INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY-BANGALORE

CS 816 - SOFTWARE PRODUCTION ENGINEERING

Final Project

Expense Sharing WebApp

Team Members:-

Aman Gupta (MT2021012)

Kunal Sharma (MT2021070)

Nikhil Mittal (MT2021083)

Under Guidance of:-

Prof. B. Thangaraju

Teaching Assistant:-

Anshul Garg

Table of Contents

1. Abstract

Expense Sharing is a web app designed to fulfil the needs of the user by reducing their efforts for the settlement of the bill. The application encourages corresponding users to help in who owes who, and for what. Aim to provide users the best approach to help users and their companion to share expenses easily.

The architecture of our project demands three layers.
- Front end
- Middle layer
- Back end

The front end of the project is handled by “Angular” Framework.The middle layer is built on “SpringBoot” Framework and communicates with mysql database to show the content on the front end via Rest Api.

2. Introduction

2.1 Overview

In the past, splitting the cost of bills and expenses was a clumsy process. You had to calculate who owes what, then go through the tedious task of collecting money from each person. And since fewer people are carrying cash less often these days, settling up gets even more difficult. That is, until an expense-sharing web app.

Expense Sharing web app lets you divide up the cost of things like group dinners and group trips so that everyone can pay their fair share. With this application, you'll be able to keep the peace between friends by cutting down on squabbles about who owes what to whom.

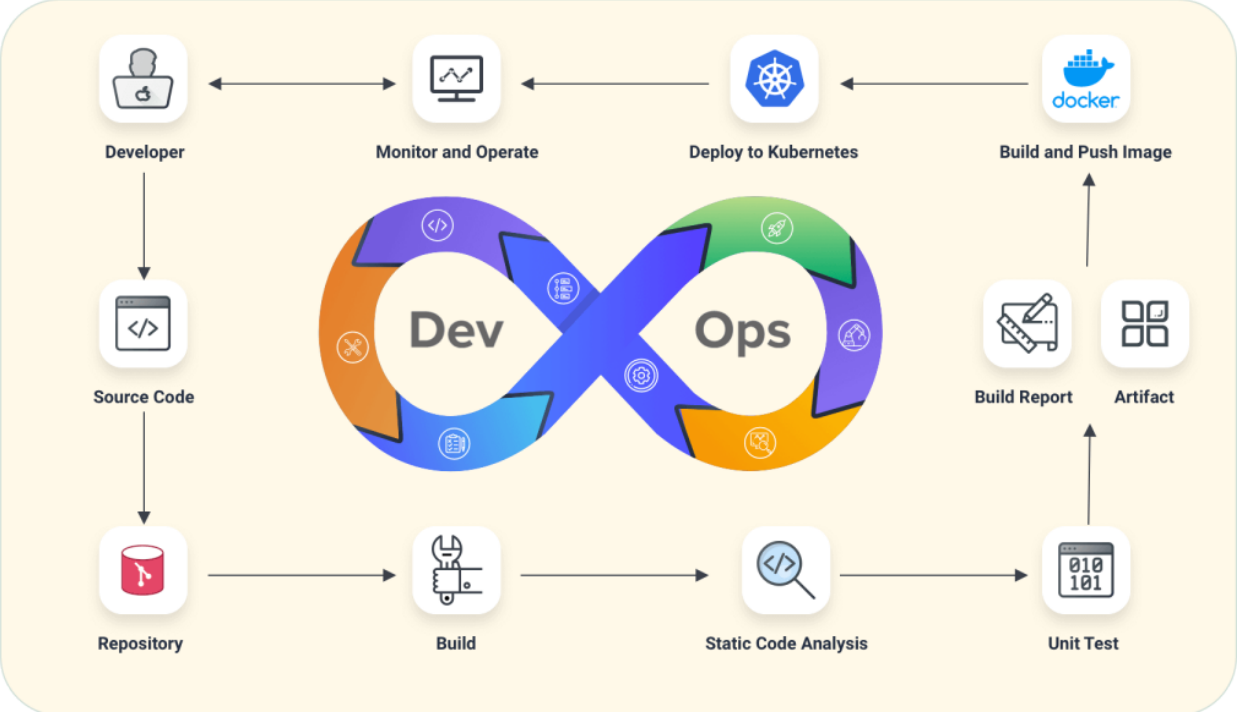
2.2 Features

- Users can create a particular group with friends and family and can divide expenses among the members of the group.
- Users can keep track of shared expenses, balances, and who owes who.
- Split expenses with any group: trips, housemates, friends, and family.
- Users can quickly add expenses on the go before you forget who paid.
- Settle up with a friend and record any cash or online payment.

2.3 Why DevOps

DevOps is a set of practices that automates the processes between software development and IT teams, in order that they can build, test, and release software faster and more reliably. The concept of DevOps is founded on building a culture of collaboration between teams that historically functioned in relative siloes. The promised benefits include increased trust, faster software releases, ability to solve critical issues quickly, and better manage unplanned work.

It unites agile, continuous delivery, automation, and much more, to help development and operations teams be more efficient, innovate faster, and deliver higher value to businesses and customers. Thus DevOps is preferred as it helps in faster release time and better management of unplanned work.



Figure(1)- DevOps Model

3. System Configuration

3.1 Operating System

Ubuntu 20.04 LTS

3.2 CPU and RAM

4 core processor and RAM 8 GB

3.3 Language

JavaScript XML,JSX, Angular web Framework

Java, Spring framework

3.4 Kernel Version

Linux Machine 5.13.0-40-generic

3.5 Database

MySQL 8.0.29 (MySQL Community Server - GPL)

3.6 Building Tools

Maven to build java application..

3.7 DevOps Tools

- Source Control Management - GitHub
- Continuous Integration - Jenkins
- Containerization - Docker
- Continuous deployment - Ansible
- Monitoring - ELK Stack (Elastic Search, Logstash, Kibana).

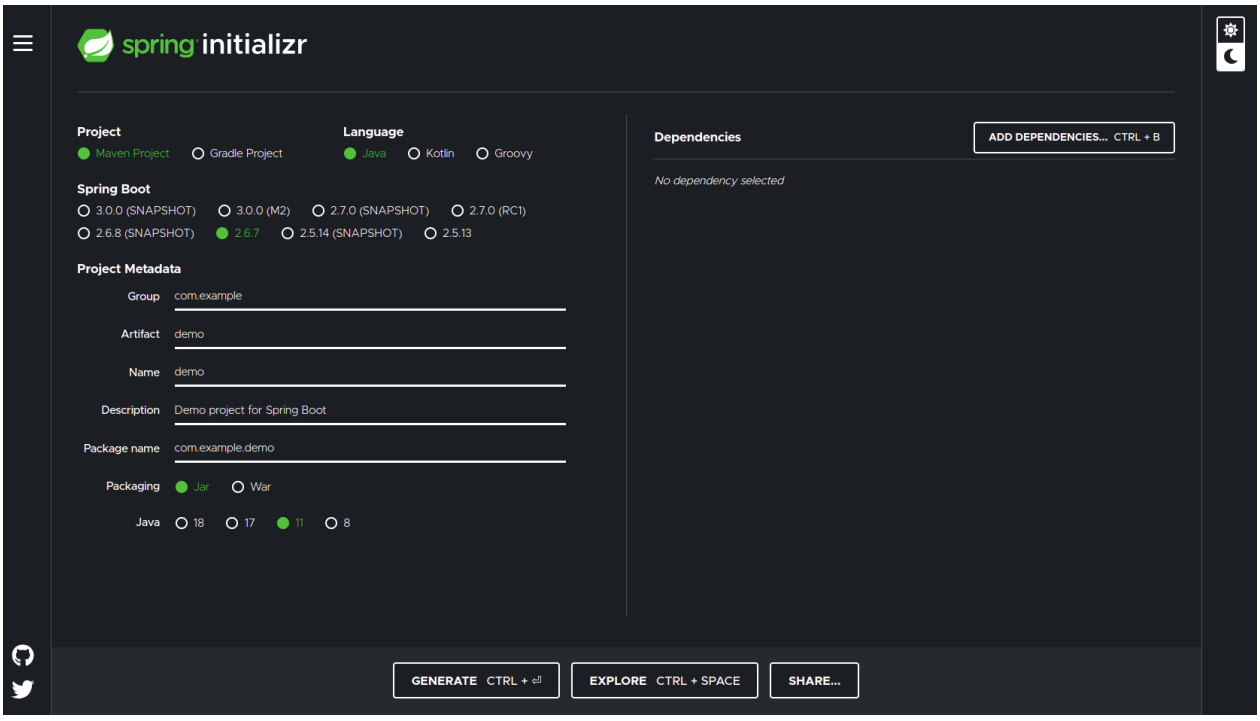
4. Software Development Life Cycle

Software development life cycle is a systematic process for building software that ensures the quality and correctness of the software built. SDLC process aims to produce high-quality software that meets customer expectations. The system development should be complete in the pre-defined time frame and cost. SDLC consists of a detailed plan which explains how to plan, build, and maintain specific software. Every phase of the SDLC life cycle has its own process and deliverables that feed into the next phase.

4.1 Installation

4.1.1 SpringBoot

Start Project.
Visit (https://start.spring.io/) → Set requirements → Download Project.



Figure(2)- SpringBoot Project download

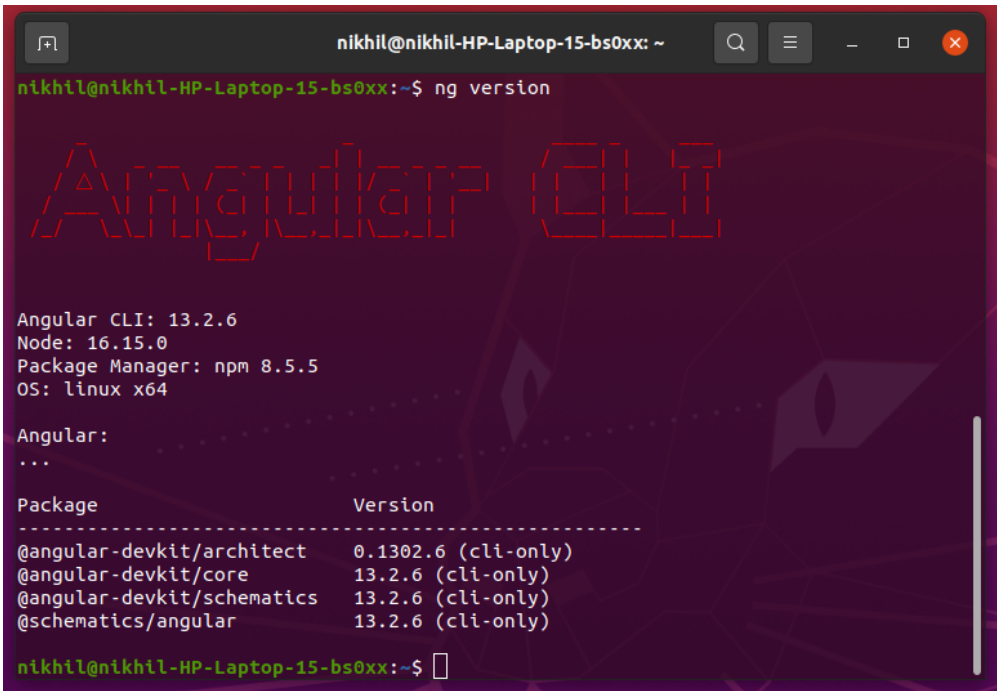
4.1.2 Angular

Angular is written in TypeScript. We need Node and npm to compile the files into JavaScript after that, we can deploy our application. For this purpose, Node.js must be installed in the system.

To check → (node - -version)

npm is used to install Angular 8 CLI. Once Node.js is installed, npm is also installed.

npm install -g @angular/cli



Figure(3)- Angular Installation

4.1.3 MySQL

MySQL is an open-source relational database management system.

Installing MySQL

- sudo apt update
- sudo apt install mysql-server
- sudo mysql_secure_installation

Configuring MySQL

- sudo mysql_secure_installation

This will take you through a progression of prompts where you can roll out certain improvements to your MySQL establishment security alternatives.

4.1.4 Docker

Docker is a continuous delivery tool. In a business enterprise, requirements keep changing and new updates keep waiting for deployment. The process of deployment has various prerequisite steps that need to be followed.
After installing some prerequisite packages, we can install docker.

Installing Docker

- apt install docker-ce
- usermod -aG docker ${USER}
- su - ${USER}

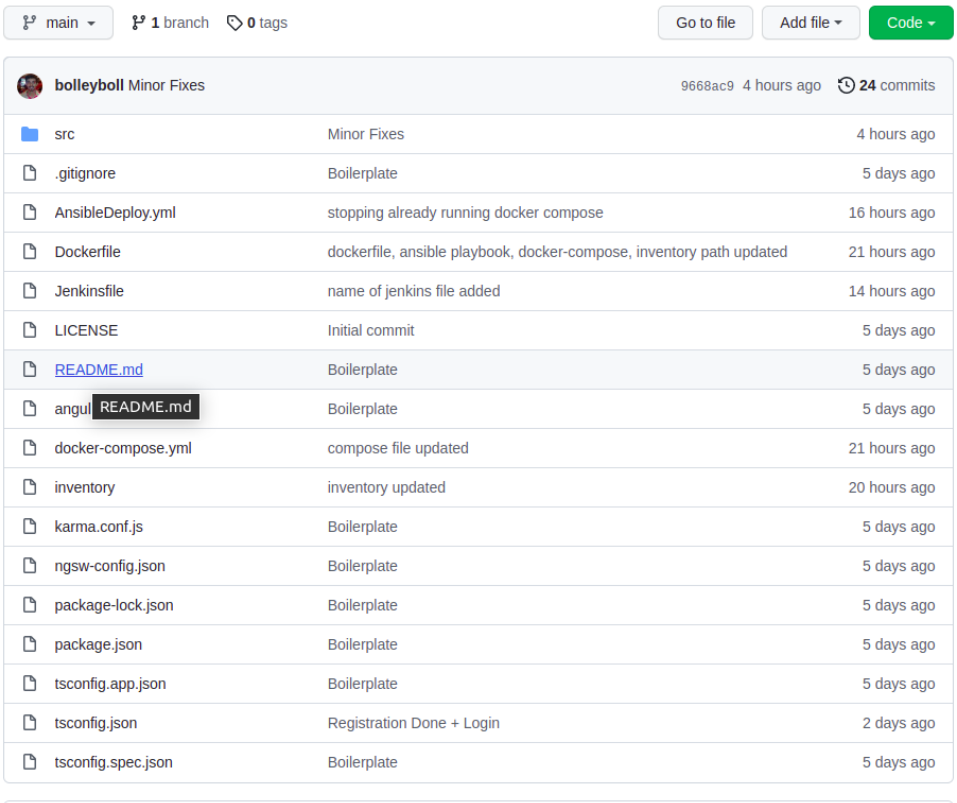
4.2 Source Control Management

We have used Git for source code management. A git repository is created on the Github with the team members as the collaborators. Hence, any of the members can push the changed code to the repository. Github helped us to work in a synchronous manner so we can change or access the code at the same time. IntelliJ provides VCS (version control system) to handle git commands and connects the local repository to the remote Github repository.

