CS 816 - Software Production Engineering

Major Project - College Event Management

Submitted By:
Shreyank Buddhadev (MT2021130)
Meet Patel (MT2021076)

Index:

Abstract:

Any large organization has many events scheduled, for a student / management It becomes really hard to keep track of all ongoing events. Sometimes it happens that a student is interested in some specific domain events but he/she might never know regarding that event.

Users of the app:

Users:

1. Students
2. Event Organizer
3. Admin / Management committee

Links:

Github: codebase
Dockerhub: server, client
API Docs: Docs
Functional Req: Sheet

Introduction:

Users and Features of the app:

Users:

1. Students
2. Event Organizer
3. Admin / Management committee

Features:

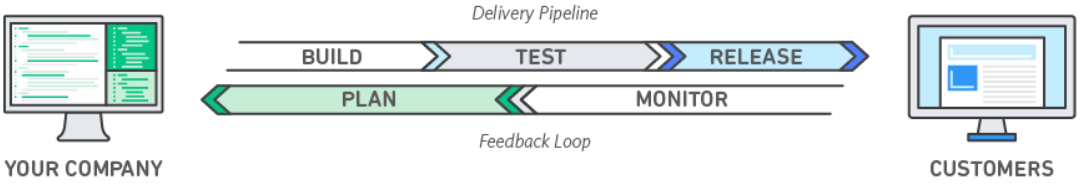
- **For Students:**
    - A student can log in and see all the ongoing / upcoming events, In which he/she can participate.
    - Students can filter events using tags such as *research talk, debate, seminar, technical event, cultural event etc.*
    - Students can directly register from the website itself for the event.
    - *Competition result*
    - Can download certificates

- **For Event Organizers:**
    - An Event organizer can raise a request to **Admin,** to organize event[s]. {*A request can contain venue, cost, expected participation, students allowed to participate*}. After successful registration event will be shown on the dashboard of students which are allowed to participate.

- **Admin / Management committee**:
    - Admin can view all the ongoing event[s] details on its dashboard.
    - Admin can approve / disapprove any event request from organizers.

What and Why DevOps?

What is DevOps?
DevOps is the combination of cultural philosophies, practices, and tools that increases an organization's ability to deliver applications and services at high velocity: evolving and improving products at a faster pace than organizations using traditional software development and infrastructure management processes. This speed enables organizations to better serve their customers and compete more effectively in the market.



Under a DevOps model, development and operations teams are no longer "siloed." Sometimes, these two teams are merged into a single team where the engineers work across the entire application lifecycle, from development and test to deployment to operations, and develop a range of skills not limited to a single function.

In some DevOps models, quality assurance and security teams may also become more tightly integrated with development and operations and throughout the application lifecycle. When security is the focus of everyone on a DevOps team, this is sometimes referred to as DevSecOps.

These teams use practices to automate processes that historically have been manual and slow. They use a technology stack and tooling which help them operate and evolve applications quickly and reliably. These tools also help engineers independently accomplish tasks (for example, deploying code or provisioning infrastructure) that normally would have required help from other teams, and this further increases a team's velocity.

Why DevOps?
- Speed
- Rapid Delivery
- Reliability
- Scale
- Improved Collaboration
- Security

System Configuration:

- **Operating System**
  - Ubuntu 20.04 / Windows 10/11 / MacOS 12.2.1

- **CPU and RAM**
  - 4 core + Min 2GB Required

- **Language**
  - Frontend: ReactJS, materialUI
  - Backend: NodeJS(version 16.0,0), express, mongoDB

- **Database**
  - mongoDB cloud
  - MongoDB Compass (To access the cloud DB locally in the system)

- **Building tools**
  - npm to build react project

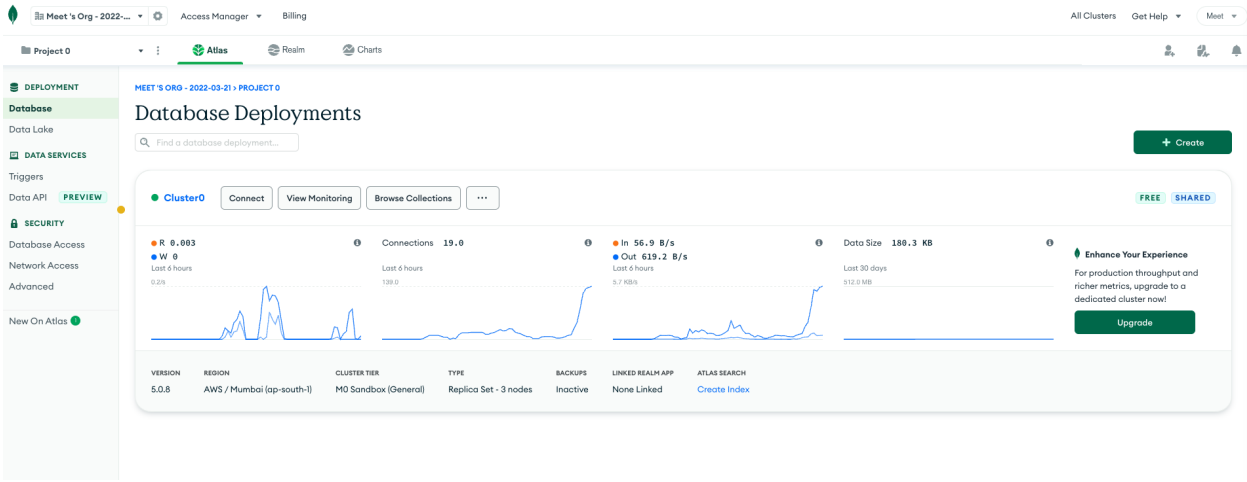
- **DevOps Tools**
  - SCM(Source Control Management) : git + github
  - Testing: Jest + Supertest
  - Build: npm
  - Continuous Integration - Jenkins
  - Containerization - Docker
  - Continuous deployment - Ansible
  - Monitoring - ELK Stack (Elastic Search, Logstash, Kibana)

Software Development Life Cycle:

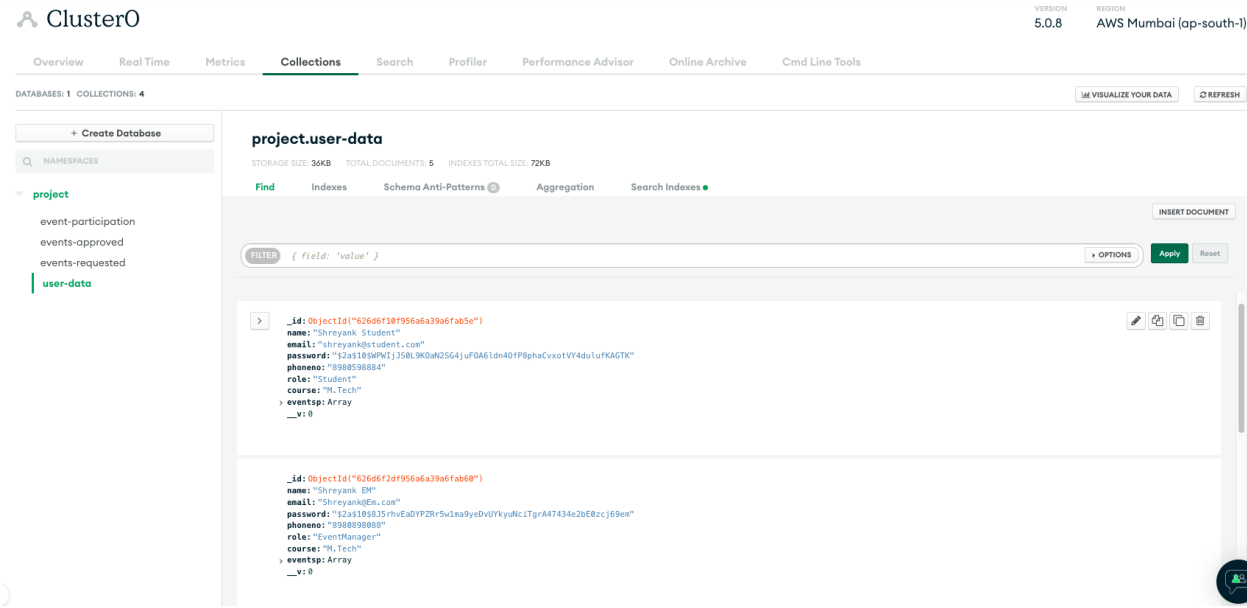
- **Installation:**
  - **ReactJS Framework**
    - React is one of the most popular JavaScript libraries in the web development field today. First, we need NPM to run the react app.
    - We can download the node from its official website https://nodejs.org/en/ which also includes the package NPM. *[Note: Recommended nodeJs version 16]*
    - Frequently used commands:
      - *npm start*
      - *npm i <package-name>*
      - *npm run build*
      - *npm run test*

  - **Node js Framework**
    - As explained in above step install NodeJS.

  - **MongoDB :**
    - To begin the sign-up process, navigate to the MongoDB Atlas sign-up page.

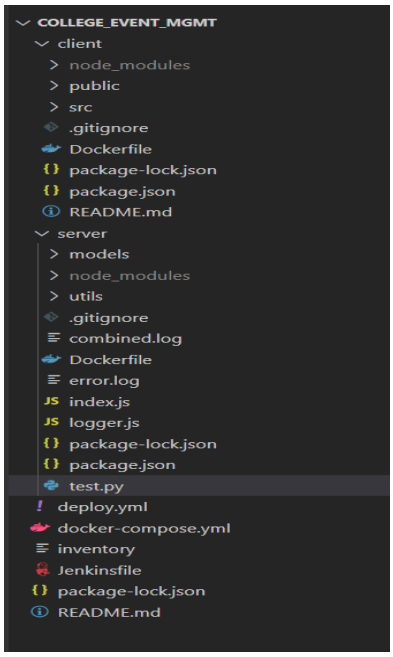


    - Above snapshot shows the MongoDB dashboard of the database.

