- STACK OVERFLOW CLONE PROJECT

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE AND ENGINEERING

Rajiv Gandhi University of Knowledge Technologies
R.K.VALLEY

Submitted by:-

M.Swapna (R171193)

Under the Esteemed guidance of Ms.S.Shabana

Department of Computer Science and Engineering

RGUKT RKValley

Rajiv Gandhi University of Knowledge Techonolgies

RK Valley, Kadapa (Dist), Andhra Pradesh, 516330

CERTIFICATE

This is to certify that the project work titled "**STACK OVERFLOW CLONE**" is a bonafied project work submitted by **M.Susmitha Chowdary and M.Swapna** in **COMPUTER SCIENCE AND ENGINEERING** in partial fulfillment of requirements for the award of degree of **Bachelor of Technology** for the year **2022-2023**carried out the work under the supervision.

INTERNAL GUIDE
(S SHABANA,Asst.Professor)

HEAD OF THE DEPARTMENT
(N SATYANANDARAM)

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whose constant guidance and encouragement crown all the efforts success.

I am extremely grateful to our respected Director,Prof. K. SANDHYA RANI for fostering an excellent academic climate in our institution.

I also express my sincere gratitude to our respected Head of the Department Mr SATYANANDARAM sir for his encouragement, overall guidance in viewing this project a good asset and effort in bringing out this project.

I would like to convey thanks to our guide at college S SHABANA for her guidance, encouragement, co-operation and kindness during the entire duration of the course and academics.

My sincere thanks to all the members who helped me directly and indirectly in the completion of project work. I express my profound gratitude to all our friends and family members for their encouragement.

INDEX

S.NO	INDEX	PAGE NUMBER
1	Abstract	5
2	Introduction	6
3	Purpose	6
4	Scope	6
5	Software and Hardware Requirements	7
6	About	8-10
7	Requirements and Specifications	11
8	Functional Requirements	12
9	Non-Functional Requirements	13
10	Design,UML Diagrams	14
11	Use Case Diagram	15
12	System Architecture,Class Diagram	16-17
13	Implementation and System Testing	18
14	Basic Folder Structure of Project	19-21
15	Creating Components	22
16	Code	23-25
17	MongoDB Atlas for Database	26-27
18	Code	28-29
19	Deployment	30
20	Screeenshots	31-34
21	Conclusion,References	35

ABSTRACT

When programmers look for how to achieve certain programming tasks, Stack Overflow is a popular destination in search engine results. In this project we will build a web-based stackOverflow Clone application using MERN stack.MERN stack expands
ReactJS,NodeJS,ExpressJS,MongoDb.Full stack technology refers to the entire depth of a computer system application,the front end and the back end.Express and Node make up the middle(application) tier.Express.js is a server-side web framework, and Node.js is the popular and powerful Javascript server platform.React.js is used for building interactive user interfaces and web applications quickly and efficiently with significantly less code.MongoDB , which is a scalable, flexible NoSQL document database platform designed to overcome the relational databases approach and the limitations of other NoSQL solutions ,will handle the user data.In this project we will make exact replica of the most popular website “STACK OVERFLOW” both front end and back end using ReactJs, NodeJs , ExpressJs and MongoDB.

Introduction

Stack Overflow is a question and answer website for programmers.It was created in 2008 by “Jeff Atwood” and “Joel Spolsky”.It features question and answers on certain computer programming topics.The website serves as a platform for users to ask and answer questions, and, through membership and active participation, to vote questions and answers up or down similar to Reddit and edit questions and answers in a fashion similar to a wiki.Users of Stack Overflow can earn “reputation points” and "badges".

As of March 2022 Stack Overflow has over 20 million registered users , and has received over 24 million questions and 35 million answers.Based on the type of tags assigned to the questions, the top eight most discussed topics on the sight are:JavaScript, Java, C#, PHP, Android, Python, Jquery and HTML.Stack Overflow is written in C# using the ASP.NET MVC (Model-View-Controller) framework, and Microsoft SQL Server for the database and the Dapper Object-relational-Model used for data access.

Purpose

The main purpose of Stack Overflow clone project is is to build a web-based Stack Overflow Clone application using MERN stack.MERN stack expands ReactJS, NodeJS, ExpressJS, MongoDb.Full stack technology refers to the entire depth of a computer system application,the front end and the backend.In this project we will make exact replica of the most popular website “Stack Overflow” both front end and back end using ReactJs, NodeJs , ExpressJs and MongoDB.

Scope

Stack Overflow is a question and answer website for programmers. Stack Overflow only accepts questions about programming that are tightly focused on a specific problem. Questions of a broader nature—or those inviting answers that are inherently a matter of opinion—are usually rejected by the site's users, and marked as closed.Unregistered users have access to most of the site's functionality, while users who sign in can gain access to more functionality, such as asking or answering a question, establishing a profile and being able to earn reputation to allow functionality like editing questions and answers without peer review or voting to close a question.