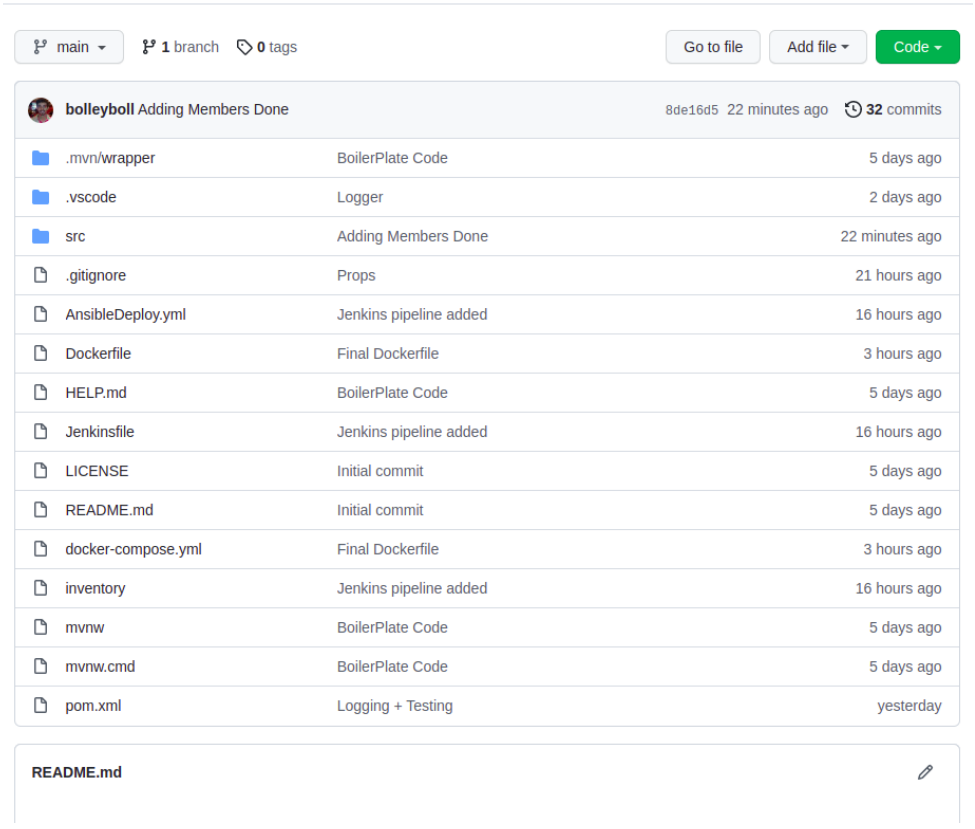
1. git init :- This command is used to create a new blank repository. It is used to make an existing project as a Git project.
2. git clone <url>:- This command is used to clone the remote repository to local system.
3. git add :- The git add command is used to add file contents to the Index(Staging Area) .This command updates the current content of the working tree to the staging area.
4. git commit -m <commit message>:- This creates a snapshot of the files and commits them by creating a SHA id.
5. git remote add origin <github-repository-url>:- This creates a new remote called origin located at given url.
6. git push -u origin <branch>:- This command is used for pushing the local changes to repository on server on specified branch.

Github repository links for the project:-

1. https://github.com/bolleyboll/Splitwise-API.git
2. https://github.com/bolleyboll/Splitwise.git



Figure(4)- Github repository for backend

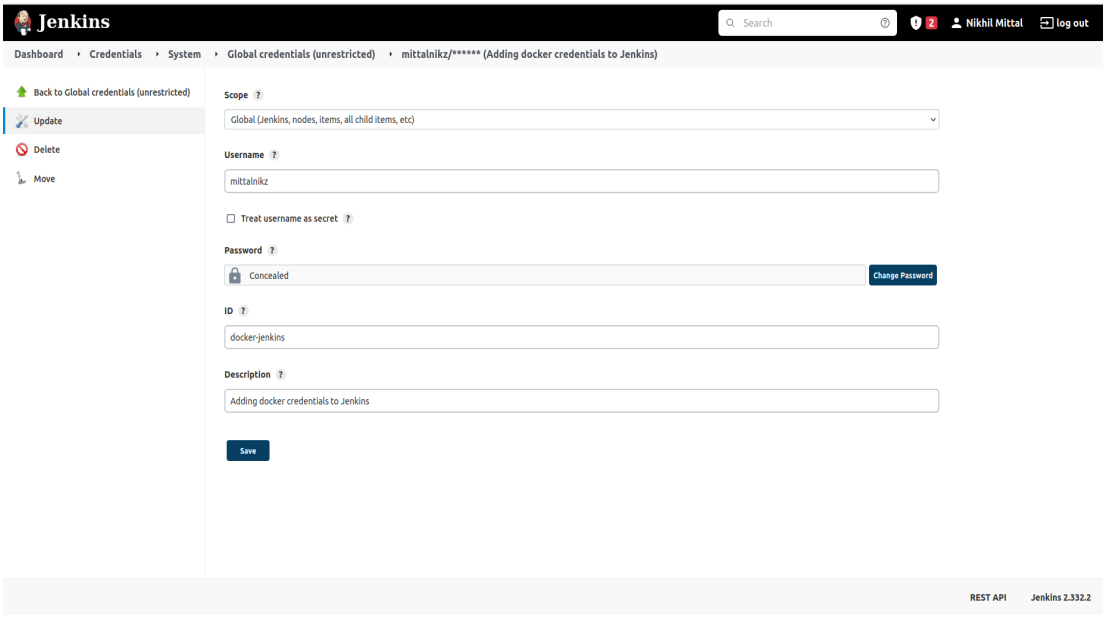


Figure(5)- Github repository for backend

4.2.1 CI Jenkins Pipeline:-

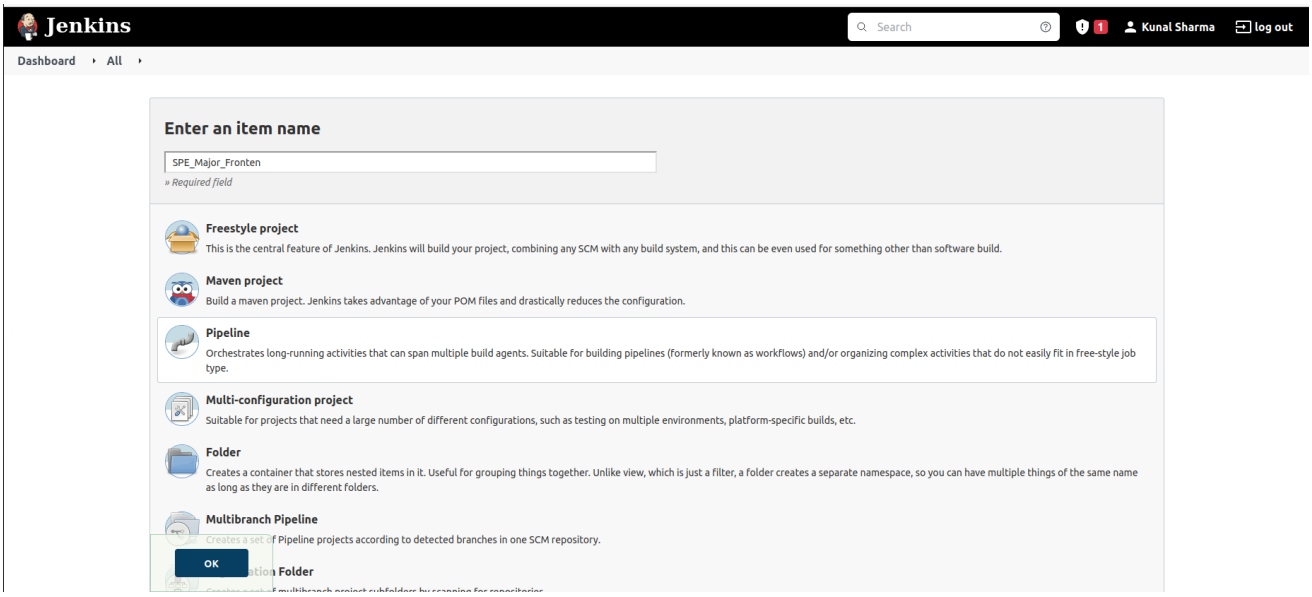
We used Jenkins Pipeline to clone these repositories from git.

➢ First we add credentials of Github repository



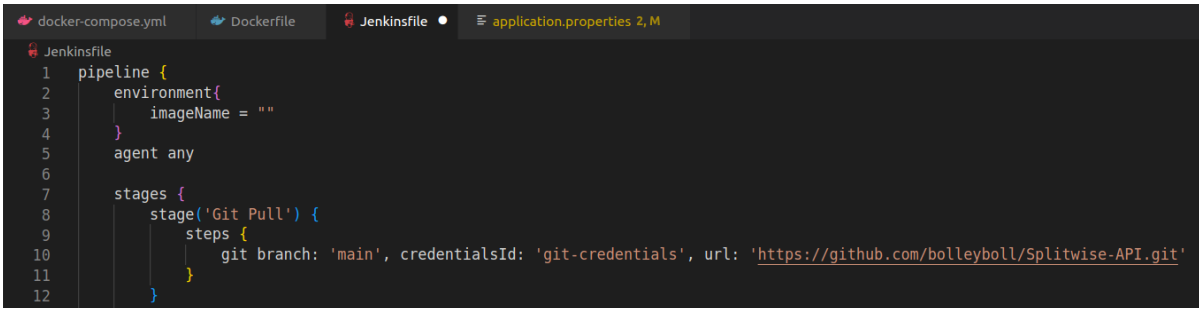
Figure(6)- Adding Github credentials

➢ Then we create a Jenkins pipeline.



Figure(7)- Jenkins pipeline.

➢ Git Script with jenkins pipeline



Figure(8)- Pipeline Script

4.3 Build

4.3.1 Maven

Apache Maven is a software project management and comprehension tool. Based on the concept of a project object model (POM), Maven can manage a project's build, reporting, and documentation from a central piece of information. It handles dependencies and plugins so we don't need to add jar files for the library explicitly. First, it checks for project dependencies in pom.xml and then compiles the code for any errors.

We have to install maven in our system so we can build what we write. To install Maven we want to first install its dependency JAVA, here we will be installing java version 8.

sudo apt update
sudo apt install openjdk-8-jdk

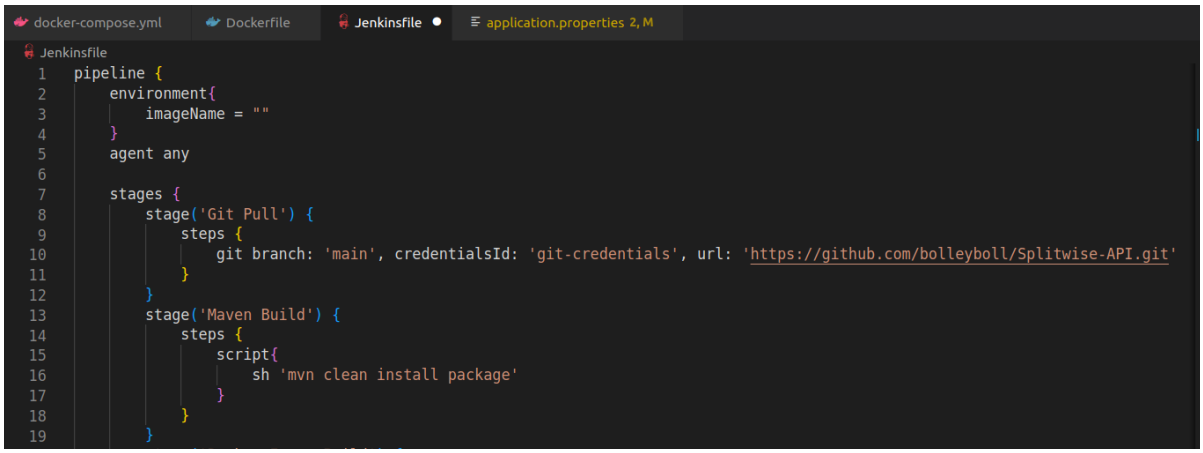
We will now install maven on our setup

sudo apt update
sudo apt install maven

4.3.2 CI Pipeline

Jenkins uses the system's maven to build the maven project, so we must have the maven integration plugin installed on our Jenkins.

- Install Maven integration plugin
- We add another stage to our pipeline.



Figure(9)- Maven build script

- In mvn clean install, it also runs all the test cases for the project and creates the executable jar file in the target folder in the current working directory.