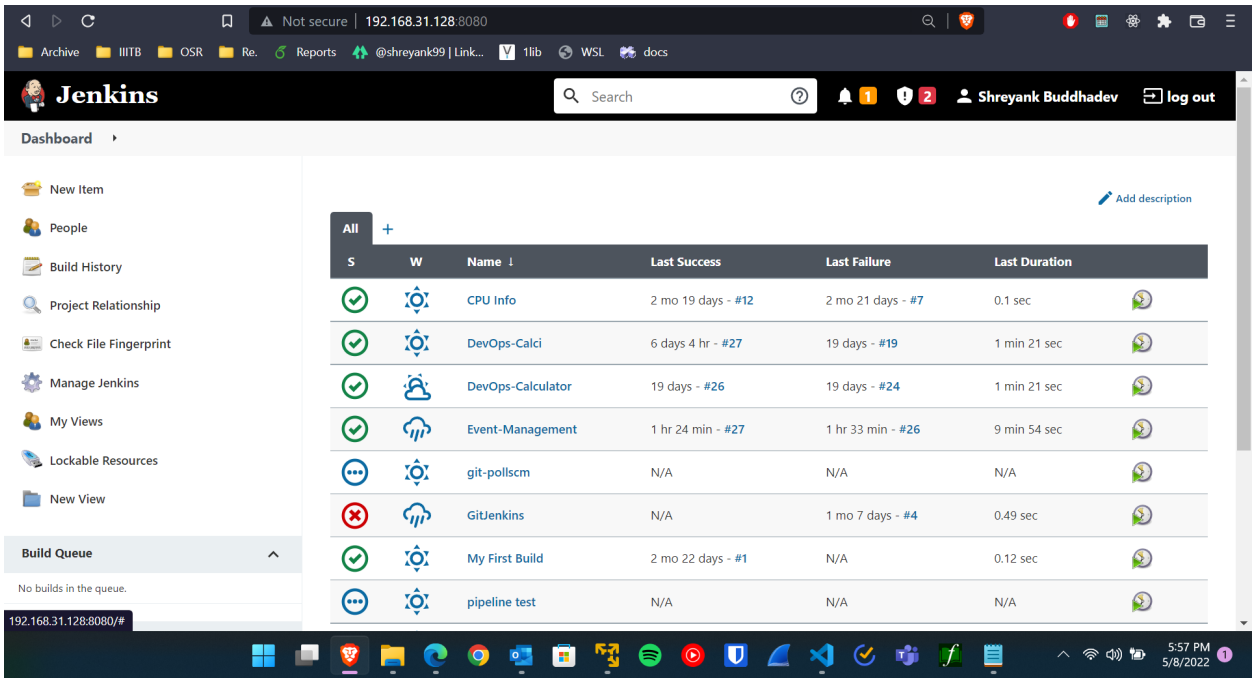


- Above snapshot shows the different tables present in the database and collection's in it.

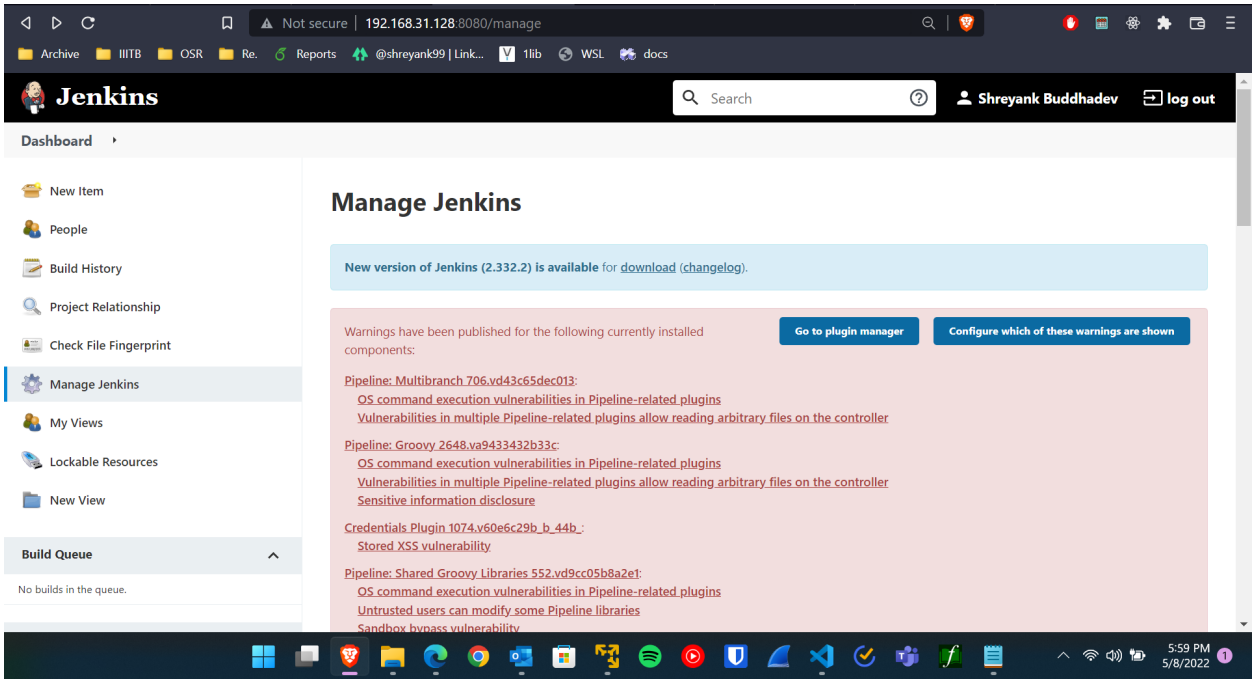
- **<u>Code Structure:</u>**
  - Below, picture depicts code structure of the project
  - *Client* folder contains front end part of the project. *Server* folder contains backend part of the project.



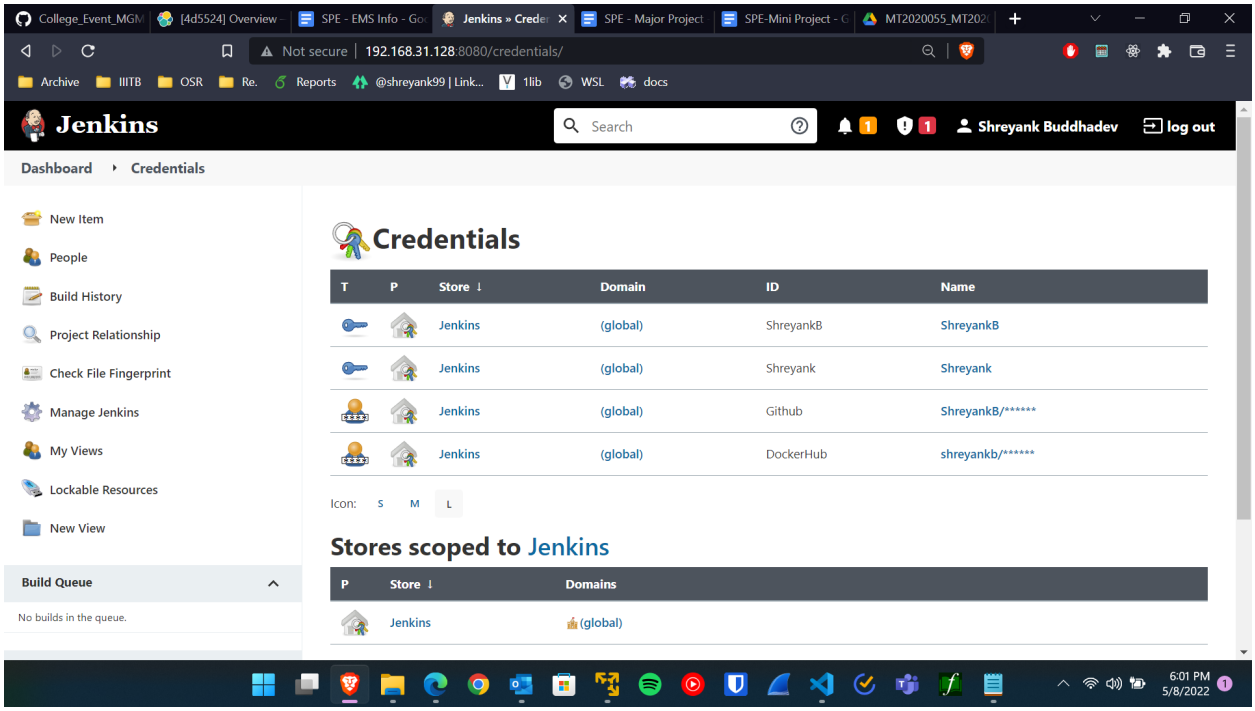
- **Pipeline Setup:**
  - Jenkins is an open-source Continuous Integration server for running a chain of actions to achieve Continuous Integration in an automated fashion. Jenkins supports the complete SDLC from building, testing,deploying, and other stages of the SDLC.
  - To start Jenkins from the terminal we run the following command command -
    - *sudo systemctl start jenkins*
  - and then we can run it on http://localhost:8080/



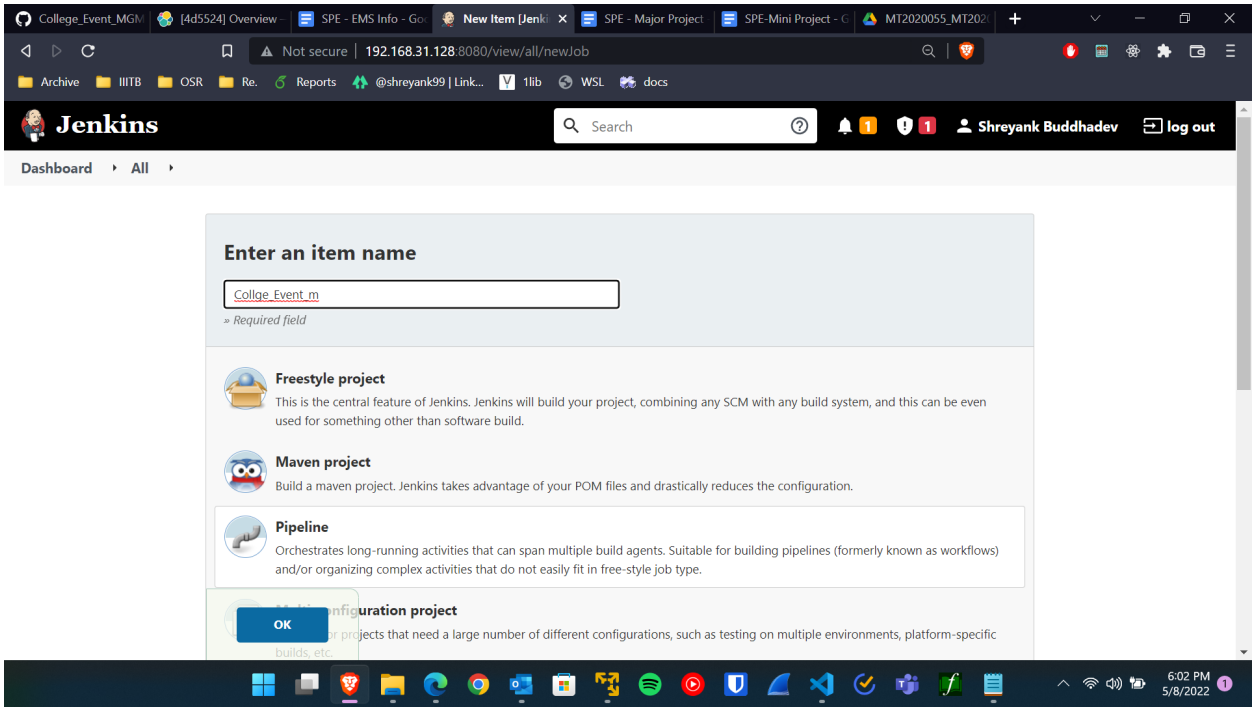
  - Now we add plugins by going to Manage Jenkins → Manage Plugins
    Then adding Docker, Git, and Ansible.