Software and Hardware Requirements

HARDWARE:

Ram	:	4GB
Hardisk	:	512GB
Processor	:	2GHz

SOFTWARE:

Language	:	MERN Stack Web Technologies.
Tools	:	Visual Studio Code.
Additional Resources	:	Webpack Yarn Cors Dotenv Npm Nodemon

ABOUT

1. HTML(Hypertext Markup Language)

- HTML (Hypertext Markup Language) is a text-based approach to describing how content contained within an HTML file is structured. This markup tells a web browser how to display text, images and other forms of multimedia on a webpage.
- HTML contain “Tags”.These “tags” contain name-Value pairs known as “Attributes”.
- There are two types of HTML elements:Block level elements which starts in a new line and Inline elements which do not start in a new line.
- Commonly used HTML tags are:<h1>, <h2>, <p>, <table>, <ol>, <ul>.
- HTML elements tell the browser how to display the content.

2. CSS (Cascading Style sheet)

- It is a style sheet language which is used to describe the look and formatting of a document written in markup language. It provides an additional feature to HTML. It is generally used with HTML to change the style of web pages and user interfaces.
- CSS is used along with HTML and JavaScript in most websites to create user interfaces for web applications and user interfaces for many mobile applications.
- We can add new looks to your old HTML documents.
- We can completely change the look of your website with only a few changes in CSS code.
- It saves lot of time and provides more attributes.

3. JAVASCRIPT

- It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages.
- JavaScript is a lightweight, interpreted programming language.
- Designed for creating network-centric applications.

- The JavaScript code is executed when the user submits the form, and only if all the entries are valid, they would be submitted to the Web Server.
- JavaScript can be used to trap user-initiated events such as button clicks, link navigation, and other actions that the user initiates explicitly or implicitly.

4. Node JS:

- Node.js is an open source, cross-platform runtime environment for developing server-side and networking applications. Node.js applications are written in JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux.
- Node.js also provides a rich library of various JavaScript modules which simplifies the development of web applications using Node.js to a great extent.
- Asynchronous and Event Driven.

5. ExpressJS:

- Express.js is a small framework that works on top of Node.js web server functionality to simplify its APIs and add helpful new features. It makes it easier to organize your application's functionality with middleware and routing. It adds helpful utilities to Node.js HTTP objects and facilitates the rendering of dynamic HTTP objects.
- Develops Node.js web applications quickly and easily.
- Includes a number of middleware modules that can be used to execute additional requests and responses activities.

6. ReactJS:

- ReactJS is JavaScript library used for building reusable UI components.
- Uses virtual DOM which is a JavaScript object. This will improve apps performance, since JavaScript virtual DOM is faster than the regular DOM.
- Can be used on client and server side as well as with other frameworks.
- Component and data patterns improve readability, which helps to maintain larger apps.
- We use use React as the V in MVC.

7. MongoDB:

- MongoDB is an open-source document database and leading NoSQL database. MongoDB is written in C++.
- A record in MongoDB is a document, which is a data structure composed of key value pairs similar to the structure of JSON objects.

- MongoDB stores data in flexible documents. Instead of having multiple tables you can simply keep all of your related data together. This makes reading your data very fast.

8. Npm:

- npm is the world's largest Software Registry.Open-source developers use npm to share software.
- The name npm (Node Package Manager) stems from when npm first was created as a package manager for Node.js.
- All npm packages are defined in files called package.json. The content of package.json must be written in JSON.

9. Nodemon:

- The nodemon Module is a module that develop node.js based applications by automatically restarting the node application when file changes in the directory are detected.
- It help to reduce the time of typing the default syntax node <file name> for execution again and again.

REQUIREMENTS AND SPECIFICATIONS

Users and use cases:

Unregistered users (any person with no account) can search questions, Stack Overflow user accounts, collaborations, associated companies and tags.

Registered users have accounts, They are capable of asking, and answering questions, create a team, join a team, follow a question, join a collective and perform some other basic actions. With more privileges, they can comment, vote down or up a question, set bounties on questions, create tags and tags synonyms, delete questions etc. More abilities are available with higher privileges.

Team members are those account owners who are belong to a team. They can ask questions and answer questions in their teams. Anything happening within a team is unavailable to any external person.

Collaborators of a team are those team members who can delete content from the team and edit articles.
Admins are team members who can activate and deactivate members of the team, invite any registered user to join the team, grant and revoke privileges to any team member.

We have five main actors in our system:

Admin: Mainly responsible for blocking or unblocking members.

Guest: All guests can search and view questions.

Member: Members can perform all activities that guests can, in addition to which they can add/remove questions, answers, and comments. Members can delete and un-delete their questions, answers or comments.

Moderator: In addition to all the activities that members can perform, moderators can close/delete/undelete any question.

System: Mainly responsible for sending notifications and assigning badges to members.