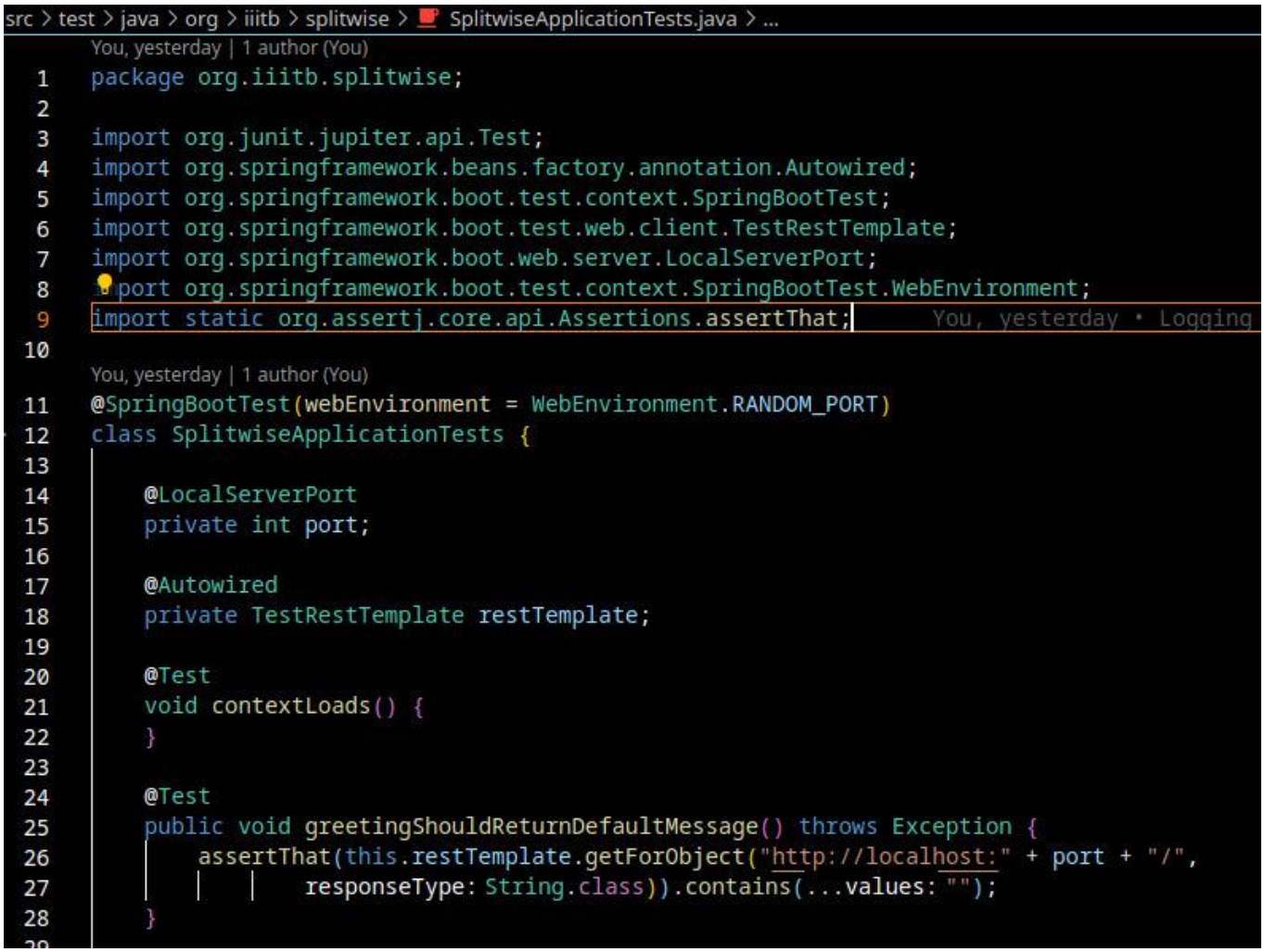
4.4 Testing

Jenkins provides us with continuous integration which includes integrated testing, so every time we push code to github it integrates all the different modules of the project and tests their proper functioning.

JUnit is a simple, open source framework to write and run repeatable tests. It is an instance of the xUnit architecture for unit testing frameworks.

JUnit promotes the idea of "first testing then coding", which emphasises on setting up the test data for a piece of code that can be tested first and then implemented.

This approach is like "test a little, code a little, test a little, code a little." It increases the productivity of the programmer and the stability of program code, which in turn reduces the stress on the programmer and the time spent on debugging.



Figure(10)- Test Cases

For testing within java code we use Junit. Junit test cases are written in covering the scope of the project. Also Junit dependency is added in pom.xml for maven to resolve the dependency.

4.5 Docker Artifact

Docker is an application that makes it simple and easy to run application processes in a container, which are like virtual machines, only more portable, more resource-friendly, and more dependent on the host operating system.

Artifact repository tools store all of the bulk binary artifact files that rarely, if ever, need to be altered or changed. Docker offers a tool like this.

First, we need to install docker on our system. Once it is installed, we can run it using following command:

$ sudo systemctl start docker

Here, our aim is to build a docker image out of the jar file that will be created in build phase and push the latest image to Docker Hub, which will then be pulled by Ansible to deploy it to other machines. All this will be automated in Jenkins pipeline. To push the docker image, use the following command:

docker push <username>/<repository_name>:tagname

To pull the docker image, use the following command:

docker pull <docker_image_name:tag>

To run the docker image, use the following command:

docker run –it <image_name>

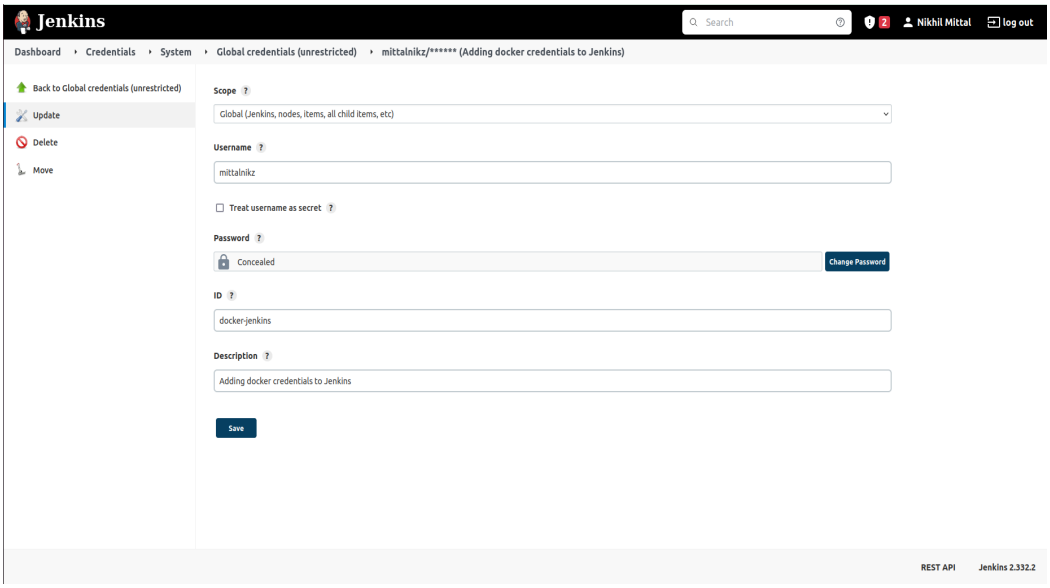
Also,make sure to add Jenkins to docker group, so that Jenkins can use docker for build docker image. This can be done by following command:

$ sudo usermod -aG docker Jenkins

Docker hub links for the project:-

1. https://hub.docker.com/r/patro30/frontend

2. https://hub.docker.com/r/patro30/backend



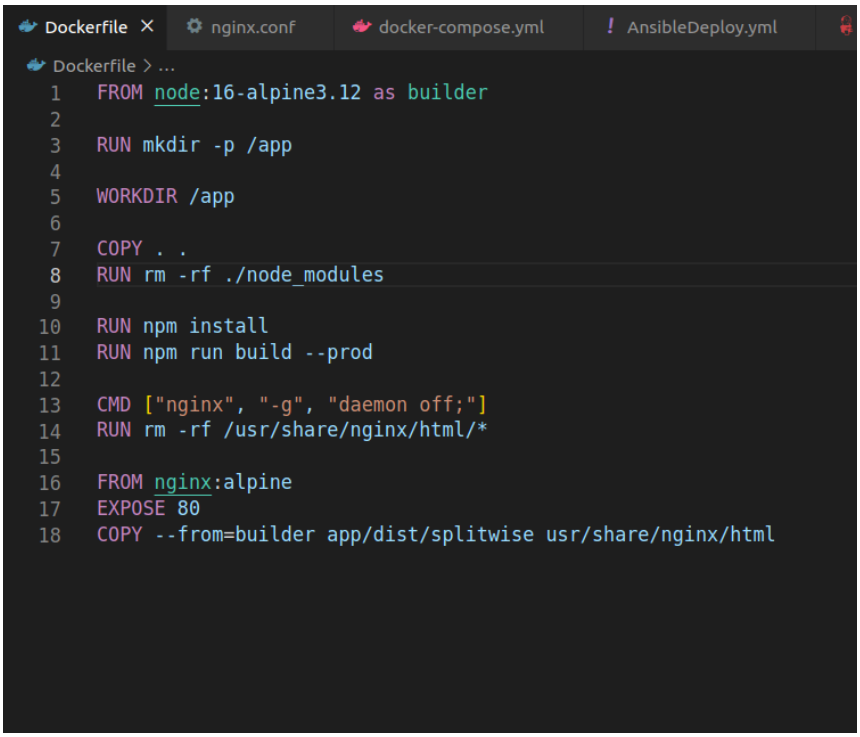
Figure(11)- Adding Docker credentials

4.5.1 CI Pipeline

- ➢ Add docker integration plugin for docker in Jenkins under Jenkins → Plugin Manager.
- ➢ Build and Push Spring Boot Image.

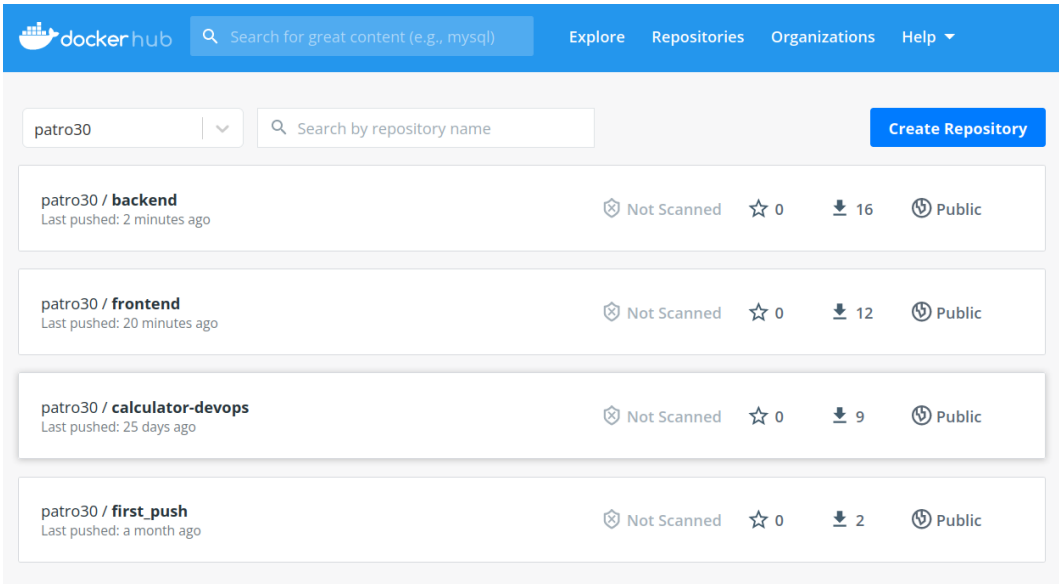
Figure(12)- Dockerfile for frontend

➢ Build and Push Angular Image.



Figure(13)- Dockerfile for backend

➢ After a successful deployment of docker images, we can see the result on our docker-hub account.



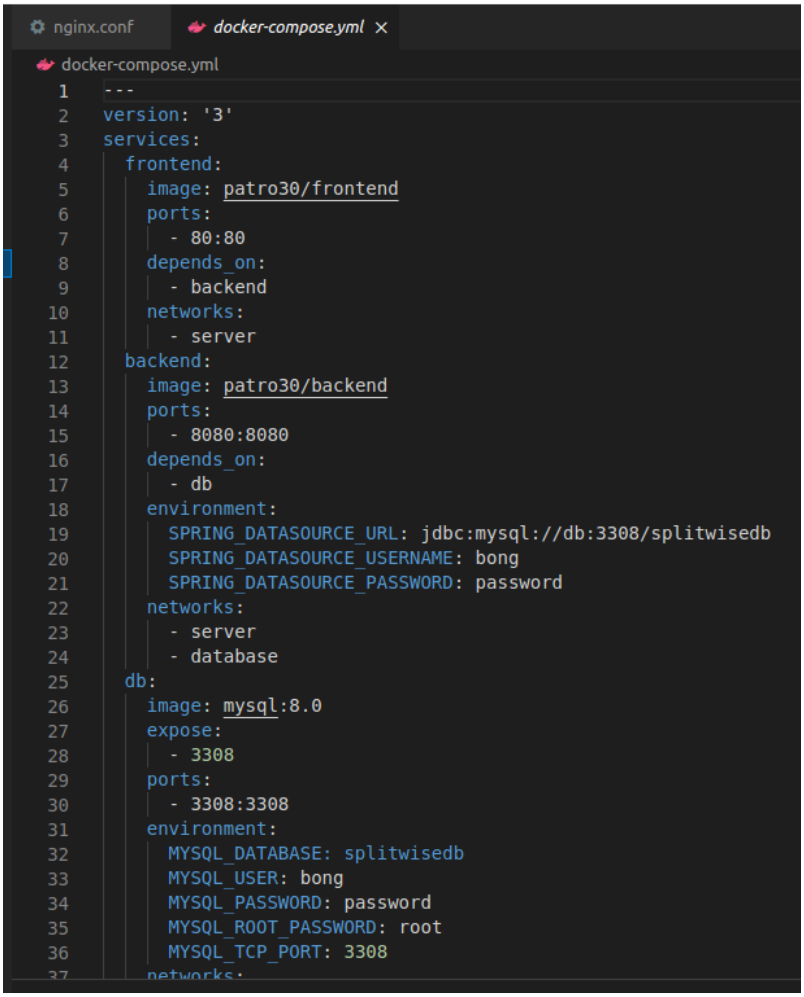
Figure(14)- DockerHub dashboard

4.5.2 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.