- Now we link our Github and Docker account by going to
  Manage Jenkins → Manage Credentials → global → Add Credential
  Fill username and password for git and docker separately and then save it



- We can now create a pipeline by going on
  New Item → Enter name of Project →Pipeline Project → Save it



- For various initializations in Jenkins, CI Pipeline is used to describe it respectively.

Source Control Management:

- SCM is the process of managing and tracking the changes in code. As software projects grow in lines of code and headcount of contributors, the costs of communication and management also grow. SCM is an important tool to reduce growing development costs.

- Following are some useful commands in Git -
    - **git clone <https://name-of-the-repository-link>**
        - - This will copy the entire data from the git url
    - **git branch <branch-name>**
        - - This command will make a new branch in the code repository.
    - **git status**
        - - This command displays the state of the working directory and the staging area.
    - **git add <file>**
        - This will add changes in the working directory to the staging area.
    - **git commit -m "commit message"**
        - - This command is used to save your changes to the local repository with -m used to give a small description that can help your teammates understand what happened.
    - **git push <remote> <branch-name>**
        - - This command pushes all the latest code to the repository.
    - **git pull <remote>**
        - - This command is used to update the local version of a repository from a remote.

```
stages {
    stage('Git clone') {
        steps {
            git url: 'https://github.com/meetcric/College_Event_MGMT.git', branch: 'master'
        }
    }
```

- **Installing Dependency:**
  - **NPM:**
  - Build App is a full-stack JS App build system. Inspired from create-react-app by Facebook and other client build systems it is going one step further by providing similar facilities for full-stack Java Script development.
  -
```
stage('Install dependency') {
    steps {
        sh 'cd ./client/ && npm i'
        sh 'cd ./server/ && npm i'
    }
}
```
- **Build:**
  -
```
stage('Build') {
    steps {
        sh 'cd ./client/ && npm run build'
    }
}
```
  -

- **Testing :**

We have used **Jest** and **SuperTest** for the testing of our project.
Jest is a JavaScript testing framework with a focus on simplicity. It can be installed with npm.
It fits into a category of utilities known as test runners.
To install Jest we run **npm install --save-dev jest** and then we have to change the test fieldin the package.json to jest --forceExit.
SuperTest provide a high-level abstraction for testing HTTP, while still allowing you to drop down to the lower-level API provided by superagent.
To install supertest we run **npm install –save-dev supertest.**

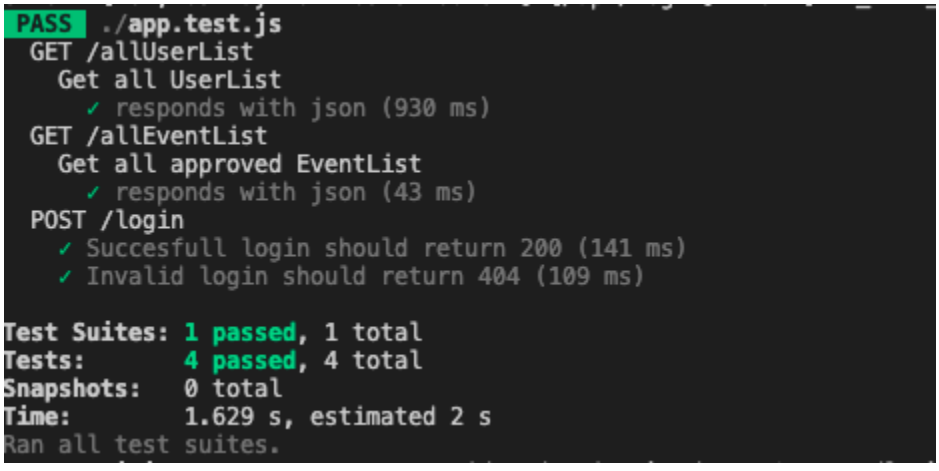
For Testing in my project, I have created a app.tests.js file which looks like this:

```
const supertest = require("supertest");
const { response } = require("./app");
const app = require("./app");

describe("GET /allUserList", () => {
  describe("Get all UserList", () => {
    it("responds with json", function (done) {
      supertest(app)
        .get("/api/allUserList")
        .set("Accept", "application/json")
        .expect("Content-Type", /json/)
        .expect(200, done);
    });
  });
});

describe("GET /allEventList", () => {
  describe("Get all approved EventList", () => {
    it("responds with json", function (done) {
      supertest(app)
        .get("/api/allEvents")
        .set("Accept", "application/json")
        .expect("Content-Type", /json/)
        .expect(200, done);
    });
  });
});
```

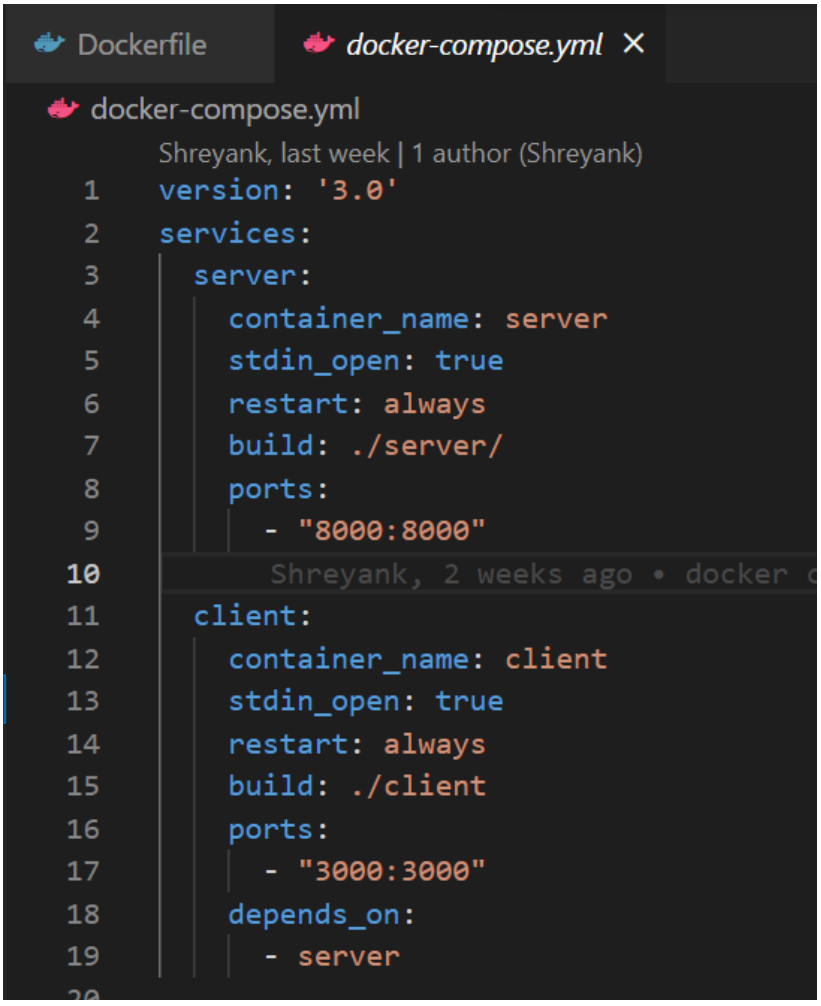
Now to do testing we have to run npm run test or npm test in the terminal. Here is the testing and result of our Project -



- **<u>Docker Artifact:</u>**
    - Docker is a tool that makes it easy to create, deploy, and run applications by using containers. Containers allow you to package up an application with every part it needs, like libraries and other dependencies, and deploy it as one package.
    - Docker hub is the world's largest repository of container images with an array of content sources including container community developers, open-source projects, and independent software vendor building and distributing their code in containers.
    - Some useful commands in Docker are as follows -
        - docker images
        - sudo docker run -it name container ID
        - sudo docker service ls
        - sudo docker swarm it

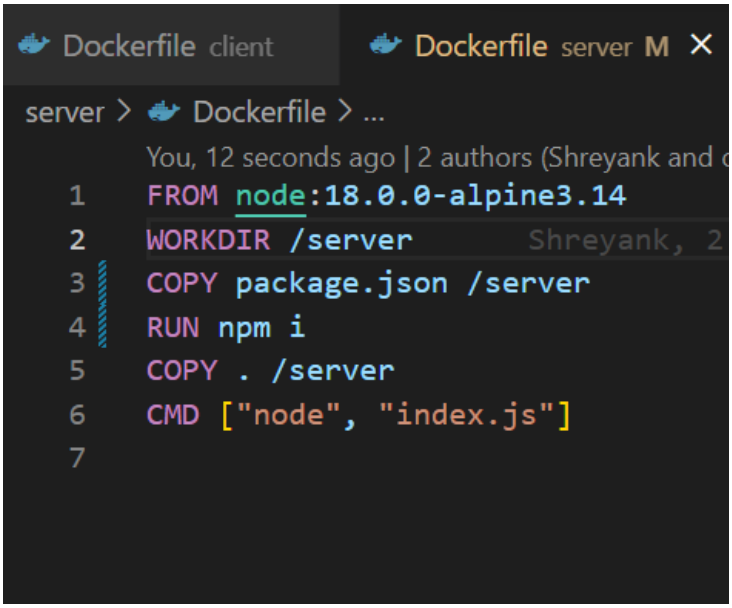
```
stage('Docker Build') {
    steps {
        // delete if image already exists
        // sh 'docker rm college_event_management_server college_event_management_server'
        sh 'docker-compose build'
        sh 'docker tag event-management_server:latest shreyankb/event_management_server:latest'
        sh 'docker tag event-management_client:latest shreyankb/event_management_client:latest'
    }
}
stage('Docker Push') {
    steps {
        withCredentials([usernamePassword(credentialsId: 'DockerHub', passwordVariable: 'dockerHubPassword', usernameVariable: 'dockerHubUser')]) {
        sh "docker login -u ${env.dockerHubUser} -p ${env.dockerHubPassword}"
        sh 'docker push shreyankb/event_management_server:latest'
        sh 'docker push shreyankb/event_management_client:latest'
    }
    }
}
stage('Clean Docker Images') {
    steps {
        sh 'docker rmi -f shreyankb/event_management_server'
        sh 'docker rmi -f shreyankb/event_management_client'
    }
}
```

    - **Docker-compose:**
    - Docker Compose is a tool for defining and running multi-container Docker applications. With Compose, you use a YAML file to configure your application's services. Then, with a single command, you create and start all the services from your configuration.
    - Docker Compose commands :
        - *docker-compose up* : Start all services(it will run docker-compose.yml file)
        - *docker-compose build* : This command will build the images with the help of docker-compose.yml file.
        - *docker-compose down* : Stop all the services.