Functional requirements:

Includes operations that any user of Stack Overflow is capable of performing.

1.Visitors:

Unregistered users who can:
. List all available public questions.
. List all companies associated to Stack Overflow.
. Sign up to Stack Overflow.
. List all available tags.
. List all communities of Stack Overflow.

2.Members:

Users who own account and depending on their privileges, they are capable of:
. Posting questions.
. Posting questions.
. Posting answers.
. Modifying answers.
. comment a question.
. Up vote or down vote a post.
. Delete asked question.
. Delete any question.
. Sign in.
. Join teams.
. Create teams.
. change email settings.
. logout and more .....

3.Team members:

Any member of the platform who participates in a team can:
. Ask a question in a team
. Answer a question in a team.

4.Team Moderators:

A member of a given team who can:
. Edit an article in a team.
. Delete contents of the team.

5.Admin:

Team members with the highest privileges:
. Deactivate or activate any member of the team.
. Invite a Stack Overflow member to join his team.

. Change team member role.

Non-Functional requirements

These requirements define how well the system is going to perform its functions. Non-Functional requirements of Stack Overflow include:

- Scalabilty:

A plat-form like StackOverflow should be capable of handling and increase in the amount of work through addition of resources.

- Performance:

This includes small response time, high throughput (amount of work done per unit time), determines how fast StackOverflow responds to user request.

- Availability:

The probability that a user can access Stack Overflow at any time should be very high even though millions of developers over the world are using StackOverflow at different moments.

- Usability:

Using the platform should be very easy. Stack Overflow has extensive use cases, hence should provide good graphical interfaces with easy to use tools for good user experience.

- Security:

Important information on users, teams, collectives and companies should protected from unauthorized individuals. Prevent users from granting themselves privileges.

- Extensibility:

Addition of new function or feature should not be difficult.

DESIGN CONSTRAINTS:

Includes decisions that were already made for the developers restricting their degree of freedom.

1.Using C# as Development language.
2.ASP.NET MVC used as development framework.
3.Using a relational database management system for data storage (Microsoft SQL server).

DESIGN

Design Introduction:

Design is the first step in the development phase for any techniques and principles for the purpose of defining a device, a process or system in sufficient detail to permit its physical realization. Once the software requirements have been analyzed and specified the software design involves three technical activities - design, coding, implementation and testing that are required to build and verify the software.

The design activities are of main importance in this phase, because in this activity, decisions ultimately affecting the success of the software implementation and its ease of maintenance are made. These decisions have the final bearing upon reliability and maintainability of the system.Design is the only way to accurately translate the customer’s requirements into finished software or a system.

Design is the place where quality is fostered in development. Software design is a process through which requirements are translated into a representation of software. Software design is conducted in two steps. Preliminary design is concerned with the transformation of requirements into data.

UML Diagrams:

Actor:

A coherent set of roles that users of use cases play when interacting with the use cases.an observable result of value of an actor.

Use case:

A description of sequence of actions, including variants, that a system performs yields
an observable result of value of an actor. actor diagram is drawned in a eclipse shape.

UML stands for Unified Modeling Language. UML is a language for specifying, visualizing and ocumenting the system. This is the step while developing any product after analysis. The goal from this is to produce a model of the entities involved in the project which later need to be built. The representation of the entities that are to be used in the product being developed need to be
designed.