Figure(15)- Docker compose file

4.6 Continuous Deployment

A Deployment pipeline is the process of taking code from version control and making it readily available to users of your application in an automated fashion.
When a team of developers are working on projects or features they need a reliable and efficient way to build, test and deploy their work.

We are using Ansible for the deployment. It is a system of configuration management written in Python programming language which uses a declarative markup language to describe configurations. It's used for automation of configuration and OS setup. It is often used to manage Linux-nodes, but Windows is also supported.

Ansible allows us to write 'Playbooks' that are descriptions of the desired state of our systems, which are usually kept in source control. Ansible then does the hard work of getting your systems to that state no matter what state they are currently in. Playbooks make your installations, upgrades, and day-to-day management repeatable and reliable.

First, we need to install ansible on our system. Also, Ansible in the backend uses python to run various small codes and uses ssh in linux to connect to the hosts machines. So, we need to make sure that python(generally installed in linux) and ssh server are installed.

We can install them using following commands:

$ sudo apt install openssh-server
$ ssh-keygen -t rsa
$ sudo apt update
$ sudo apt install ansible

After the ansible is installed, we can verify it by checking its version using

$ ansible –version

4.6.1 CI Pipeline

➢ Jenkins would call Ansible to deploy the docker image on the deployment node.

```
36          stage('Ansible Pull Docker Image') {
37              steps {
38                  ansiblePlaybook becomeUser: null,
39                  colorized: true,
40                  disableHostKeyChecking: true,
41                  installation: 'Ansible',
42                  inventory: 'inventory.txt',
43                  playbook: 'ansible/playbook.yml',
44                  sudoUser: null
45              }
46          }
47      }
48  }
```

Figure(16)- Ansible pipeline script

➢ Ansible uses YAML syntax for expressing Ansible playbooks.

YAML uses simple key-value pairs to represent the data. The dictionary is represented in key: value pair.

docker-compose.yml | Dockerfile | Jenkinsfile | playbook.yml

ansible > playbook.yml

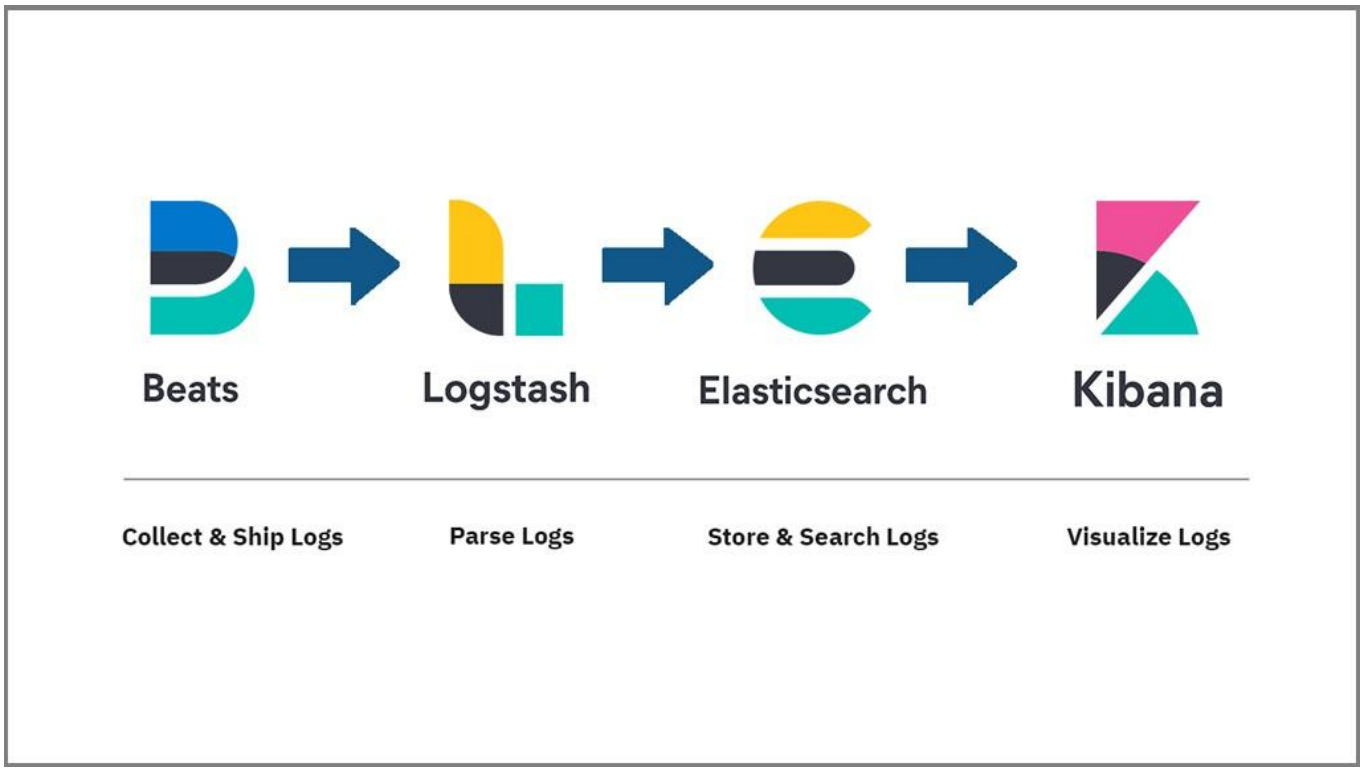
```
1  ---
2  - name: Deploy Splitwise app
3    hosts: all
4    tasks:
5      - name: Copy docker compose file to host
6        copy:
7          src: docker-compose.yml
8          dest: docker-compose.yml
9      - name: Remove already existing image
10       docker_image:
11         name: patro30/backend
12         state: absent
13         force: yes
14     - name: Pull updated image
15       docker_image:
16         name: patro30/backend
17     - name: Stop already running docker compose
18       shell: docker-compose down
19     - name: Run docker compose
20       shell: docker-compose up -d
```

Figure(17)- playbook file

4.7 Monitor

The final stage in the DevOps paradigm is monitoring.

In this step, developers keep track of user interaction and another usage of the application accordingly. The architecture used for monitoring is the ELK stack. "ELK" is the acronym for three open source projects: Elasticsearch, Logstash, and Kibana.

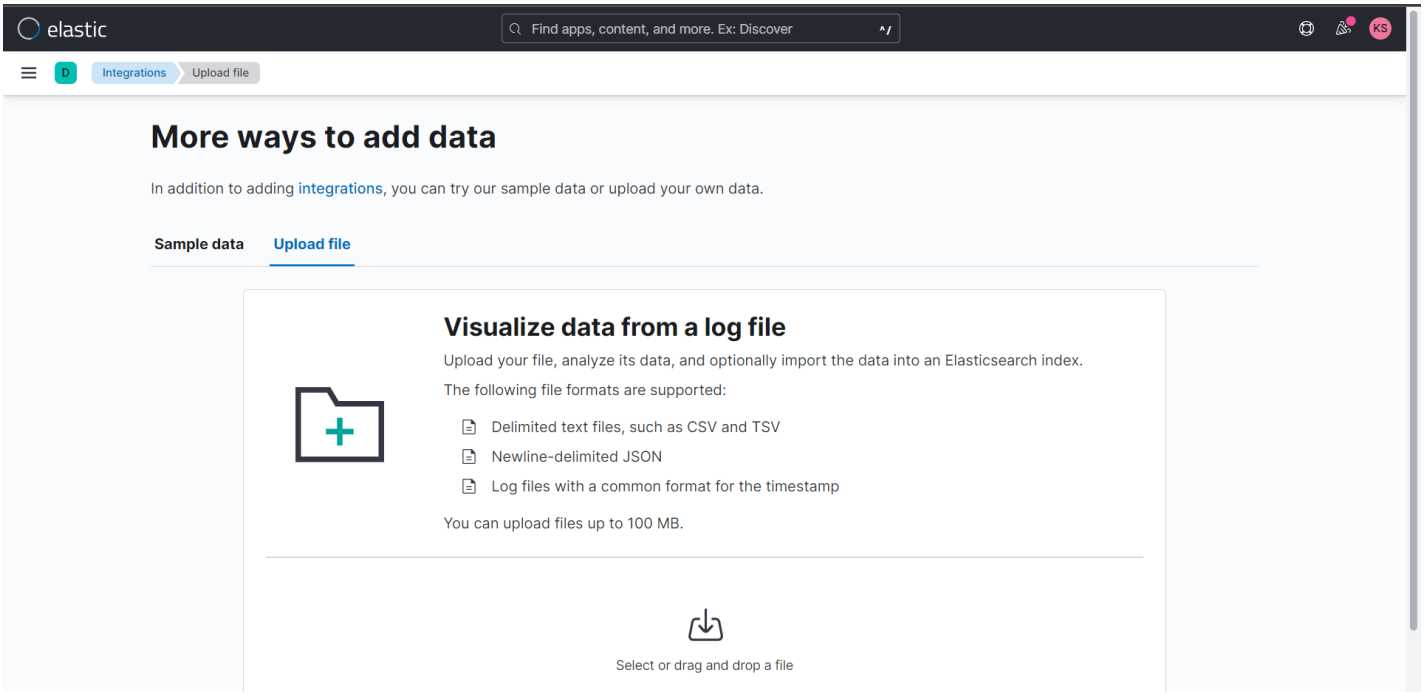


● **Elasticsearch** is a search and analytics engine.

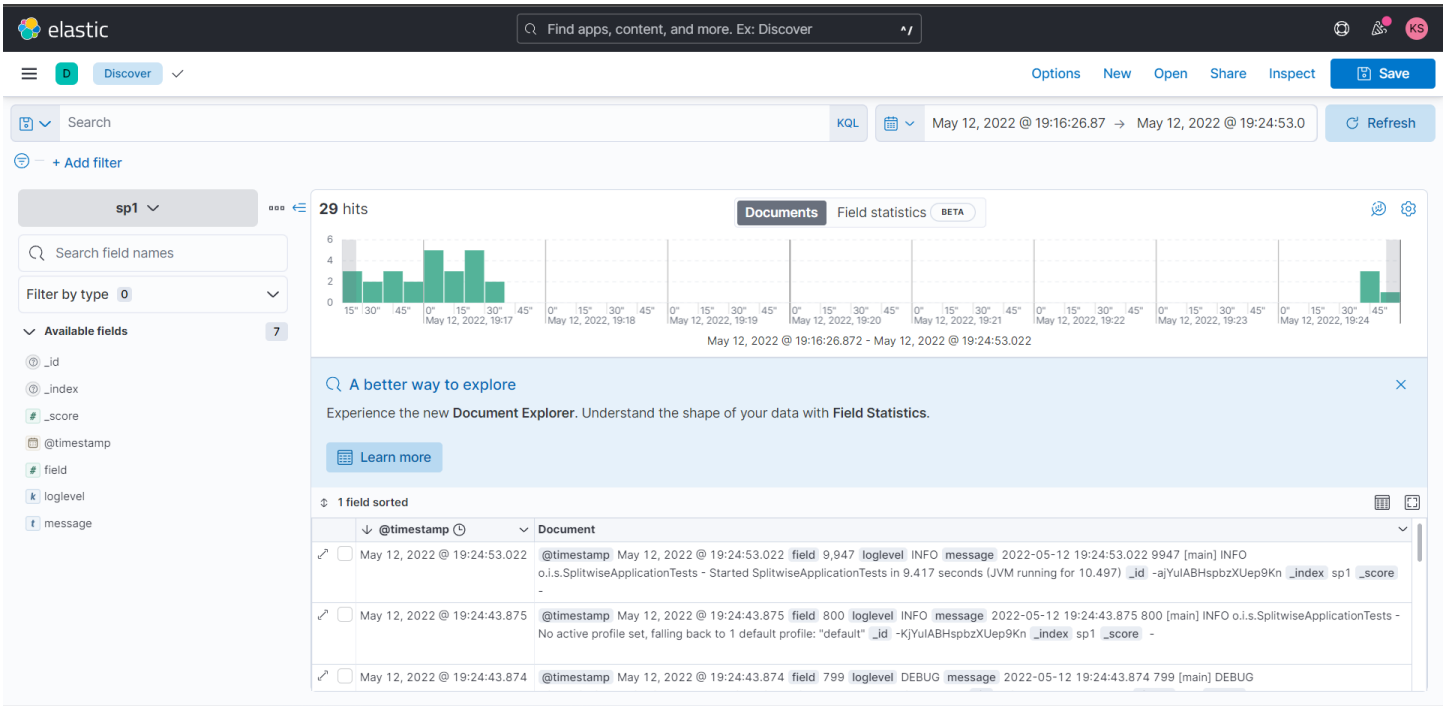
● **Logstash** is a server-side data processing pipeline that ingests data from multiple sources simultaneously, transforms it, and then sends it to a "stash" like Elasticsearch.

● **Kibana** lets users visualise data with charts and graphs in Elasticsearch.