- 

- **Docker file for frontend:**

- **Docker file for backend:**

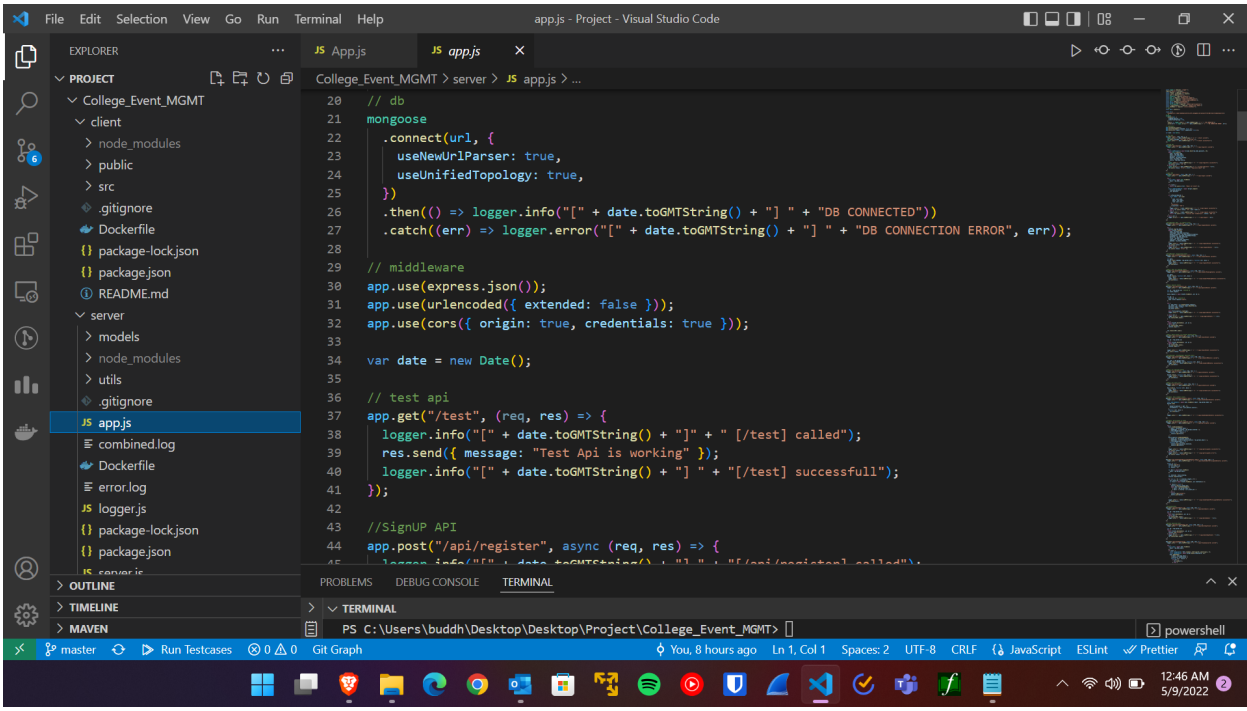


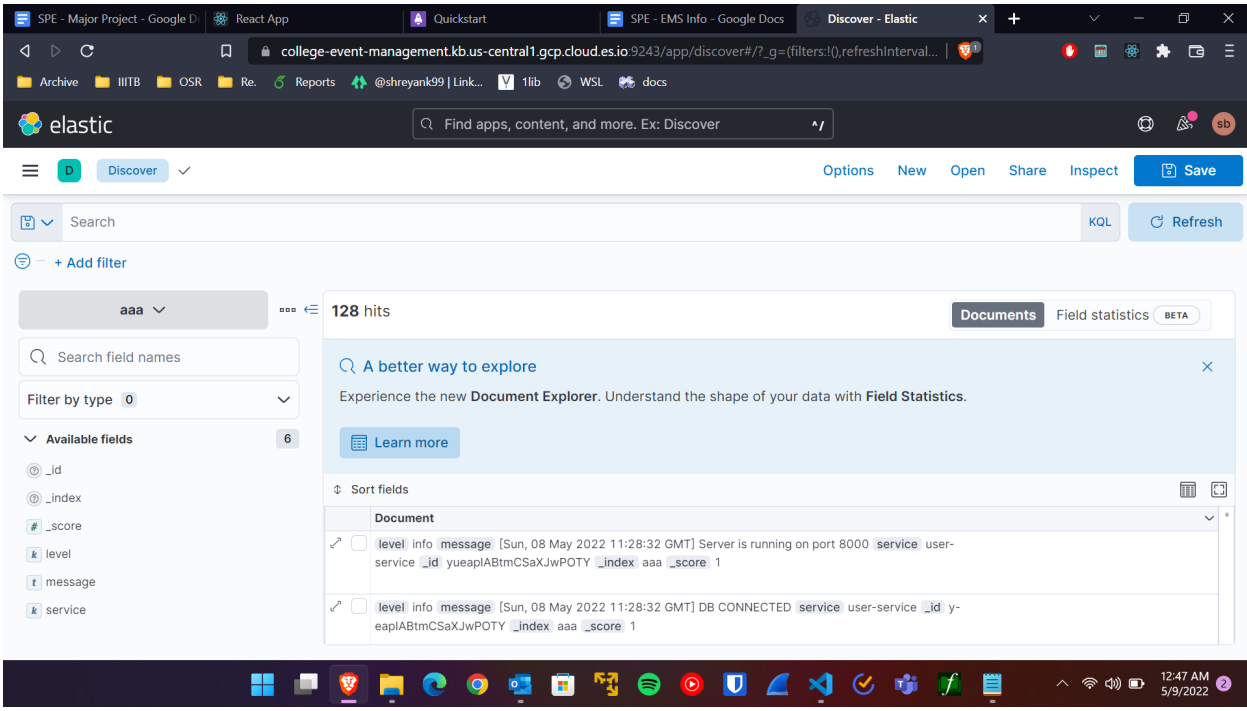
- **Deploy**

  - **Ansible:**
  - Ansible is an open source IT Configuration Management, Deployment & Orchestration tool. This tool is very simple to use yet powerful enough to automate complex multi-tier IT application environments.It runs on many Unix-like systems and can configure both Unix-like systems and Windows.

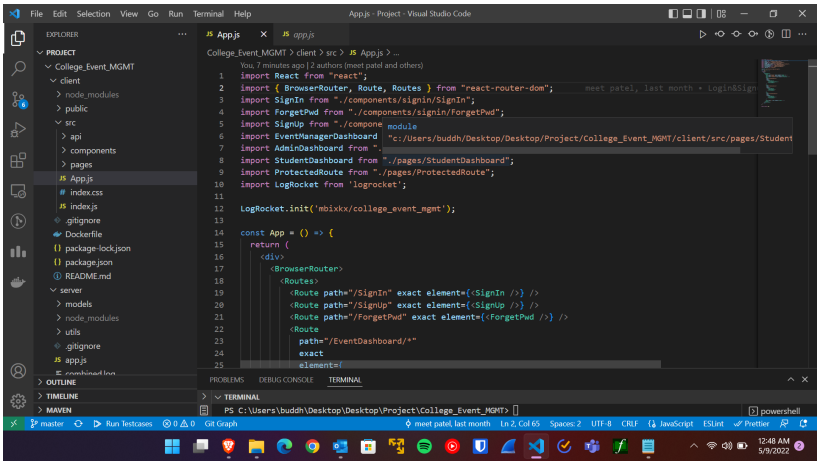
```
stage('Ansible Deploy') {
    steps {
        ansiblePlaybook colorized: true, disableHostKeyChecking: true, installation: 'Ansible', inventory: 'inventory', playbook: 'deploy.yml'
    }
}
```

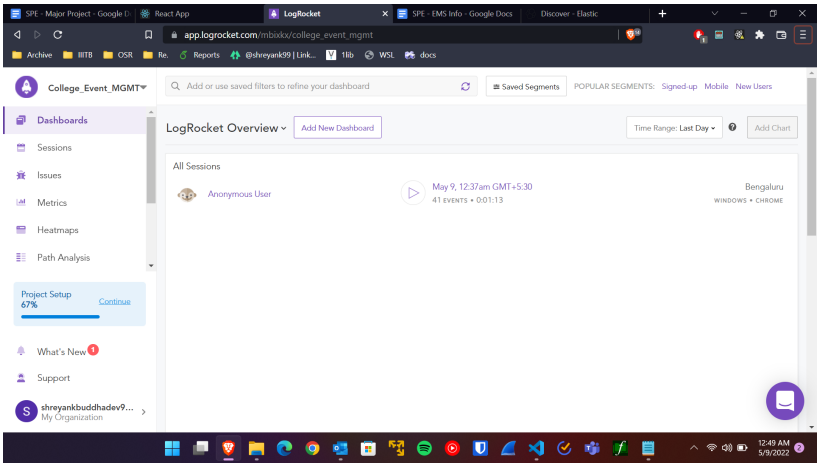
- **Monitor**
  - **Monitoring for backend**
  - To monitor the backend logs ELK Stack is used. **Winston logger** is used to generate the backend logs.