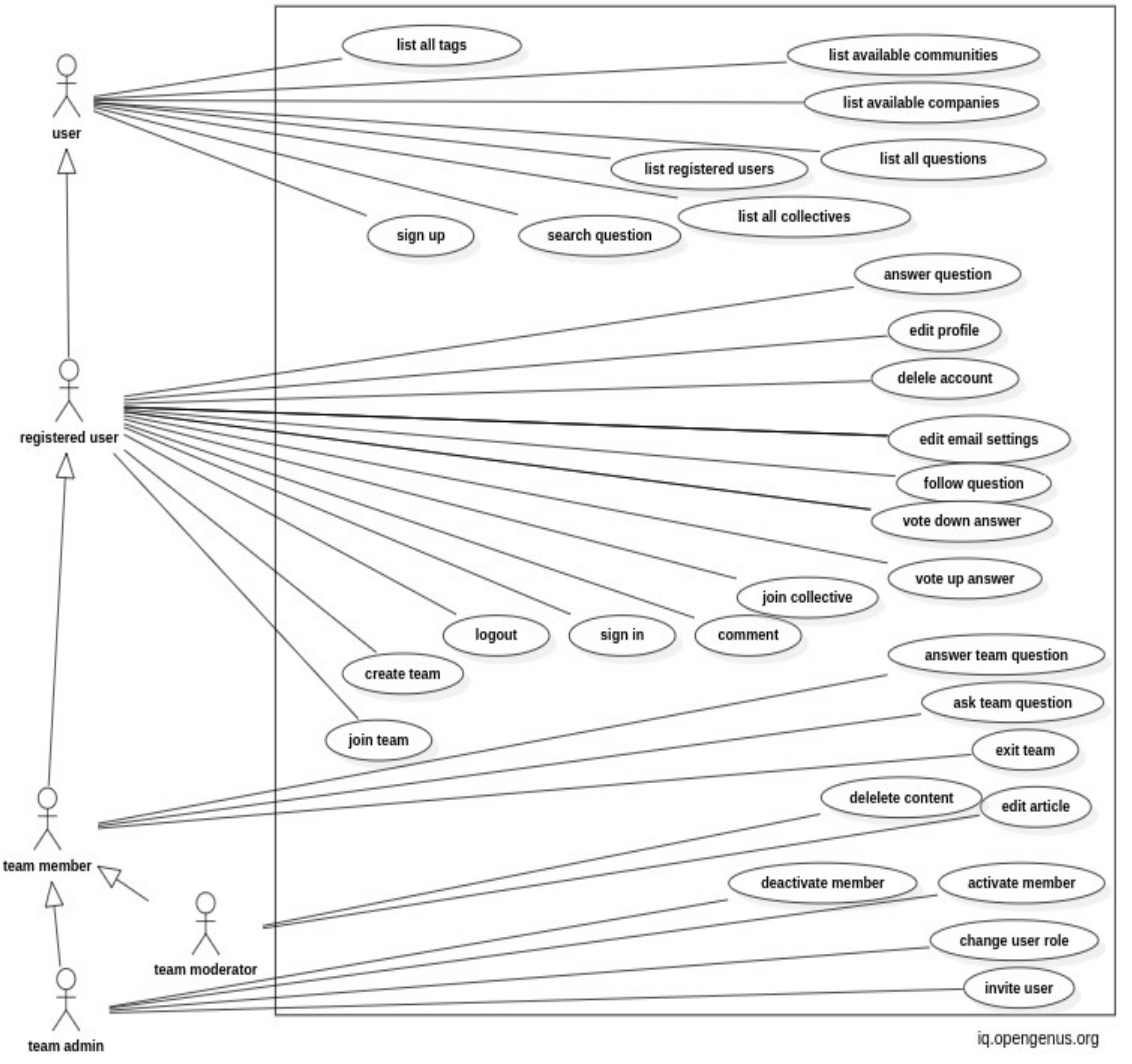
USECASE DIAGRAMS:

Use case diagrams model behavior within a system and helps the developers understand of what the user require. The stick man represents what's called an actor.

Use case diagram can be useful for getting an overall view of the system and clarifying that can do and more importantly what they can't do.

Use case diagram consists of use cases and actors and shows the interaction between the use case and actors.

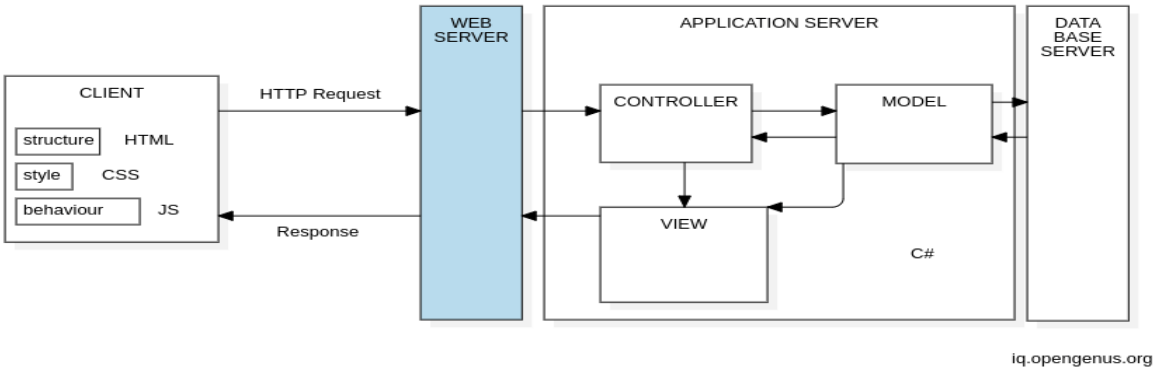
1.The purpose is to show the interactions between the use case and actor.
2.To represent the system requirements from user's perspective.
3.An actor could be the end-user of the system or an external system.



Fig(i): Usecase Diagram of Stack Overflow website.

System Architecture:

Stack Overflow is designed with respect to the Model-View-Controller Architectural pattern (MVC) which is a modern representation of a monolithic architecture. Such an architecture can be represented as follows**:**



Fig(ii):Stack overflow website system Architecture

Class Diagram:

Modeling a system like Stack Overflow is done using the Object Oriented approach. The internal structure of the software system will depend on the structure of what is to be modeled, "an online platform for developers to share knowledge", which consists of entities such as developers (users), questions, teams, answers, solutions, tags etc , some of which are represented below:

Here are the main classes of Stack Overflow System:

Question: This class is the central part of our system. It has attributes like Title and Description to define the question. In addition to this, we will track the number of times a question has been viewed or voted on. We should also track the status of a question, as well as closing remarks if the question is closed.