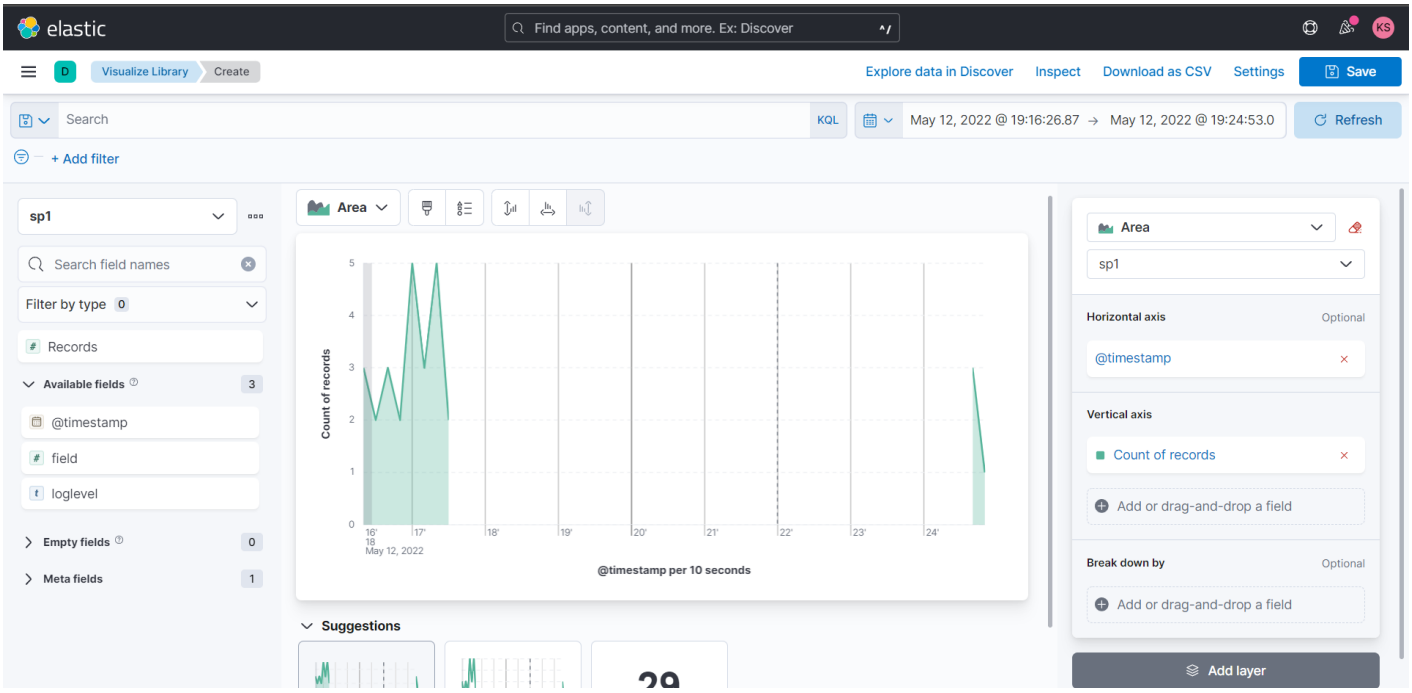
The ELK Stack is famous in light of the fact that it satisfies a need in the log, the executives and investigation space.



Figure(18)- uploading log file



Figure(19)- Kibana visualisation

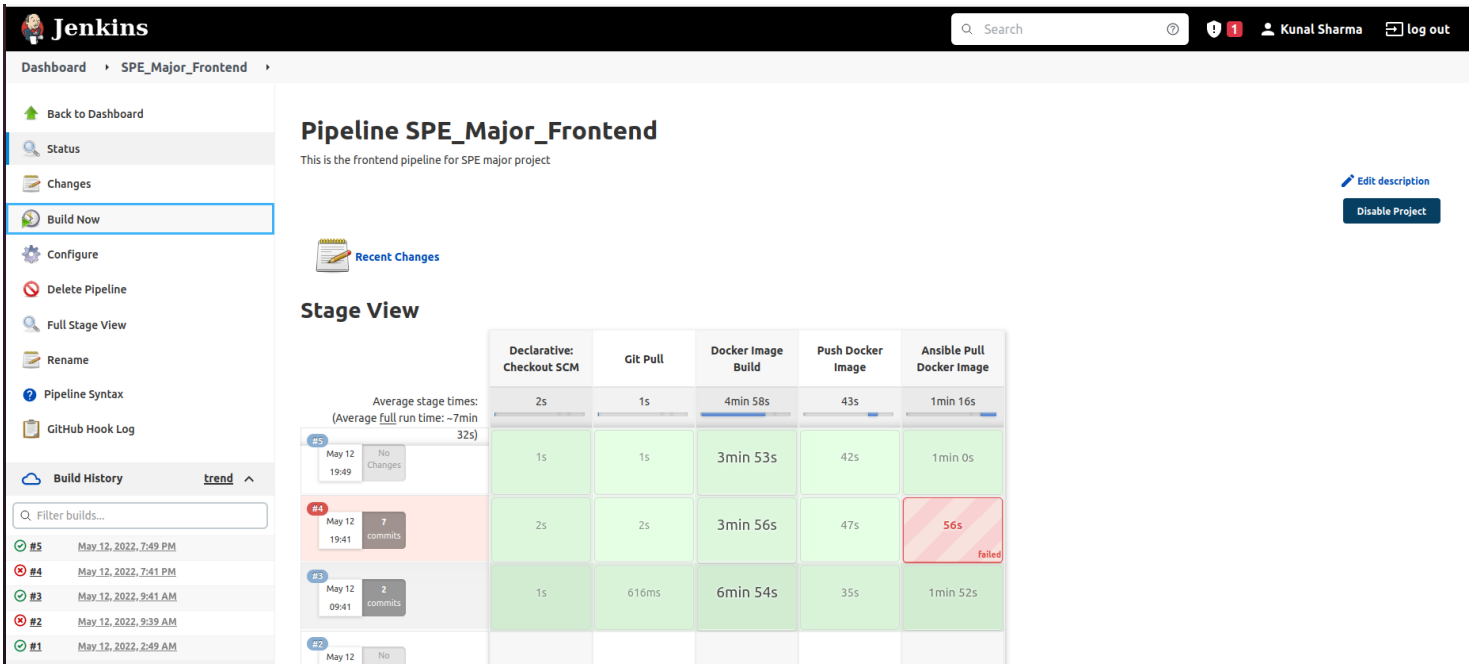


Figure(20)- Kibana visualisation

4.8 Building Pipeline and Running on Deployed Machine

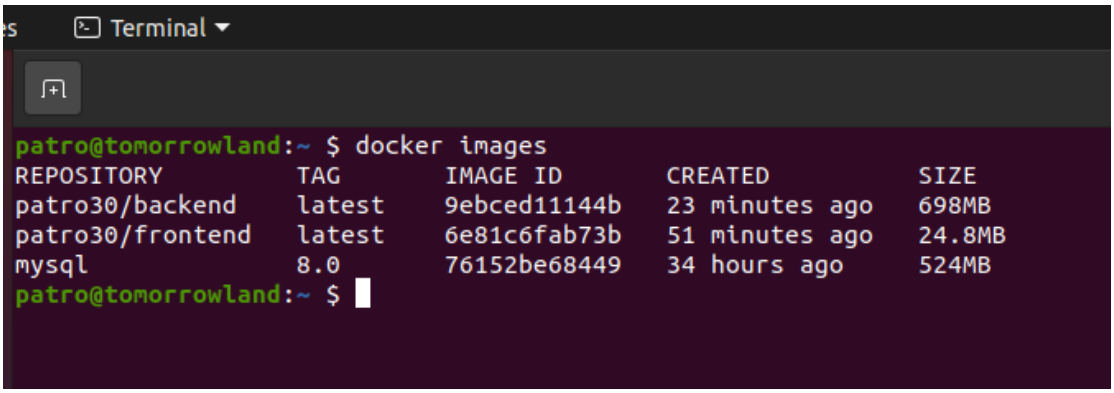
After completing all stages of SDLC we deployed the images successfully to the deployment node. We pulled the repo from git, build from maven , build and deploy the docker images and at last configured Ansible with Jenkins.

- Pipeline stages in Jenkins.



Figure(21)- Pipeline stages

- Deployed images on the target machine.



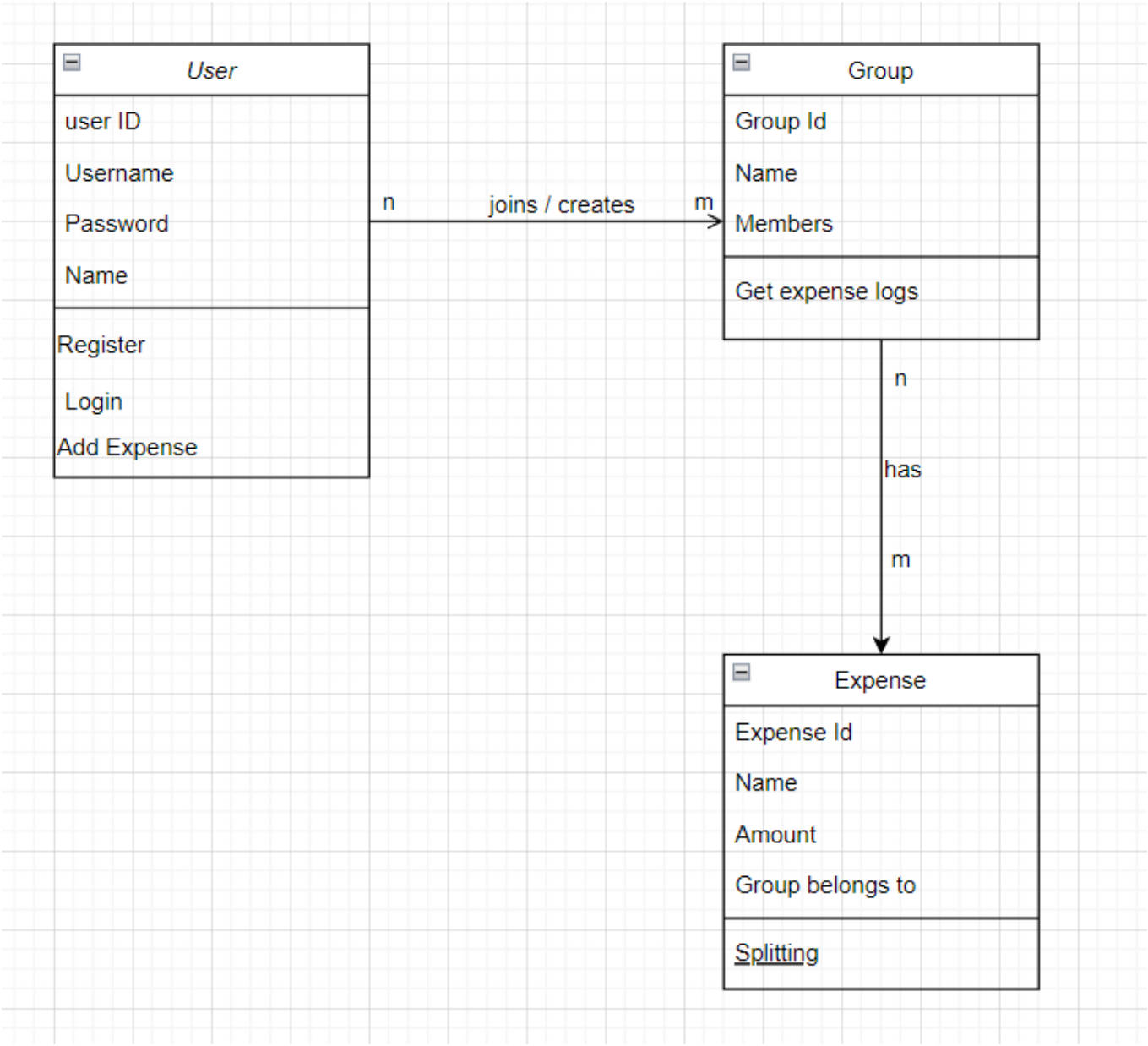
Figure(22)- Images on target machine

5. Experimental Setup

5.1 Functional Requirements

- Users can sign in into their account and can see all their previous expenses and bill splits.
- Users can create a particular group with friends and family and can divide expenses among the members of the group.
- Users can keep track of shared expenses, balances, and who owes who.
- Split expenses with any group: trips, housemates, friends, and family.
- Users can quickly add expenses on the go before you forget who paid.
- Settle up with a friend and record any cash or online payment.