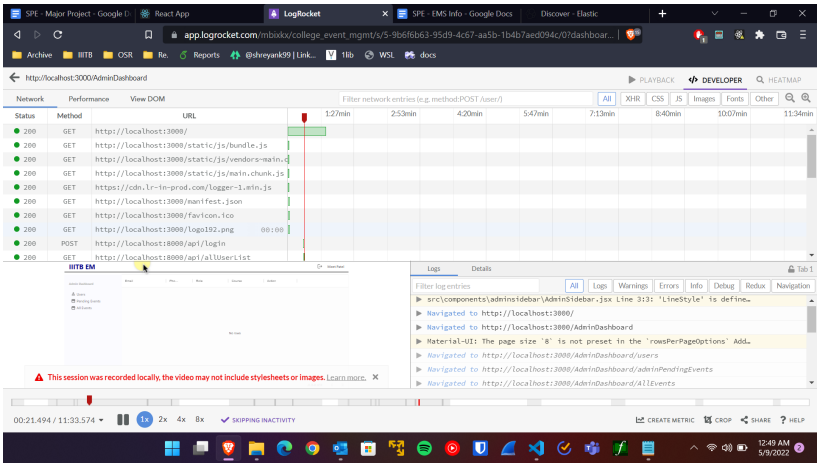




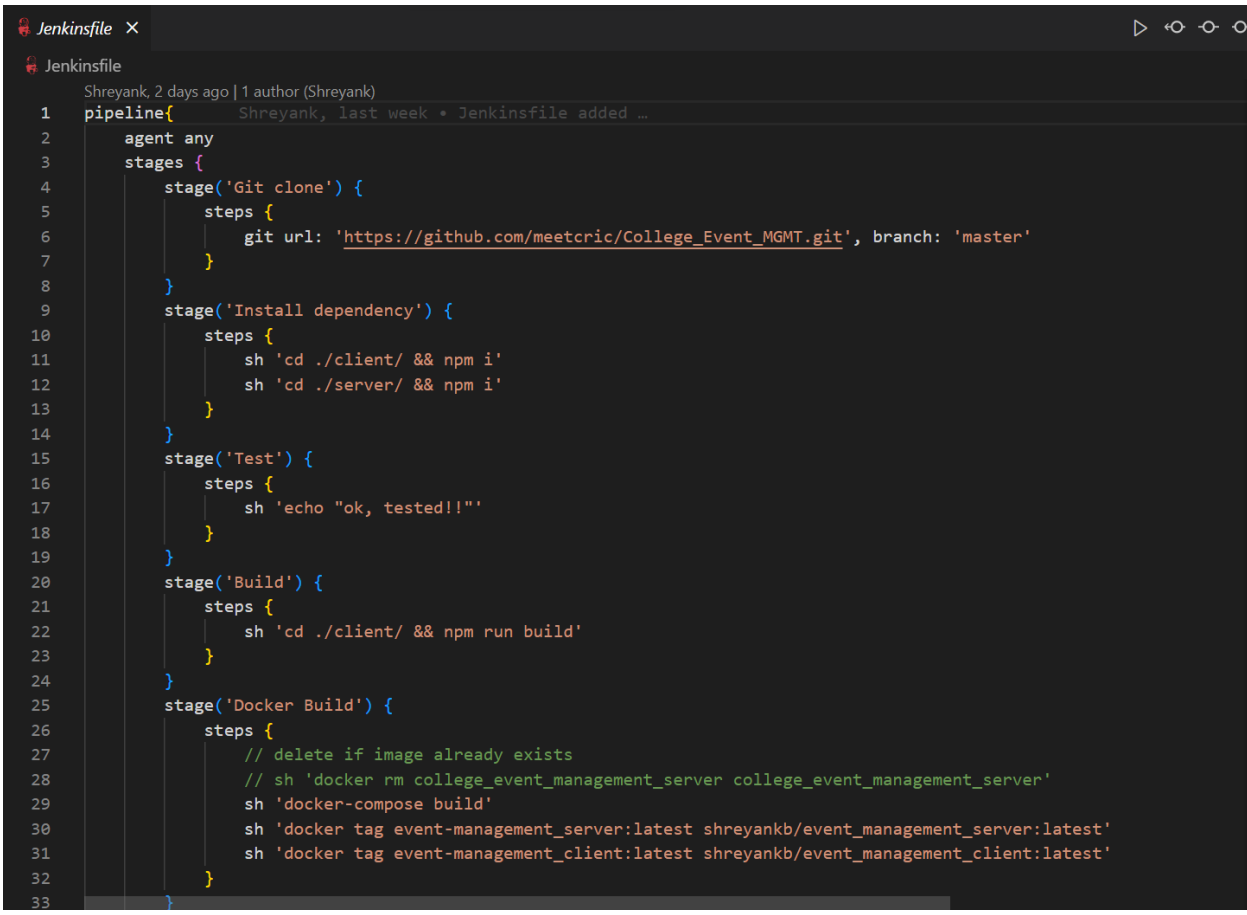
- **Monitoring for frontend**
- To monitor front end logs **logrocket** is used. It records all front end activities and generates a video for it as well.







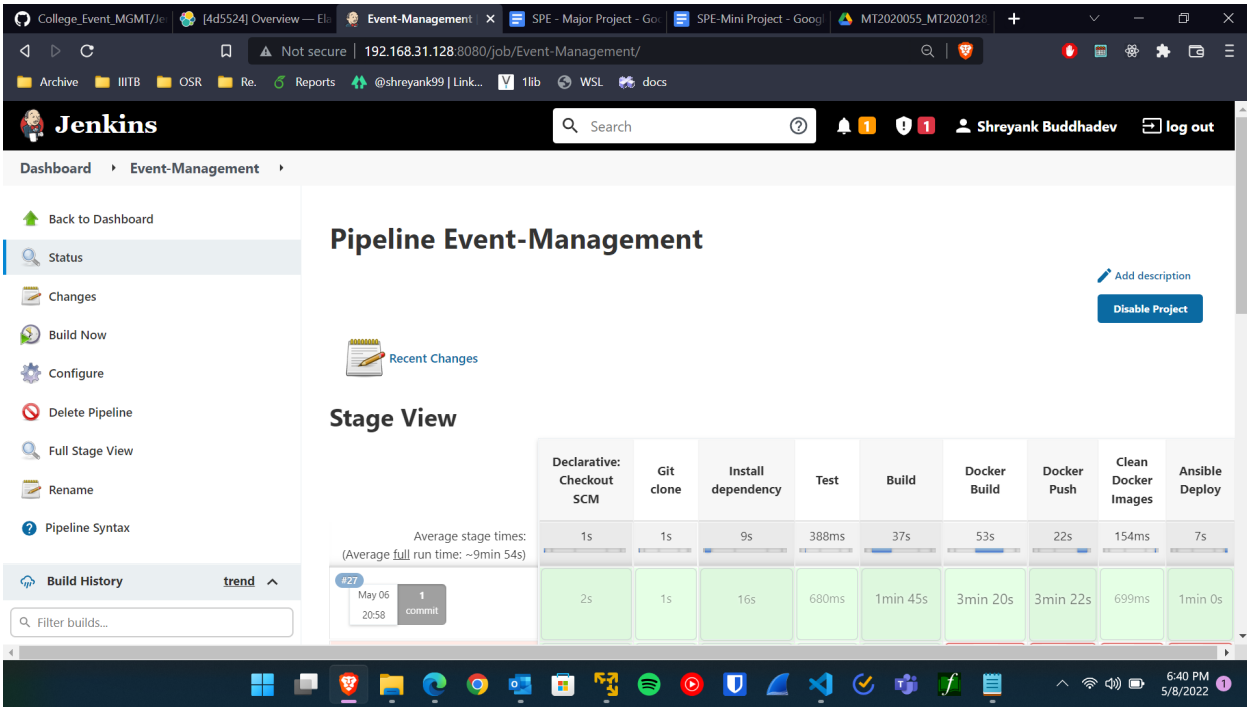
- **Building Pipeline and Running on Machine:**
  - In the above sections we have integrated npm, ansible, docker and other necessary tools with jenkins. We were able to trigger ansible from Jenkins and also build docker images with Jenkins. Here is a figure of the entire pipeline and its final full-stage view after a successful run of Jenkins pipeline.



```
Jenkinsfile
Jenkinsfile
Shreyank, 2 days ago | 1 author (Shreyank)
pipeline{       Shreyank, last week • Jenkinsfile added …
    agent any
    stages {
        stage('Git clone') {
            steps {
                git url: 'https://github.com/meetcric/College_Event_MGMT.git', branch: 'master'
            }
        }
        stage('Install dependency') {
            steps {
                sh 'cd ./client/ && npm i'
                sh 'cd ./server/ && npm i'
            }
        }
        stage('Test') {
            steps {
                sh 'echo "ok, tested!!"'
            }
        }
        stage('Build') {
            steps {
                sh 'cd ./client/ && npm run build'
            }
        }
        stage('Docker Build') {
            steps {
                // delete if image already exists
                // sh 'docker rm college_event_management_server college_event_management_server'
                sh 'docker-compose build'
                sh 'docker tag event-management_server:latest shreyankb/event_management_server:latest'
                sh 'docker tag event-management_client:latest shreyankb/event_management_client:latest'
            }
        }
```

```
34          stage('Docker Push') {
35              steps {
36                  withCredentials([usernamePassword(credentialsId: 'DockerHub', passwordVariable: 'dockerHubPassword',
37                  sh "docker login -u ${env.dockerHubUser} -p ${env.dockerHubPassword}"
38                  sh 'docker push shreyankb/event_management_server:latest'
39                  sh 'docker push shreyankb/event_management_client:latest'
40              }
41              }
42          }
43          stage('Clean Docker Images') {
44              steps {
45                  sh 'docker rmi -f shreyankb/event_management_server'
46                  sh 'docker rmi -f shreyankb/event_management_client'
47              }
48          }
49          stage('Ansible Deploy') {
50               steps {
51                    ansiblePlaybook colorized: true, disableHostKeyChecking: true, installation: 'Ansible', inventory:
52               }
53          }
54      }
55  }
56
```