Answer: The most important attributes of any answer will be the text and the view count. In addition to that, we will also track the number of times an answer is voted on or flagged. We should also track if the question owner has accepted an answer.

Comment: Similar to answer, comments will have text, and view, vote, and flag counts. Members can add comments to questions and answers.

Tag: Tags will be identified by their names and will have a field for a description to define them. We will also track daily and weekly frequencies at which tags are associated with questions.

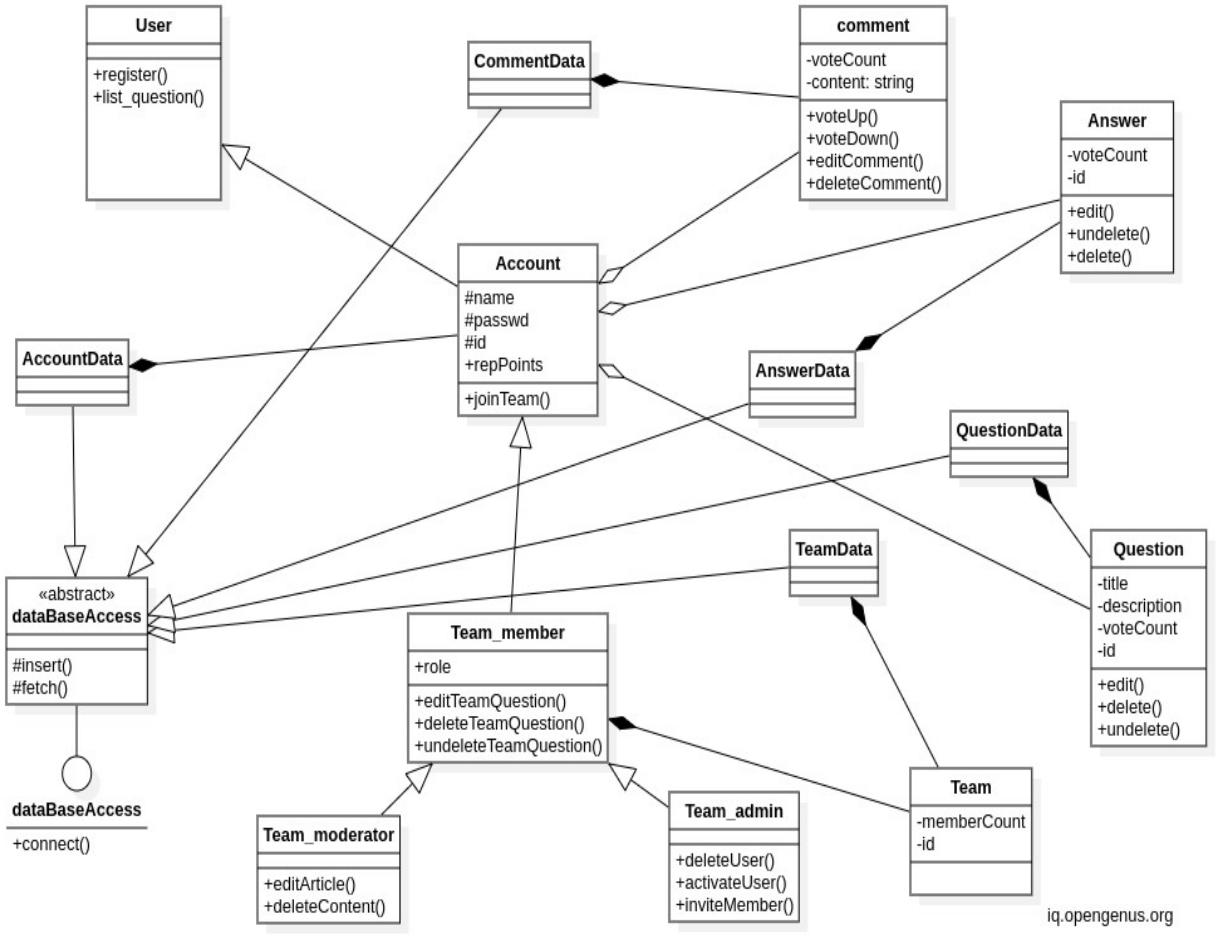
Badge: Similar to tags, badges will have a name and description.

Photo: Questions or answers can have photos.

Bounty: Each member, while asking a question, can place a bounty to draw attention. Bounties will have a total reputation and an expiry date.

Account: We will have four types of accounts in the system, guest, member, admin, and moderator. Guests can search and view questions. Members can ask questions and earn reputation by answering questions and from bounties.

Notification: This class will be responsible for sending notifications to members and assigning badges to members based on their reputations.