5.2 Class Diagram



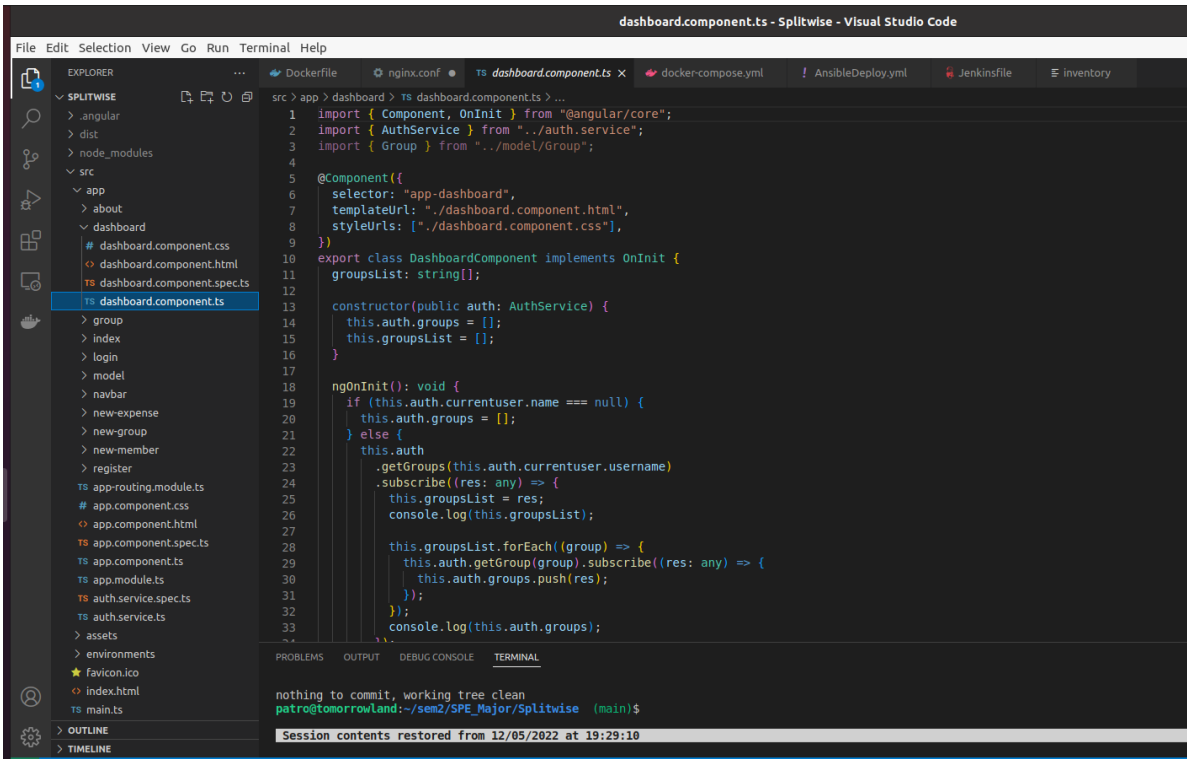
Figure(23)- Class diagram

5.3 API Documentation

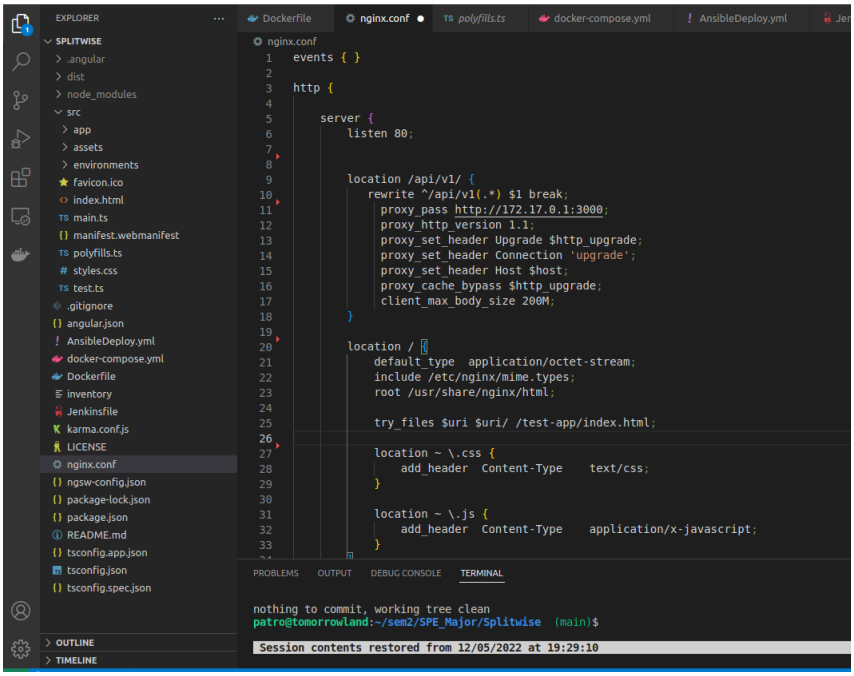
S.no	Use Case	API	Request Method	Role
1	User Registration	/register	POST	USER
2	User Login	/login	POST	USER
3	Get list of all the users	/users	GET	ADMIN
4	Adding the expense to the group	/expense/add	POST	USER
5	Get the group details	/group/{id}	GET	USER
6	Creating a group	/group/create	POST	USER
7	Get the expense details of particular group	/group/expenses/{id}	GET	USER
8	Adding another user to the group	/group/add/{username}/{grpId}	PUT	USER
9	Get all the groups for particular user	/user/group/{username}	GET	USER

5.4 Code Walkthrough

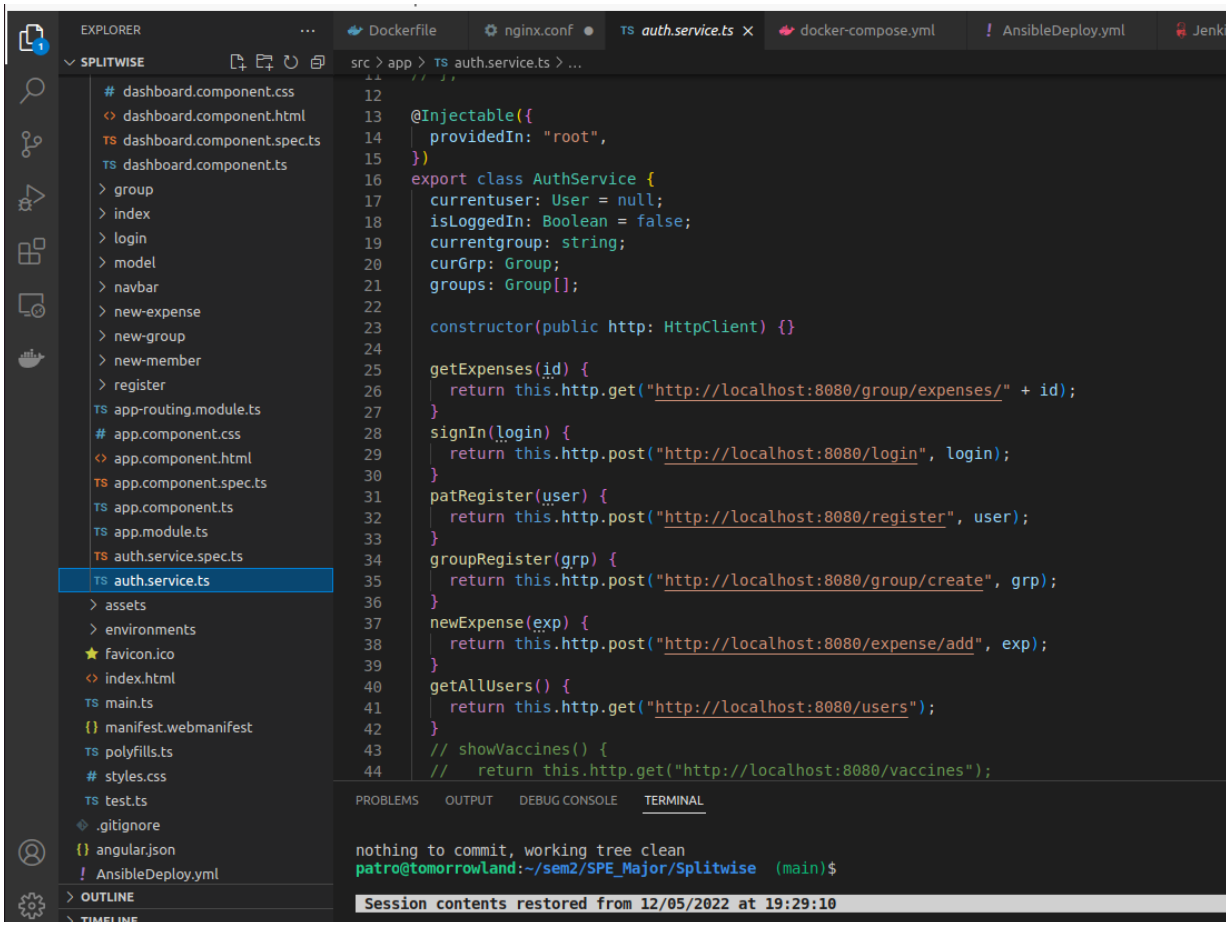
- Below code shows the modules that we have used in the front-end and components that we have created to drive our app.

