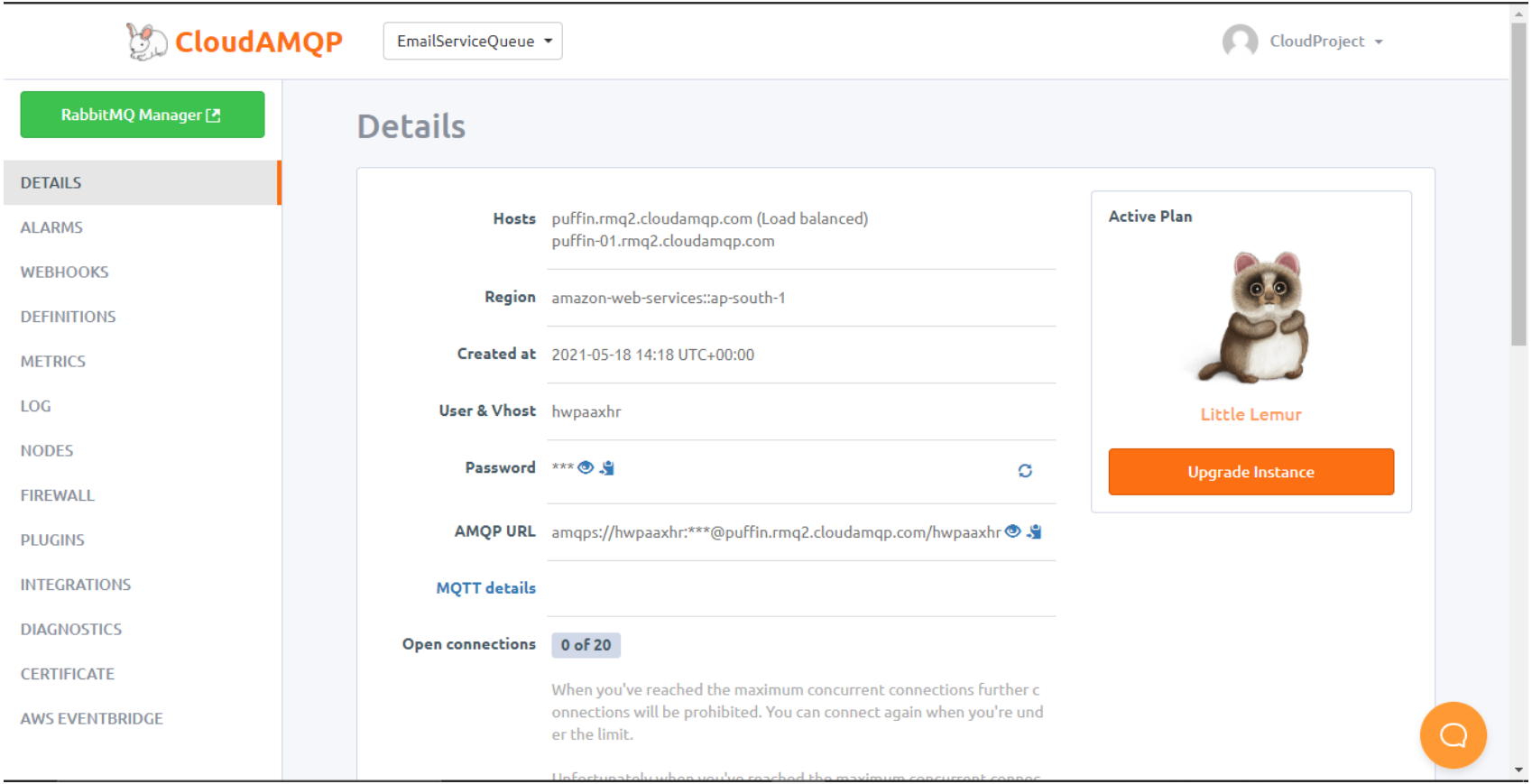




- After the instance is created its name will be visible here. Click on the URL that is the instance name.



- This window will be visible after clicking that URL. In this window you can find the AMQP URL . Copy this URL . We will require this to connect to the service from our code.



- After publishing the queue from code , they will be visible under RabbitMQ Manager. For our project we have created two queue , one for dealing with club related emails and other with event related emails.

6) Sending Mail

- We are using SendGrid a email service provided by Twilio.
- Create a SendGrid account
- Then go to API keys
- Generate an API key
- Use this API key to connect to SendGrid using sendgrid package

Aditya Kathal

- Dashboard
- Email API
- Marketing NEW
- Design Library
- Stats
- Activity
- Suppressions
- Settings

REPUTATION 100%

VIEW ACCOUNT USAGE

Ready to unlock the full potential of Twilio SendGrid?

Finish your account setup in order to select a paid Email API or Marketing Campaigns plan.

Let's do it!

Hello Aditya! Here's your recent email activity.

Wk Mo

REQUESTS	DELIVERED	OPENED	CLICKED	BOUNCES	SPAM REPORTS
11	36.36% 4	90.91% 10	0.00% 0	0.00% 0	0.00% 0

Activity

Suppressions

Settings

Account Details
Alert Settings
API Keys
Inbound Parse
IP Access Management
IP Addresses
Mail Settings
Sender Authentication
Subuser Management
Teammates
Tracking
Two-Factor Authentication

REPUTATION 100%

VIEW ACCOUNT USAGE

Finish your account setup in order to select a paid plan and unlock the full potential of Twilio SendGrid. Let's do it! Remind Me Later

Aditya Kathal

API Keys

Create API Key

NAME	API KEY	ACTION
cloudproject API Key ID: FPfsL7XFS1aQMNgBK-Nbiw	••••••••••••••••••••	

- Dashboard
- Email API
- Marketing NEW
- Design Library
- Stats
- Activity
- Suppressions
- Settings
  - Account Details
  - Alert Settings

References

- https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/create-deploy-python-django.html
- https://devcenter.heroku.com/articles/deploying-python
- https://help.ubuntu.com/community/CronHowto
- https://semaphoreci.com/blog/cicd-pipeline