Fig(iii):Class Diagram of stack overflow website

Implementation and System Testing

After all phase have been perfectly done, the system will be implemented to the server and the system can be used.

System Testing:

The goal of the system testing process was to determine all faults in our project .The program was subjected to a set of test inputs and many explanations were made and based on these explanations it will be decided whether the program behaves as expected or not. Our Project went through two levels of testing.
1. Unit testing
2 .Integration testing

Unit Testing:

Unit testing is commenced when a unit has been created and effectively reviewed .In order to test a single module we need to provide a complete environment i.e. besides the section we would require The procedures belonging to other units that the unit under test calls Non local data structures that module accesses .A procedure to call the functions of the unit under test with appropriate parameters.

Testing SignUp page:
If user is not sign Up already then try entering username,Mail and set a password for this account.If user try to signup using already existing account then it must display a message indicating ‘account already exist”.

Testing Login page:
If user is already have an account then try entering Mail and Password and check whether able to sign in or not.Also if user try to login with a non-existing account then he/she won’t be able to login.

Testing Users page:
Users page contain all the signup users.Check whether all the registered users details are displayed in users page or not.

Integration Testing:

In the Integration testing we test various combination of the project module by providing the input.

The primary objective is to test the module interfaces in order to confirm that no errors are occurring when one module invokes the other module.

Basic Folder Structure of project:

create a folder with name STACK-OVERFLOW-CLONE.
Inside this folder create two sub folders with name “Client” and “Server”.

1.Install NodeJS if not already present: NodeJS is needed since the Libraries Required for React are downloaded using node package manager (npm).

2.Install create-react-app Node Package:create-react-app node package helps to set up a React project. Install create-react-app node package globally using the following command.

npm install -g create-react-app

3.Install all the dependencies on client side: The following are the dependencies which we will use in the client-side during the entire project.Install those dependencies using following command:

yarn add axios jwt-decode moment react-copy-to-clipboard react-router-dom redux react-redux redux-thunk

4.Install following dependencies on server side: The following are the dependencies which we will use in the server-side during the entire projetc.Install using following command:

npm i bycrypt jsonwebtoken cors dotenv express mongoose nodemon

Create the project: The project can be created using create-react-app. Use the following command to create the project.

npx create-react-app ./

In the client-side use following command:*yarn start*

Go to your browser and go the following URL **localhost:3000** You should be able to see that your application is running. The Application will look like this in your browser:

To get started, edit `src/App.js` and save to reload.

Basic Folder Structure Explained:

1.package.json: This File has the list of node dependencies which are needed.

2.public/index.html: When the application starts this is the first page that is loaded. This will be the only html file in the entire application since React is generally Written using JSX. Also, this file has a line of code . This line is very significant since all the application components are loaded into this div.

3.src/index.js: This is the javascript file corresponding to index.html. This file has the following line of code which is very significant. ReactDOM.render(<App />, document.getElementById(‘root’));

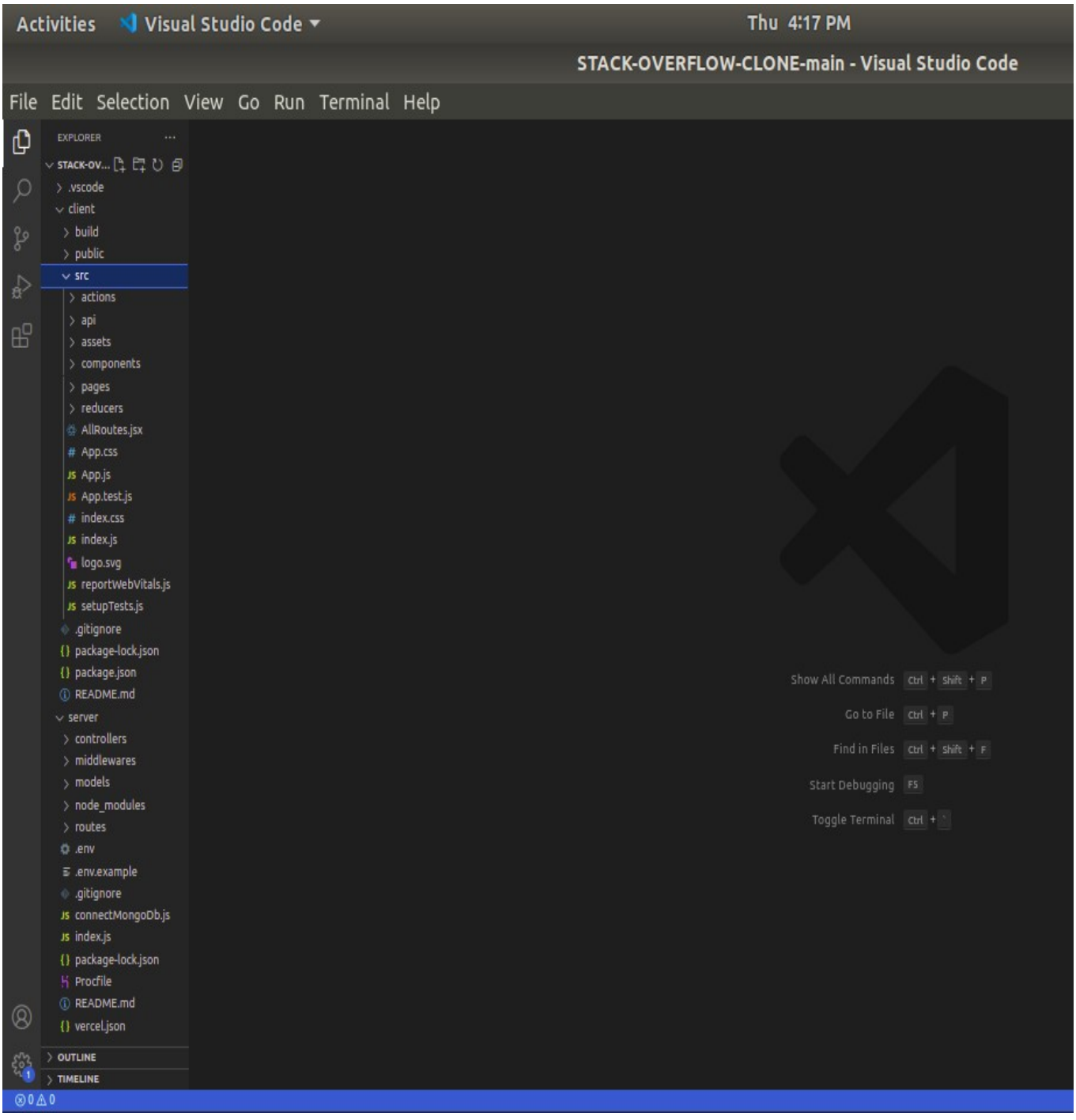
4.The above line of code is telling that App Component has to be loaded into an html element with id root. This is nothing but the div element present in index.html.