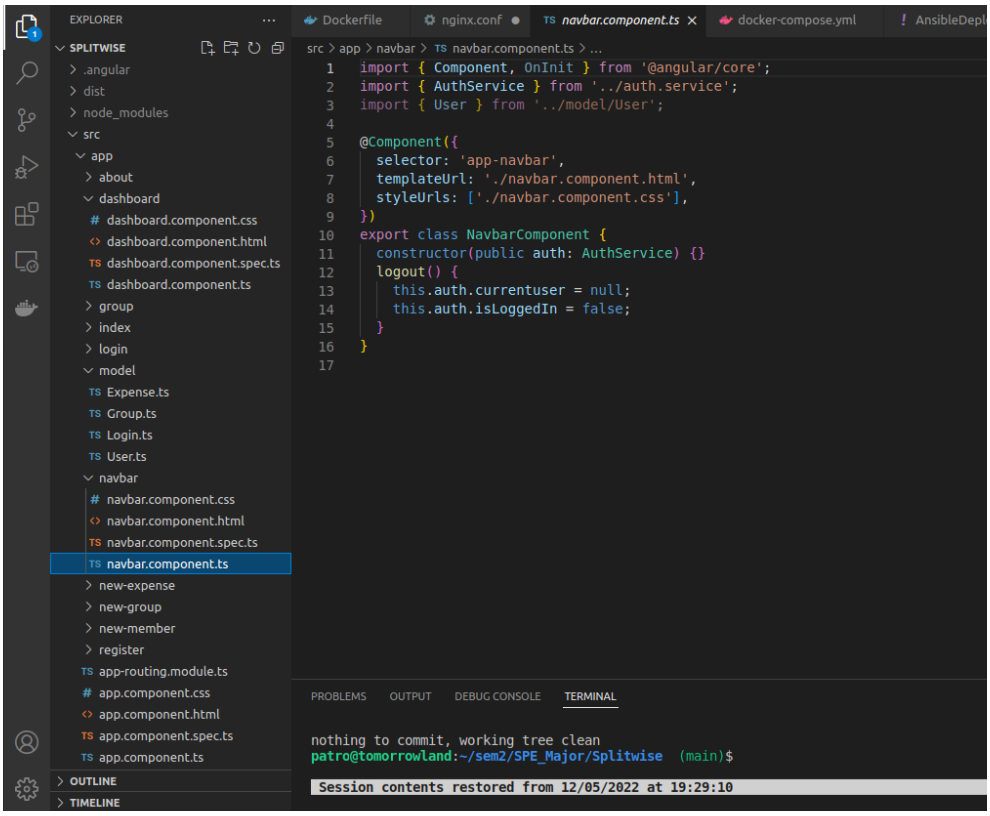


Figure(24)- Dashboard Components

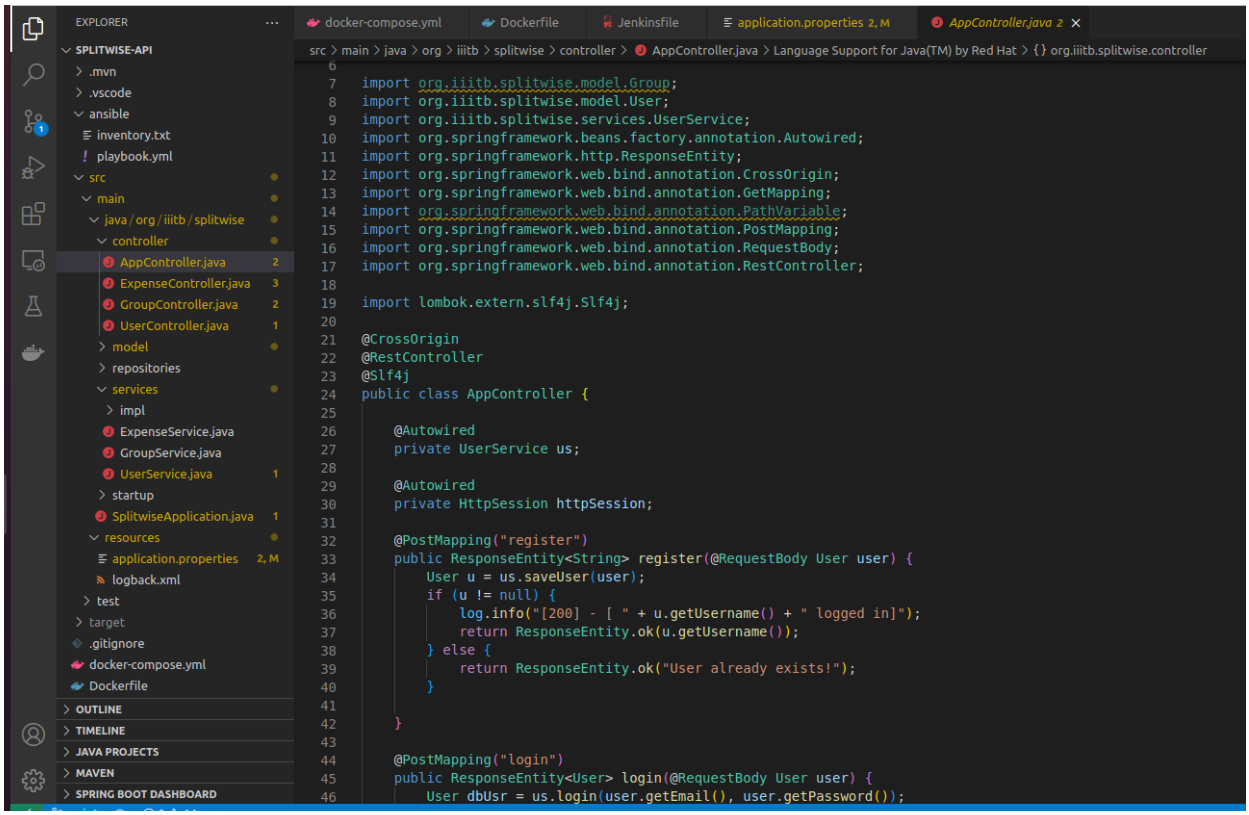


Figure(25)- nginx configuration

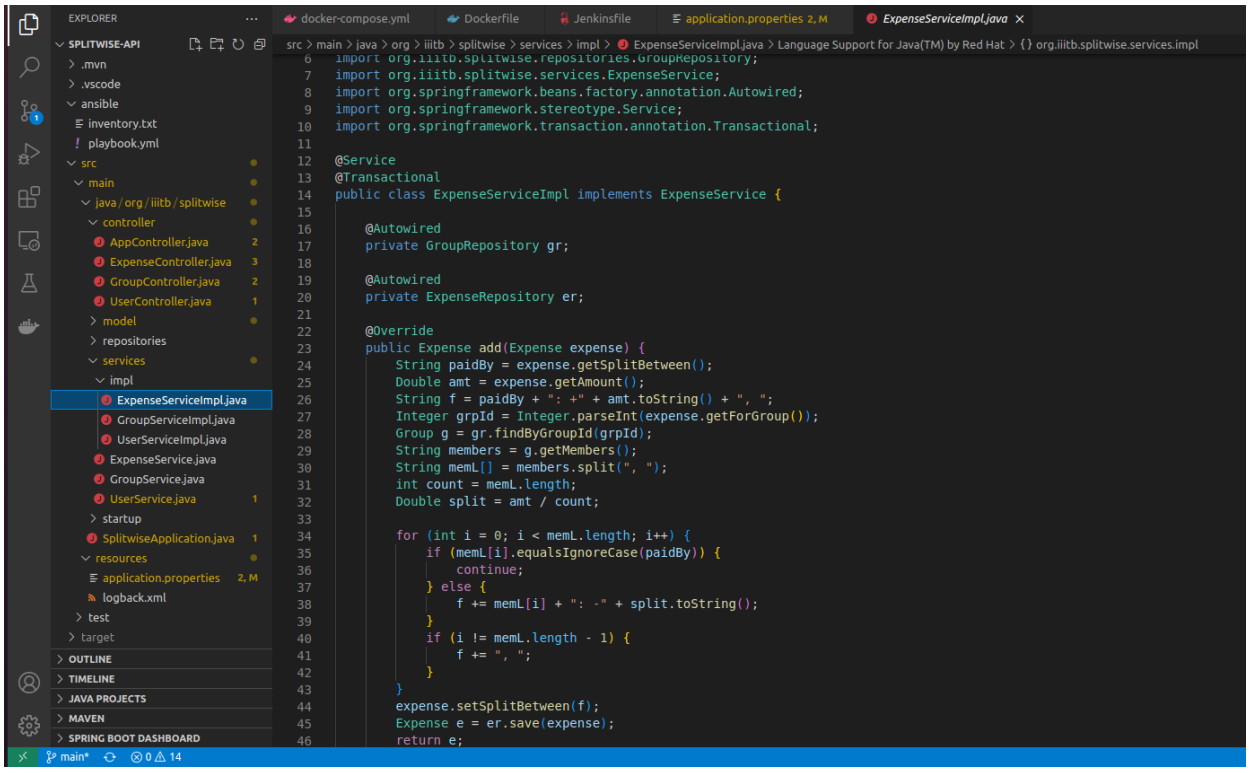




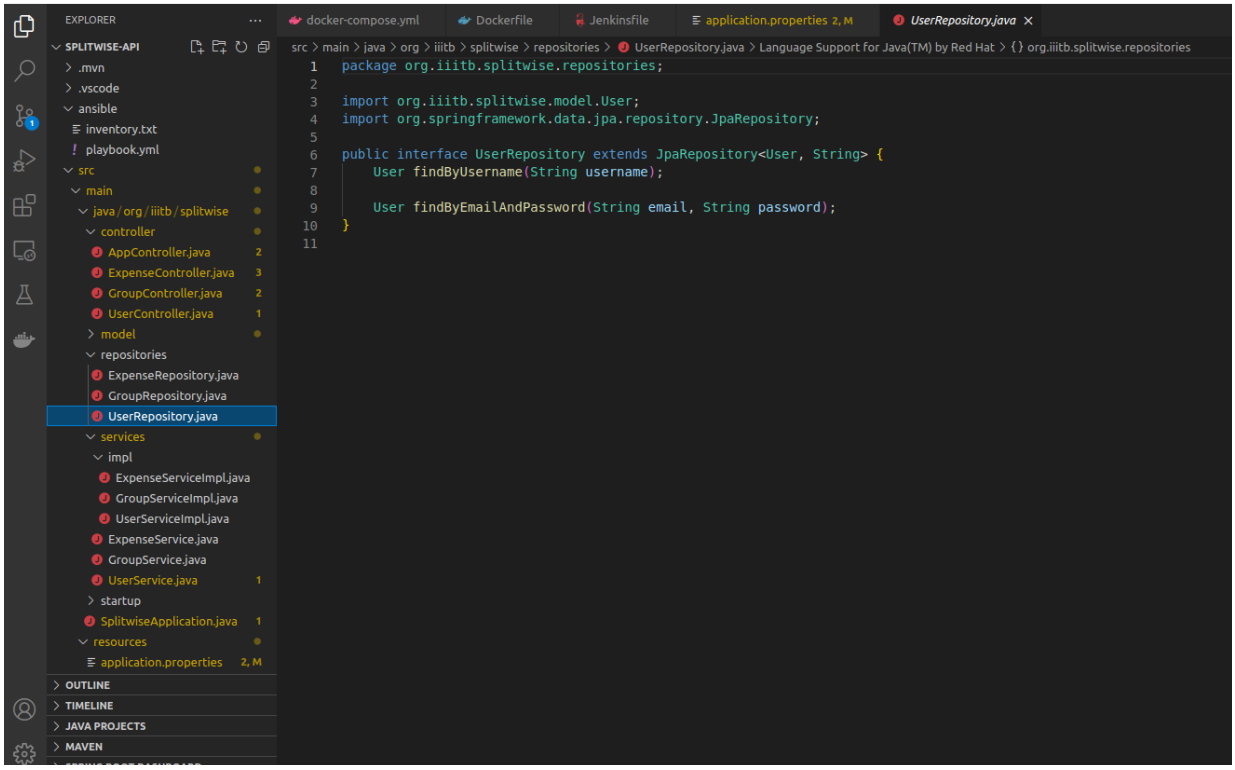
Figure(26)- Navbar component



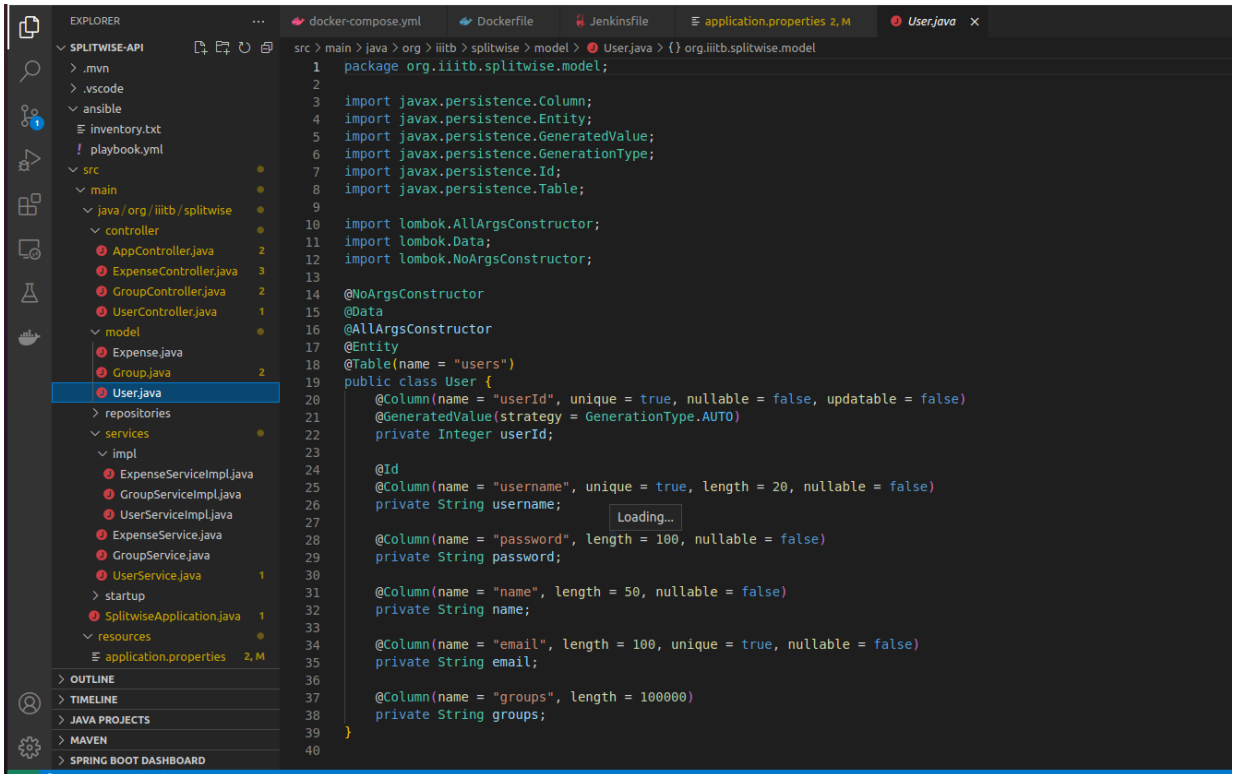
Figure(27)- App controller



Figure(27)- Expense Service



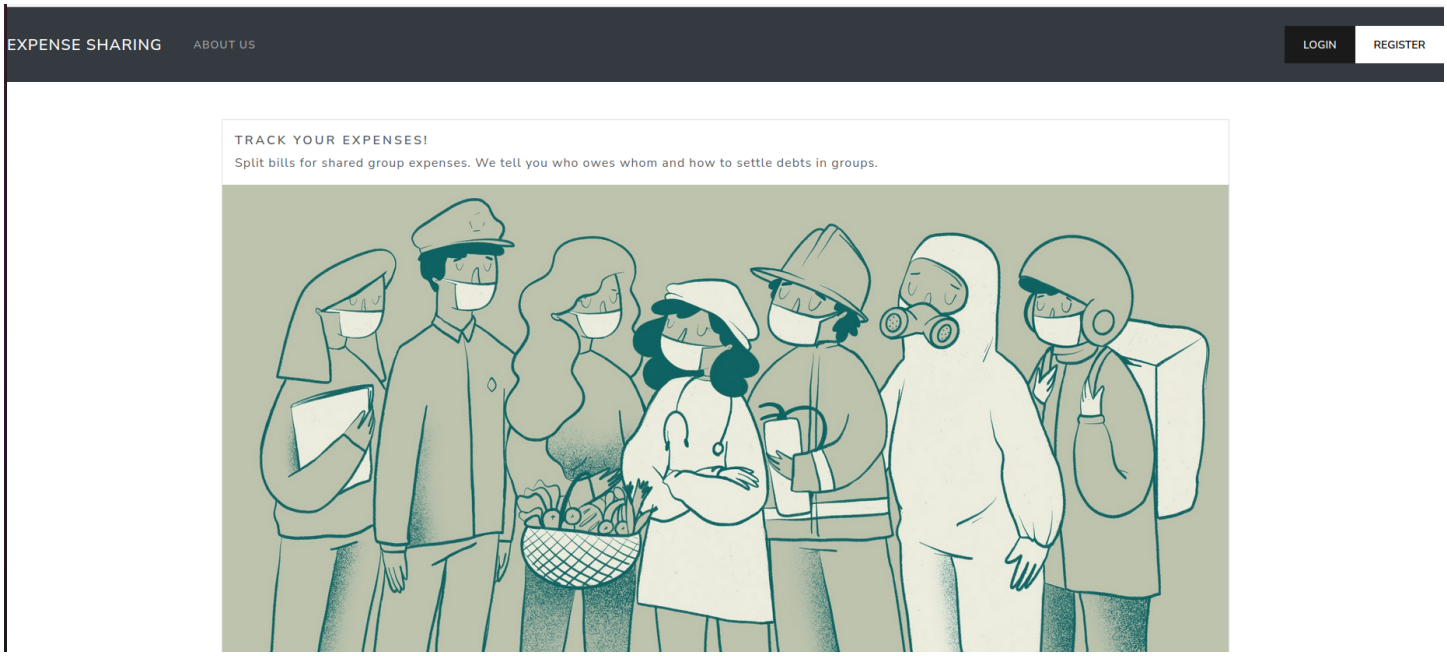
Figure(28)-User Repository



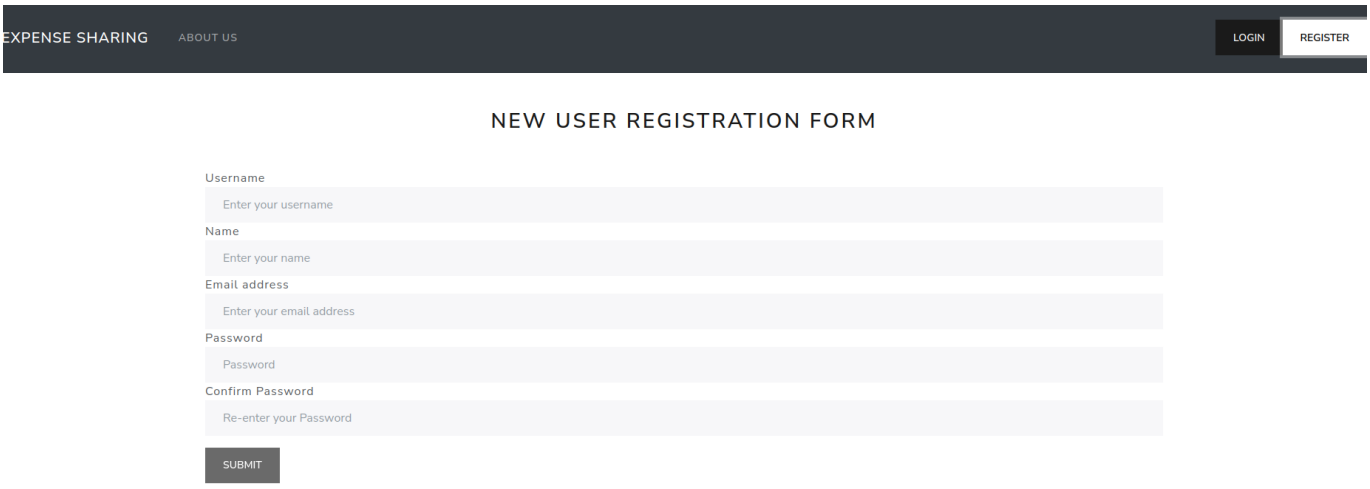
Figure(29)-User file

6. Results and Discussion

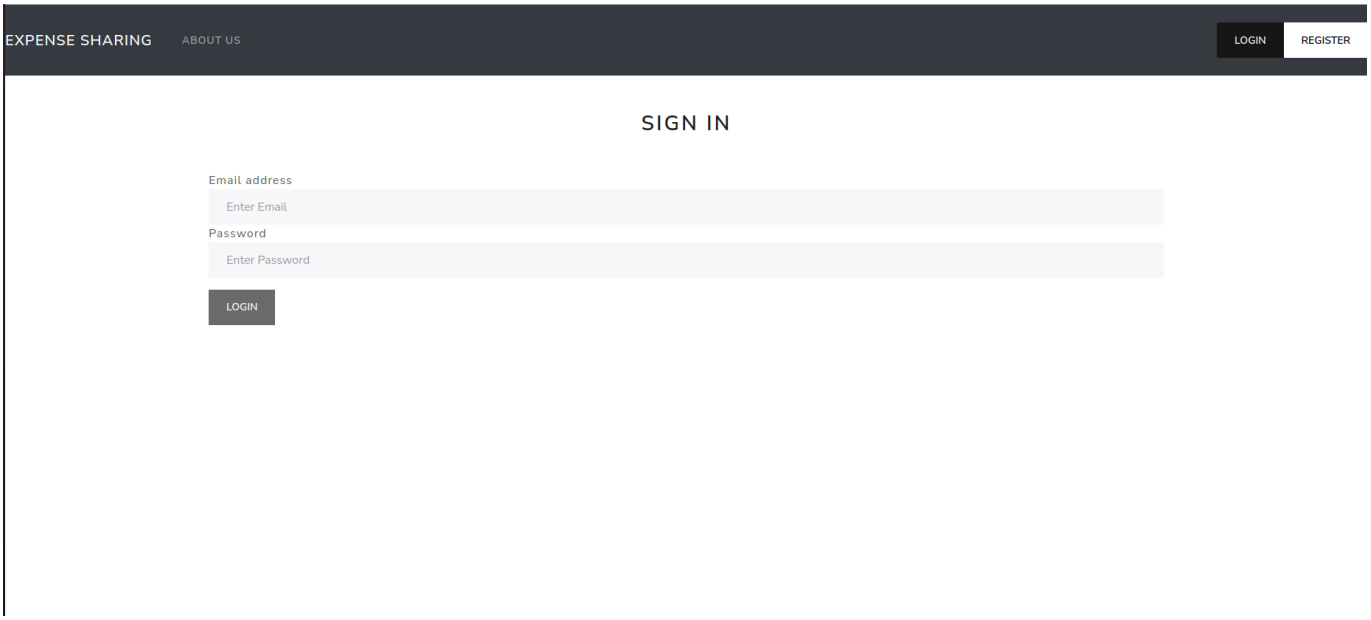
6.1 Home Page



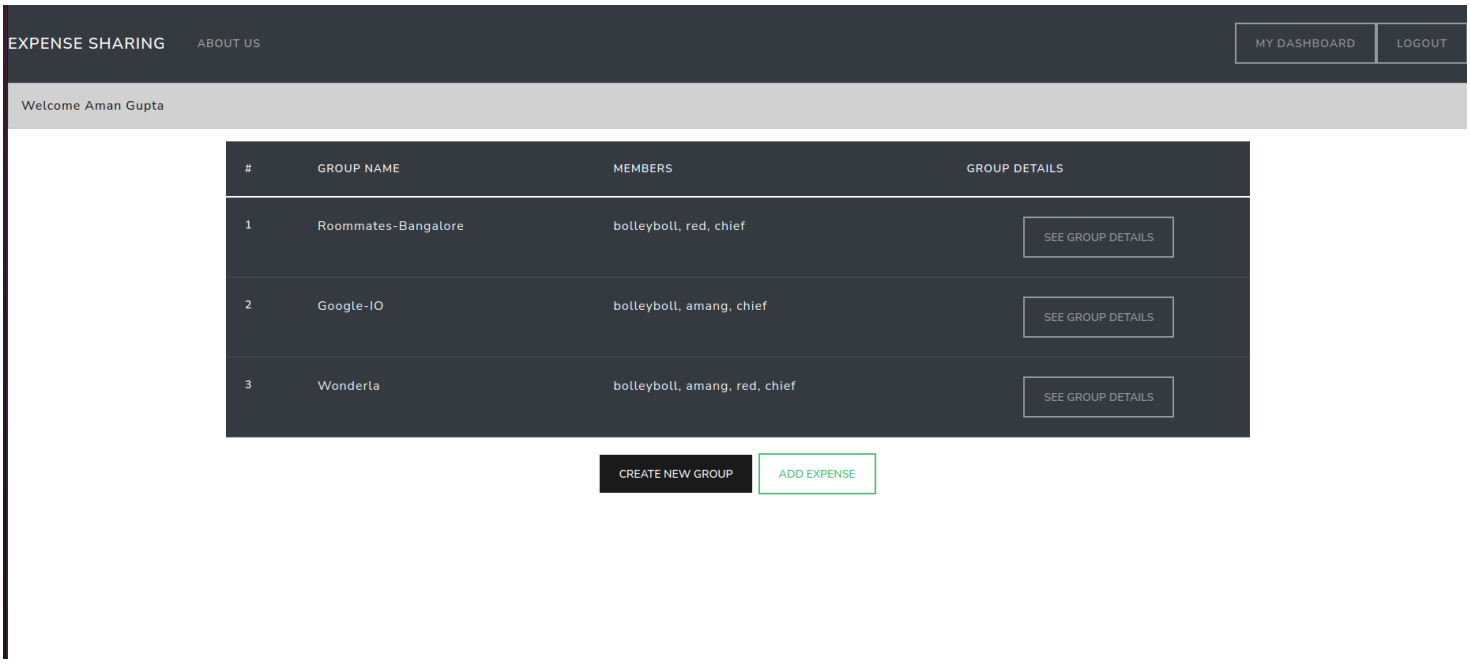
6.2 Sign Up Page



6.3 Sign In Page

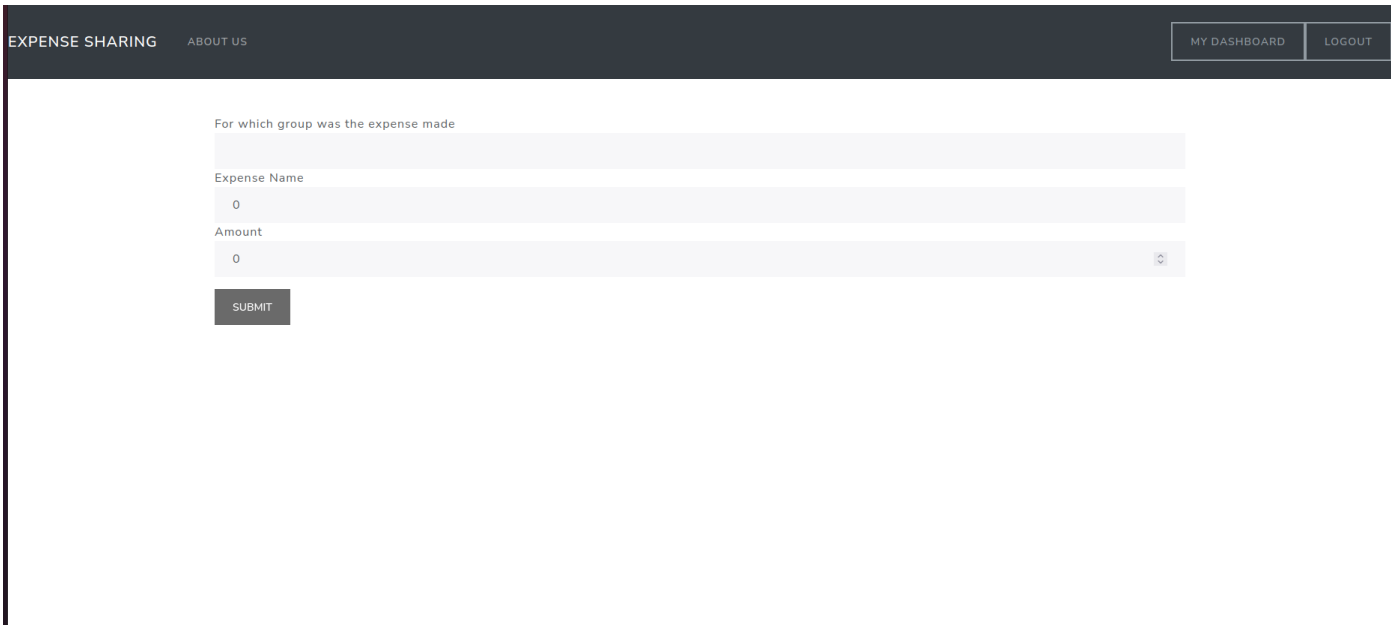


6.4 User Dashboard

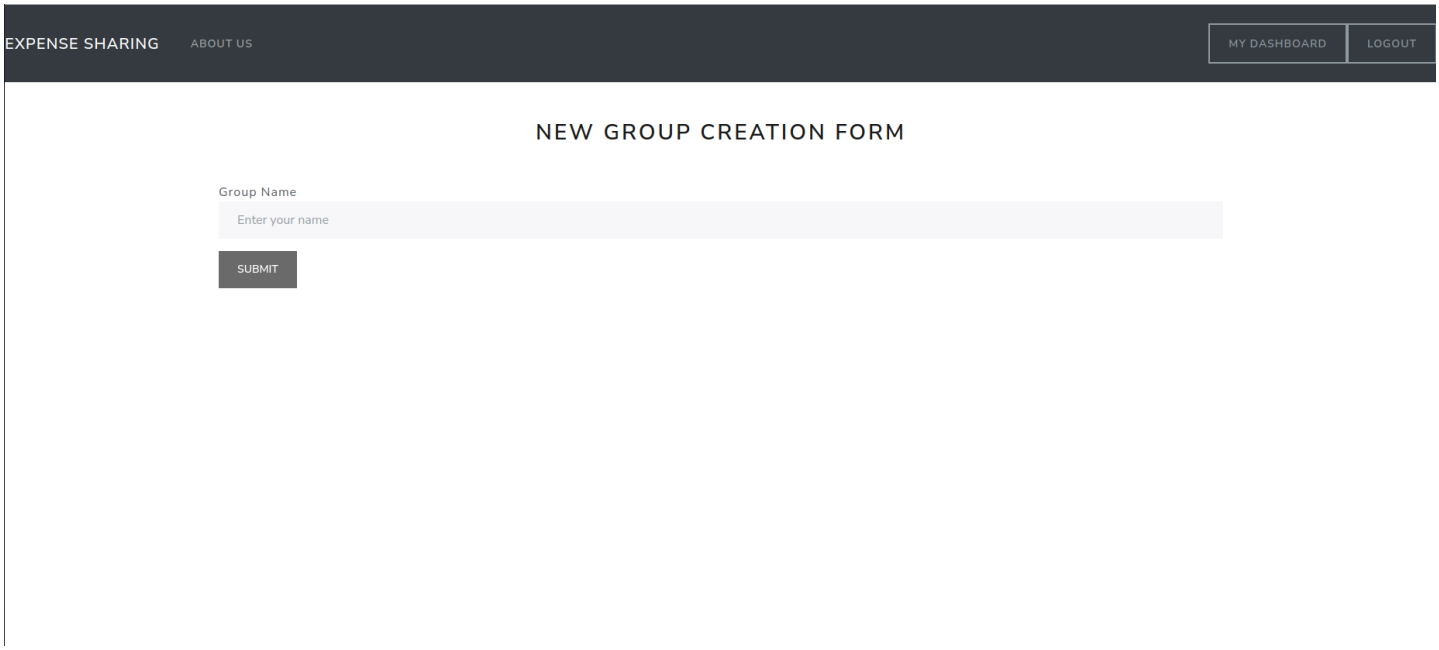


6.5 Group Details

6.6 Adding Expenses



6.7 Adding Group



7. Scope for Future Work

1. Payment Integrations

We can make this application integrate with payment merchants to enable users to settle the bill on the application itself.

2. Currency Conversion

One feature that can be added in this application is for users that live in different countries. We can enable the currency to be converted based on the current currency rates in the world.

3. Receipt scanning

This feature will enable the users to just scan the receipt/bill and the whole information will be reflected on the application that will reduce the manual work that has to be done by the users.

4. Multilingual