- Here is the successful build of of our Jenkins Pipeline



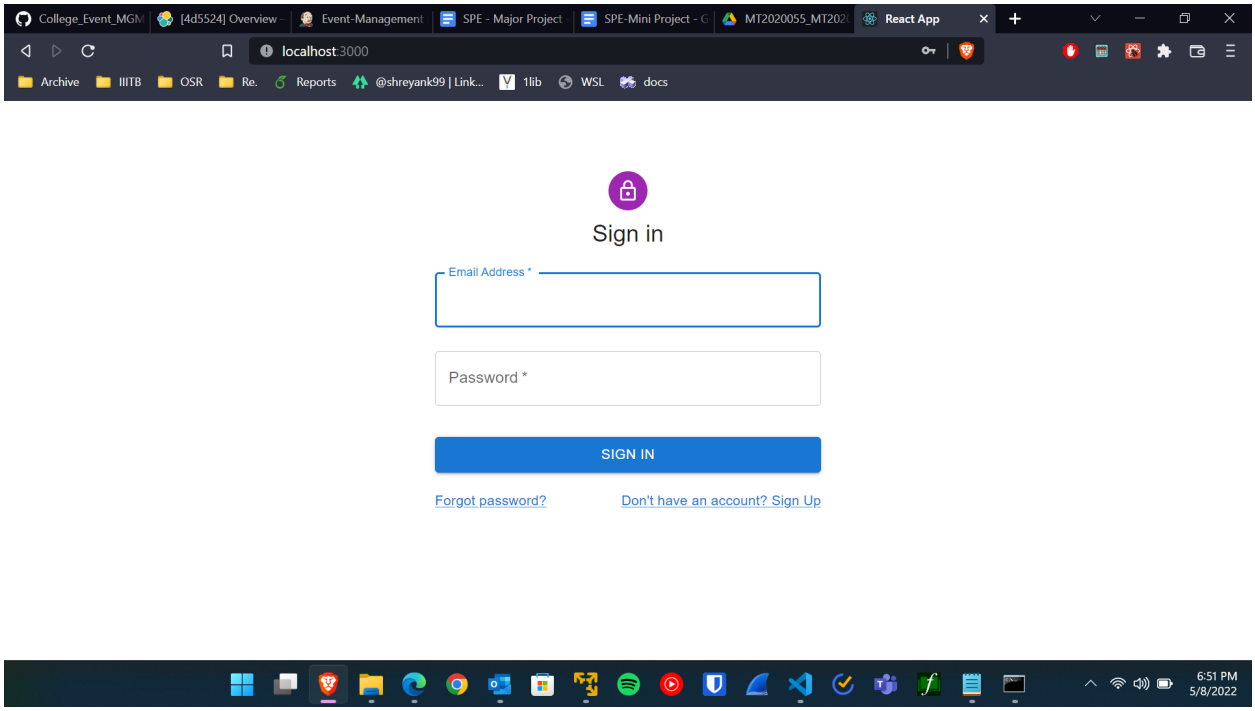
- The docker image which is being pushed to the local host of our system is as shown in the below figure
- This can be checked using the command : docker images

```
Last login: Fri May  6 21:07:39 2022 from 127.0.0.1
doshreyank@ubuntu:~$ docker images
REPOSITORY                          TAG                 IMAGE ID            CREATED             SIZE
event-management_client             latest              ed4d85f18e2b        46 hours ago        28MB
shreyankb/event_management_client   latest              ed4d85f18e2b        46 hours ago        28MB
event-management_server             latest              dd22e0908cae        46 hours ago        189MB
shreyankb/event_management_server   latest              dd22e0908cae        46 hours ago        189MB
```

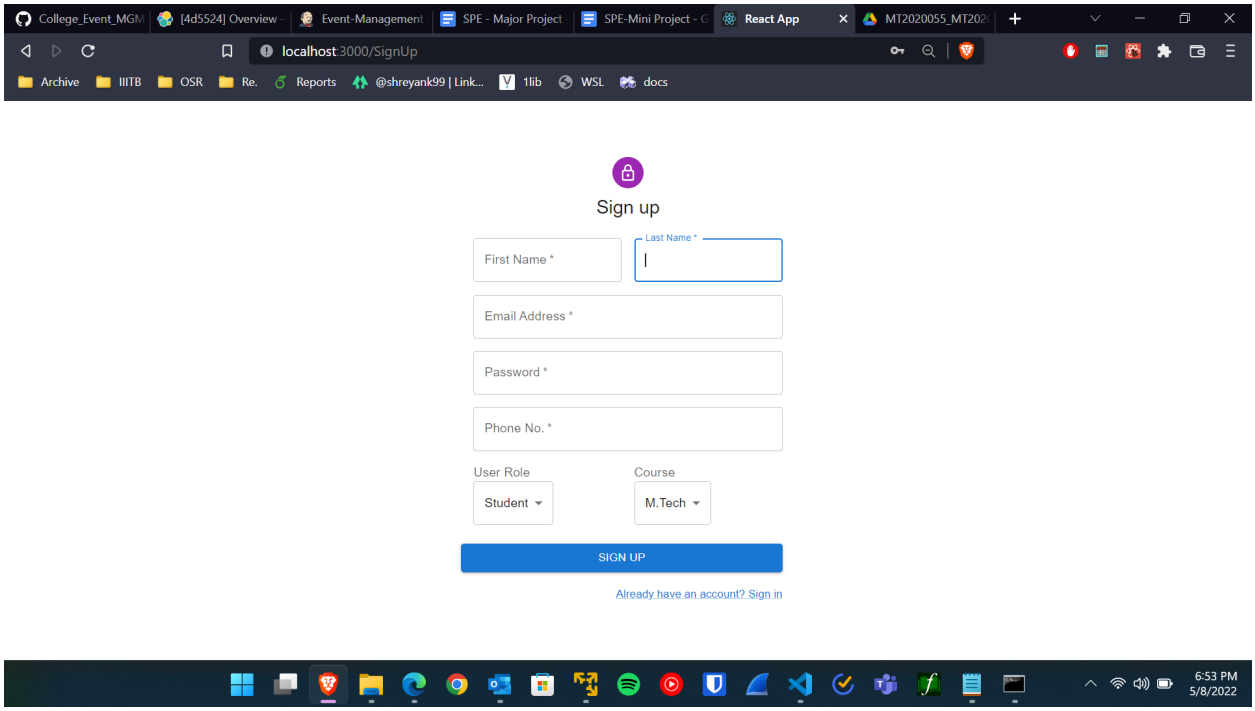
- We don't have to run docker images on client machine to run our code because docker-compose did it for us. So now our server is running on http://localhost:8000 url and Frontend running on http://localhost:3000 url.