5.src/index.css: The CSS file corresponding to index.js.

6.src/App.js : This is the file for App Component. App Component is the main component in React which acts as a container for all other components.

7.src/App.css : This is the CSS file corresponding to App Component.

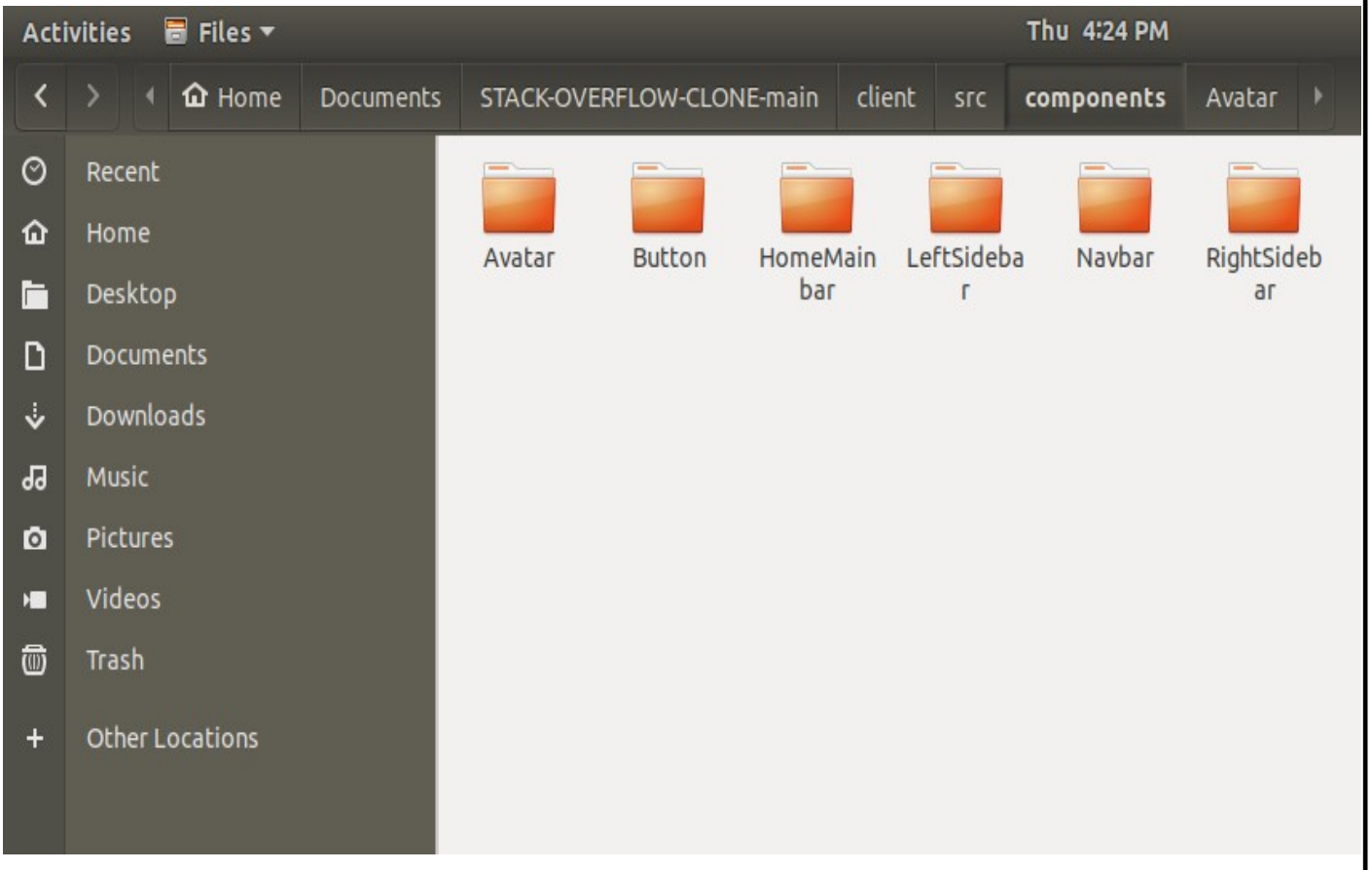
8.Build: This is the folder where the built files are stored. React Apps can be developed using either JSX, or normal JavaScript itself, but using JSX definitely makes things easier to code for the developer . But browsers do not understand JSX. So JSX needs to be converted into javascript before deploying. These converted files are stored in the build folder after bundling and minification.