This application can be transformed into multi-language application thus increasing the accessibility of the users from different parts of the country.

8. Conclusion

We successfully built the web application Expense Sharing Application that can be used between friends and family to split the bills easily and conveniently . This application can prove to be very useful in day to day activities.

The project taught us a great deal about all the DevOps tools that we used for our projects. Through usage of these tools we understood the importance of automation in software development. We learnt about technologies like AngularJS which was used to make the front-end of our website, SpringBoot to make the back-end, SCM tool of Git, CI/CD tools like Jenkins, Deployment platform like Docker, and CM tool like ELK. These tools diminished our work impressively.

9. References

1. Angular SpringBoot JWT Authentication :-
https://loizenai.com/angular-10-spring-boot-jwt-authentication-example/#angular-spring-boot-jwt-authentication-example

2. Angular + SpringBoot + MySQL CRUD Tutorial from “B2 Tech” :-
https://youtube.com/playlist?list=PLA7e3zmT6XQXgBjcPyePwqgpEkI14MEXE

3. Angular + SpringBoot CRUD Full stack Application from “Java Guides” :-
https://youtube.com/playlist?list=PLGRDMO4rOGcNzi3CpBWsCdQSzbjdWWy-f

4. https://www.docker.com/

5.https://wkrzywiec.medium.com/build-and-run-angular-application-in-a-docker-container-b65dbbc50be8

6. https://www.ansible.com/overview/how-ansible-works