Result and Discussion

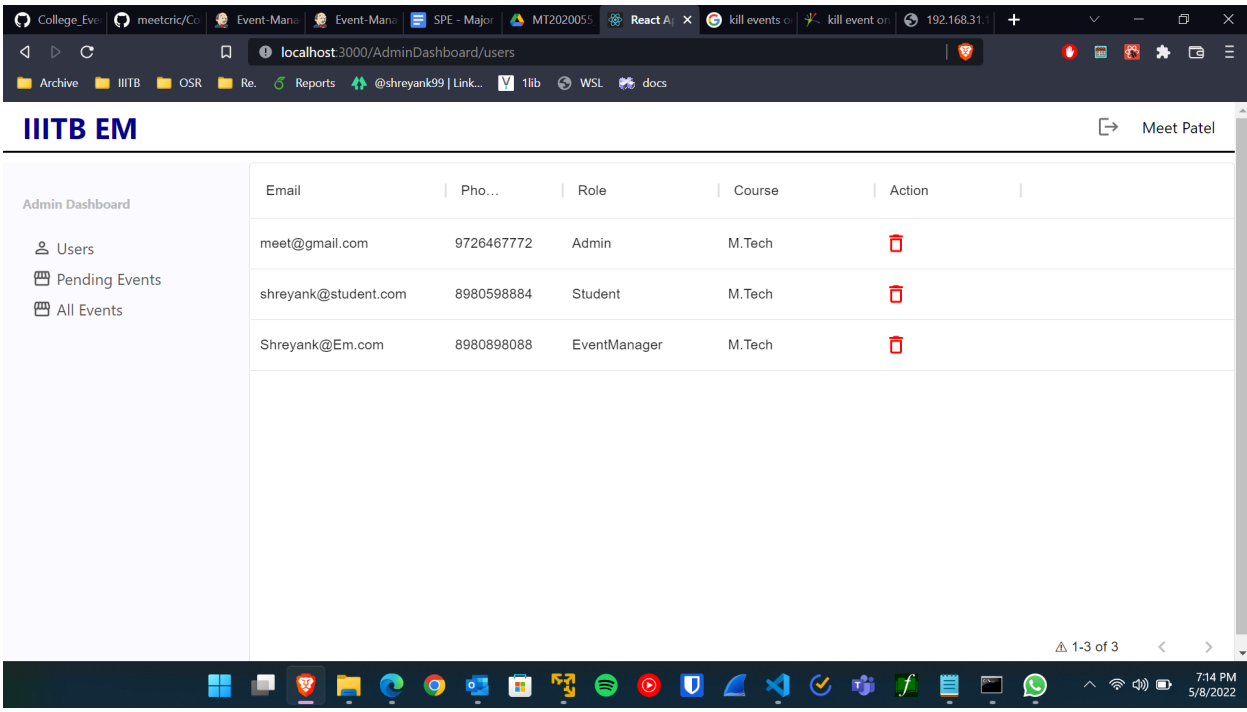
- **Sign In page:**



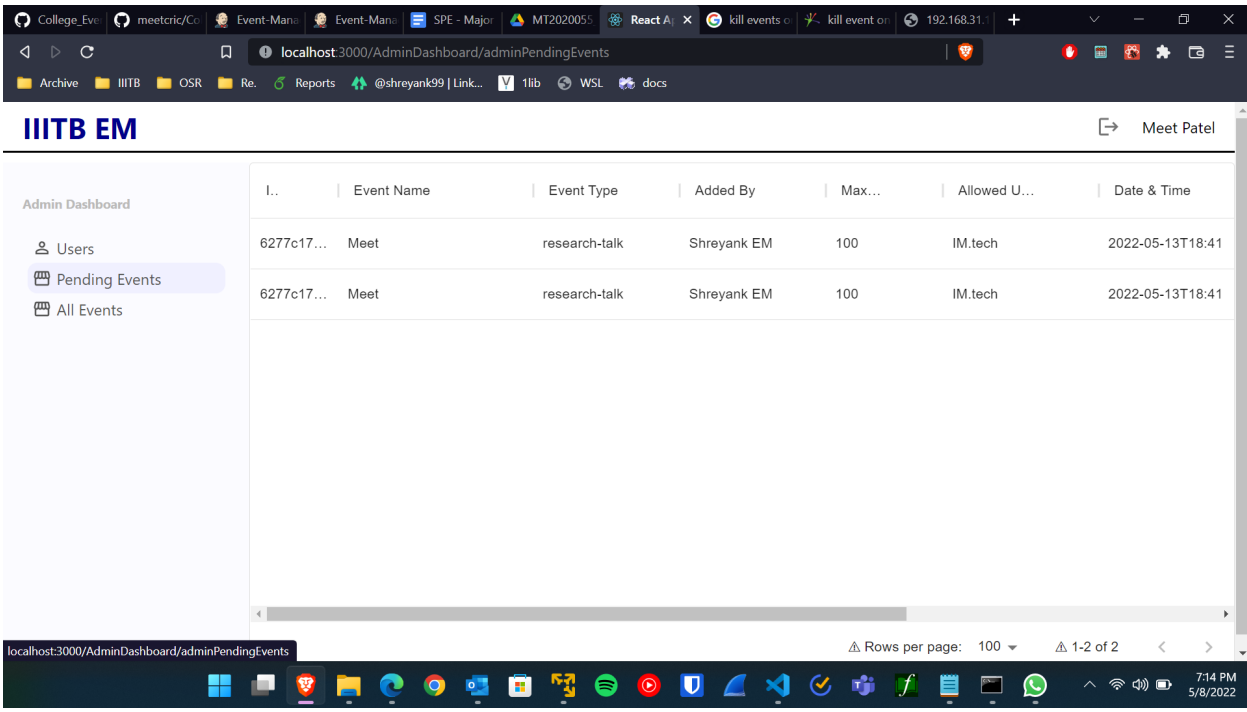
- **Register Page:**



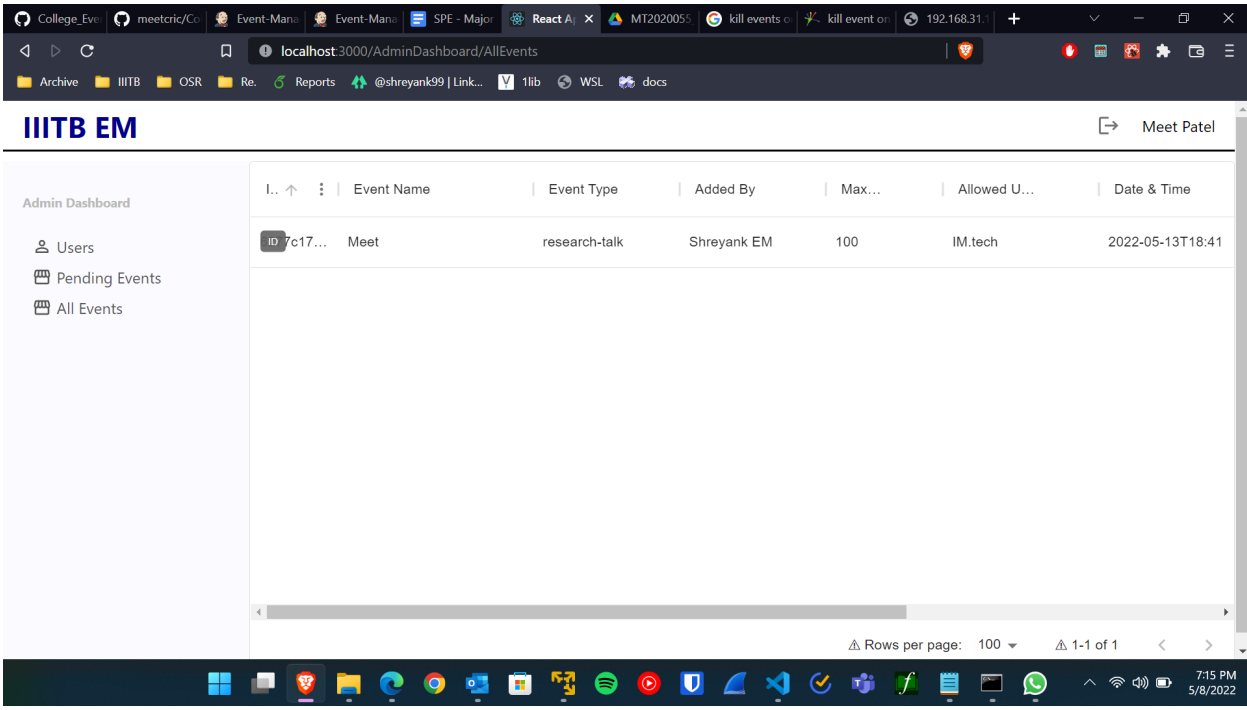
- **Admin Dashboard**
  - **User tab**



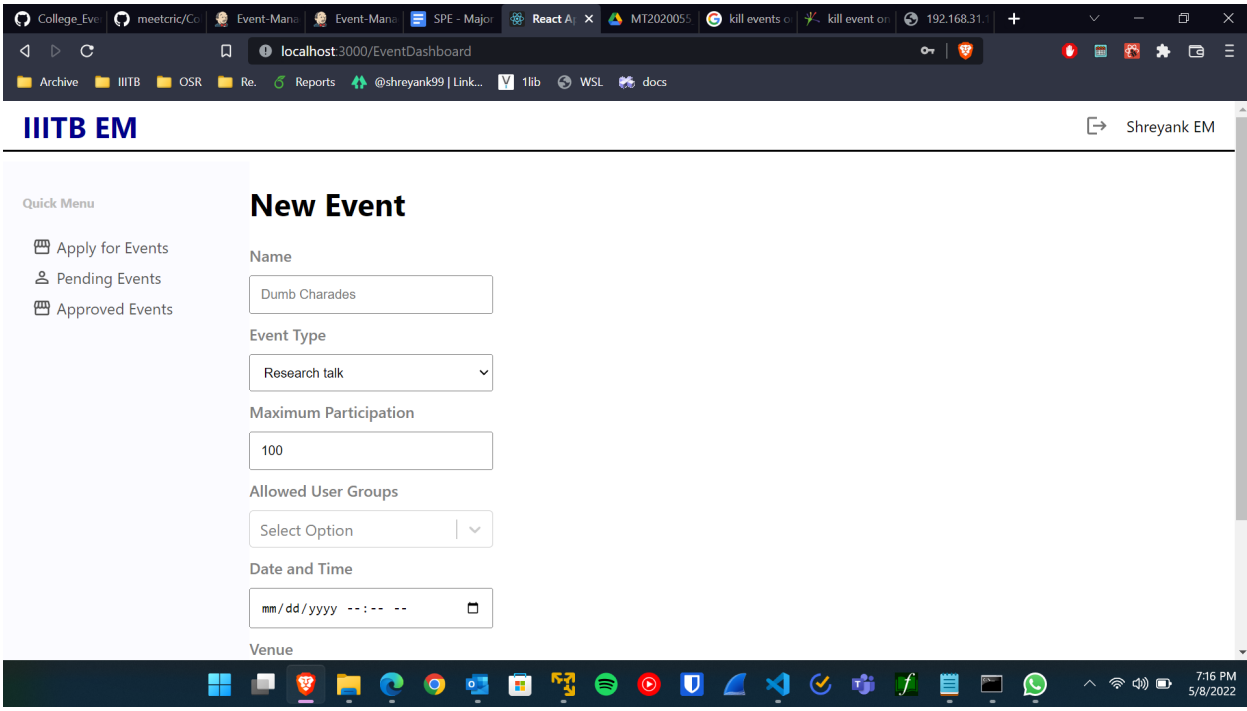
  - **Pending Events**



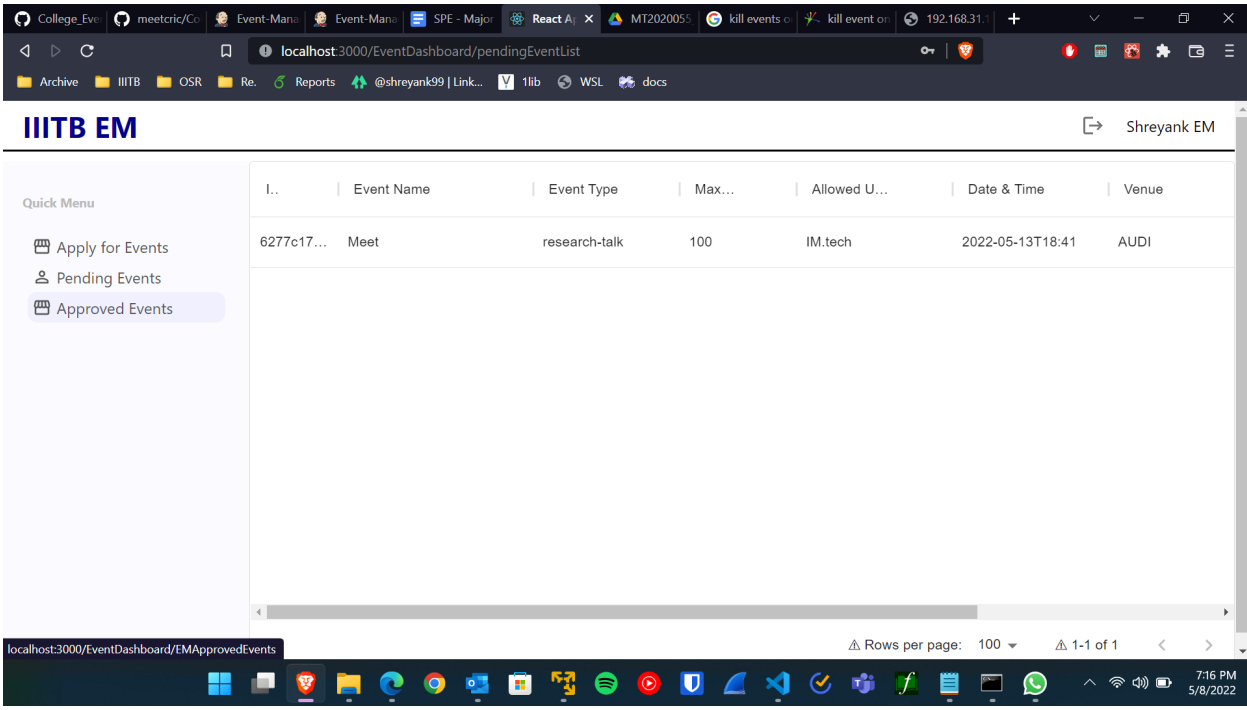
  - **All Events**



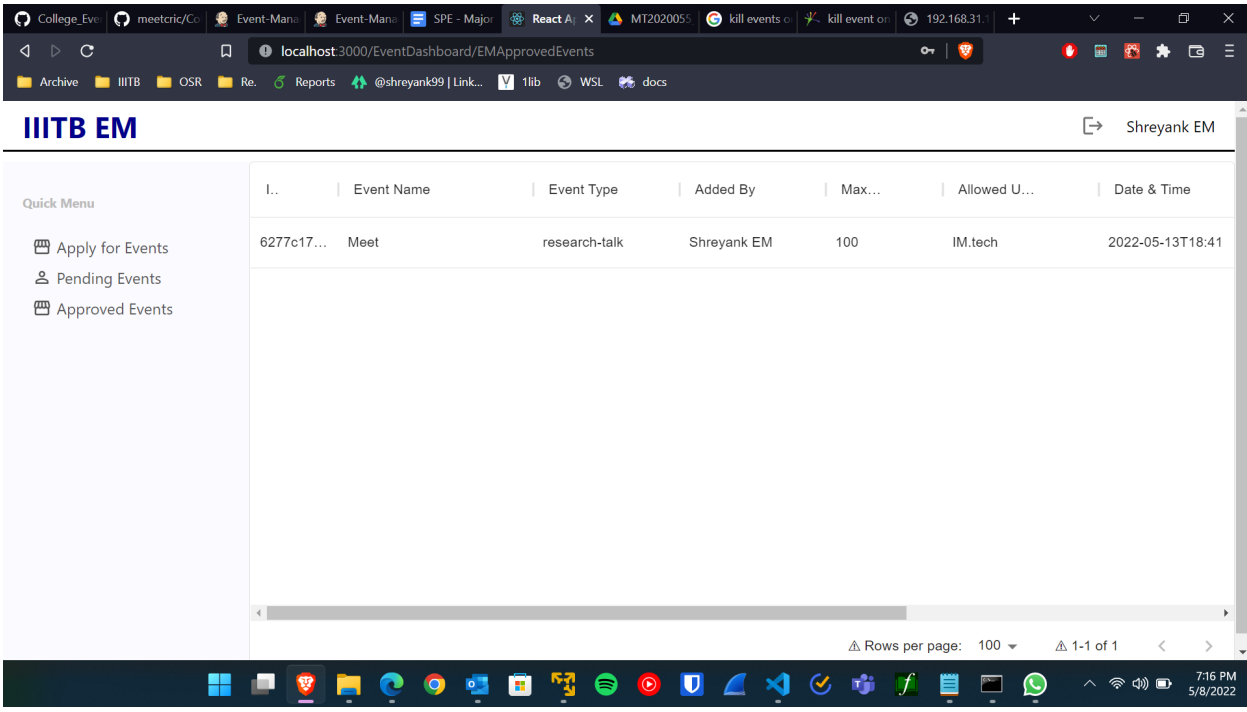
- **Event Manager Dashboard**
  - **Apply for Events**



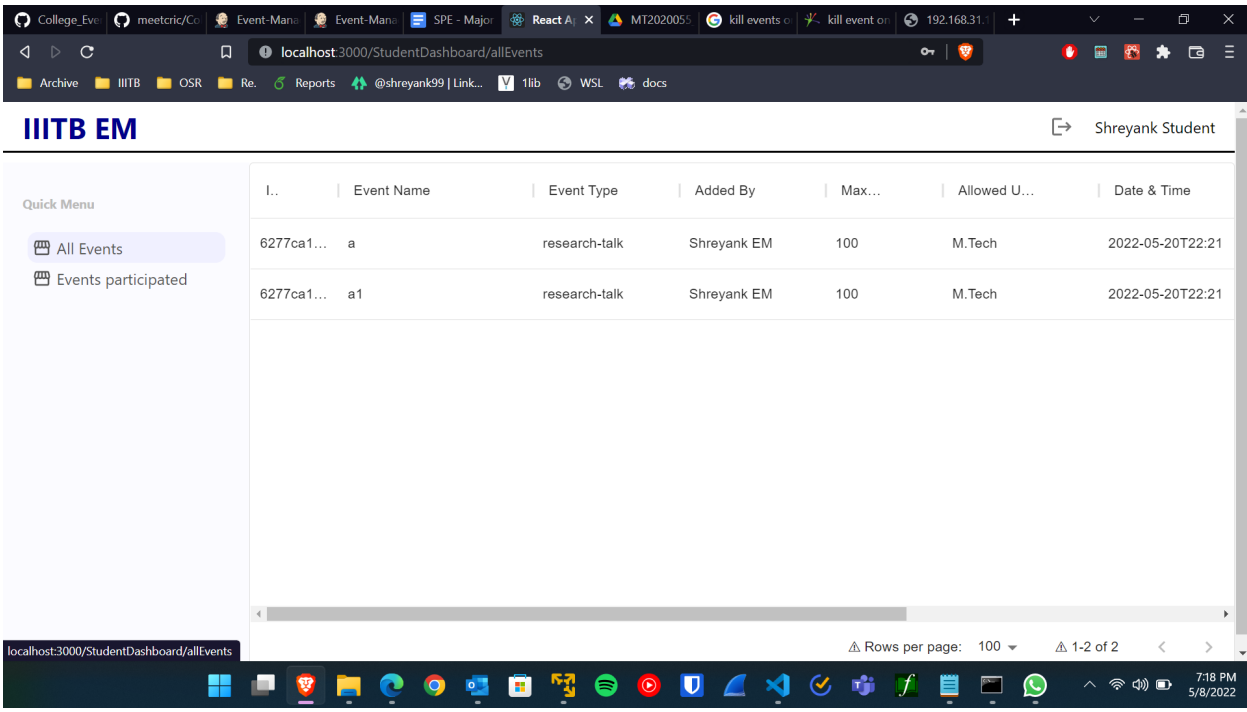
  - **Pending Events**



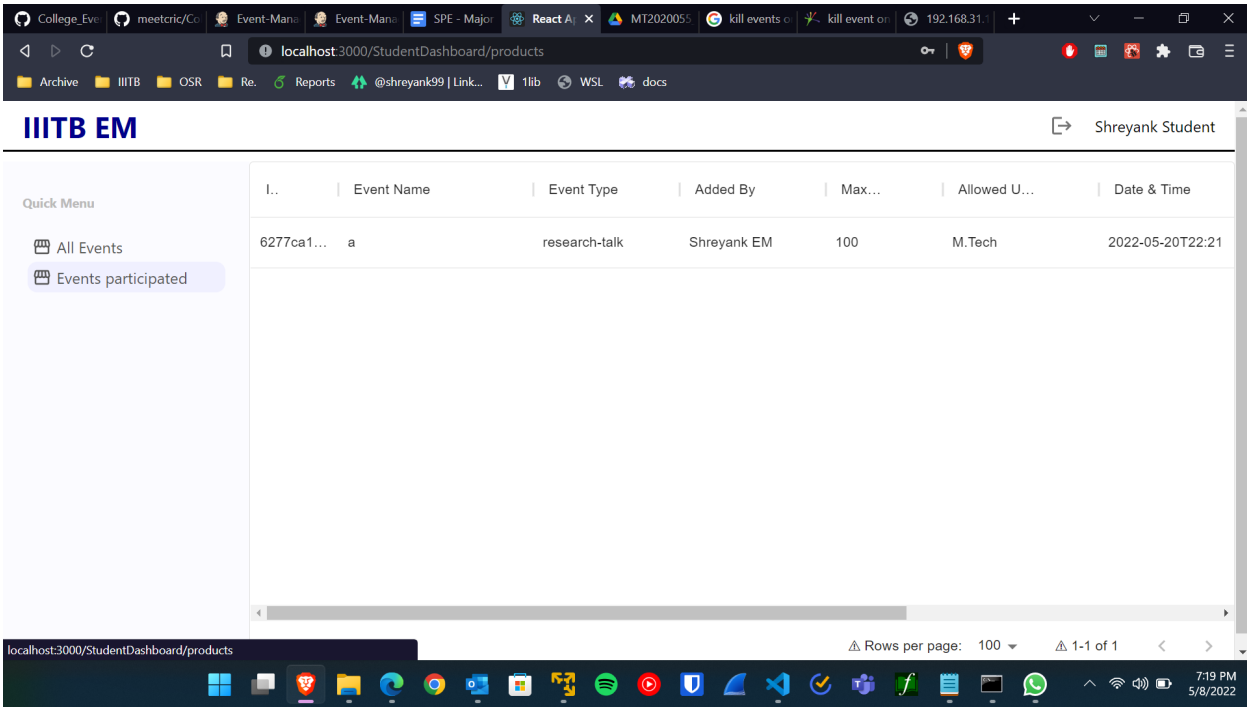
- **Approved Events**



- **User Dashboard**
  - **All Events**



  - **Events Participated**



Scope for future works

Cloud deployment

We can deploy our project using amazon AWS and other cloud services which makes our project platform-independent so that it can be used from anywhere on the earth.

Generalization

We can also generalise this project and increase its scope to accommodate all the requirement of a college event management portal,So that next year INFIN8 team can use this code instead of using google form to manage events.

Other Feature Enhancement

We can build more modules like live Dashboard for admin,Event managager. Such modules will help in showing live statics of event.
We can enhance the Student experience by including modules like result of each event and other required features.(Aim to provide the experience same as HackerEarth).
We can emphasize more on the security of the both frontend,backend part.

Conclusion

We successfully built an College event management system using DevOps tools. The tools we used are: GitHub, Jenkins, Docker, Ansible. These tools are integrated using Jenkins. The entire pipeline has been automated. For deployment of the entire project, it's taking a minimum of 15 minutes.
The Devops methodology and lifecycle tools prove to be better than the Agile methodology in terms of technical, cultural and business benefits. By minimizing friction between independent teams, DevOps enables a collaborative approach for enterprise software development and delivery that reflects the needs of the entire application lifecycle for today's modern enterprises.
Thus, we can develop, test, deploy and monitor the application easily.