Fig(iv):Folder structure of stack-overflow clone project

Creating Components:

A Component in React does a specific functionality. An Application is nothing but a collection of components. Each component can have multiple child components and the components can communicate with each other.



Fig(v):Components in stack-overflow clone project

CODE:

index.html:

<!doctype html><html lang="en"><head><meta charset="utf-8"/><link rel="icon" href="/favicon.ico"/><meta name="viewport" content="width=device-width,initial-scale=1"/><meta name="theme-color" content="#000000"/><meta name="description" content="Web site created using create-react-app"/><title>StackOverflow Clone</title><script defer="defer" src="/static/js/main.33257b31.js"></script><link href="/static/css/main.9af40bb1.css" rel="stylesheet"></head><body><noscript>You need to enable JavaScript to run this app.</noscript> </body></html>

index.js:

```
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';
import reportWebVitals from './reportWebVitals';
import {Provider } from 'react-redux'
import {createStore,applyMiddleware,compose} from 'redux'
import thunk from 'redux-thunk'
import Reducers from'./reducers'
const store= createStore(Reducers, compose(applyMiddleware(thunk)))
ReactDOM.render(
  <Provider store={store}>
    <React.StrictMode>
      <App/>
    </React.StrictMode>
  </Provider>,
  document.getElementById('root')
);
reportWebVitals();
```

index.css:

```
body {
  margin: 0;
  font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', 'Roboto', 'Oxygen',
    'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue',
    sans-serif;
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
}
code {
  font-family: source-code-pro, Menlo, Monaco, Consolas, 'Courier New',
    monospace;
}
```

App.css:

```
.home-container-1{
    min-height:100vh;
    max-width:1250px;
    width:100%;
    display:flex;
    justify-content:space-between;
    margin: 0% auto;
}
.home-container-2{
    max-width:1100px;
    width: calc(100% - 164px);
    border:1px solid #d6d9dc;
    padding:24px;
    box-sizing: border-box;


}
```

App.js:

```
import { BrowserRouter as Router } from 'react-router-dom'
import { useEffect } from 'react';
import { useDispatch } from 'react-redux';
import './App.css';
import Navbar from './components/Navbar/Navbar'
import AllRoutes from './AllRoutes'
import { fetchAllQuestions } from './actions/question';
import { fetchAllUsers } from './actions/users';
function App() {
  const dispatch = useDispatch()
  useEffect(() => {
    dispatch(fetchAllQuestions())
    dispatch(fetchAllUsers())
  }, [dispatch])
  return (
    <div className="App">
      <Router >
        <Navbar />
        <AllRoutes />
      </Router>
    </div>
  );
}
export default App;
```

MongoDB Atlas for Database:

MongoDB is a document database and can be installed locally or hosted in the cloud.

MongoDB Atlas, a cloud database platform. This is much easier than hosting your own local database.

Steps to use MongoDB Atlas:

1.Sign up for a free MongoDB Atlas account to get started.

2.After you have created your account, set up a free "Shared Cluster" then choose your preferred cloud provider and region.

3.By default, MongoDB Atlas is completely locked down and has no external access.

4.You will need to set up a user and add your IP address to the list of allowed IP addresses.

5.Under "Database Access", create a new user and keep track of the username and password.

6.Next, under "Network Access", add your current IP address to allow access from your computer.

Connect to the database:

1.To connect to your database, you will need your database specific connection string.

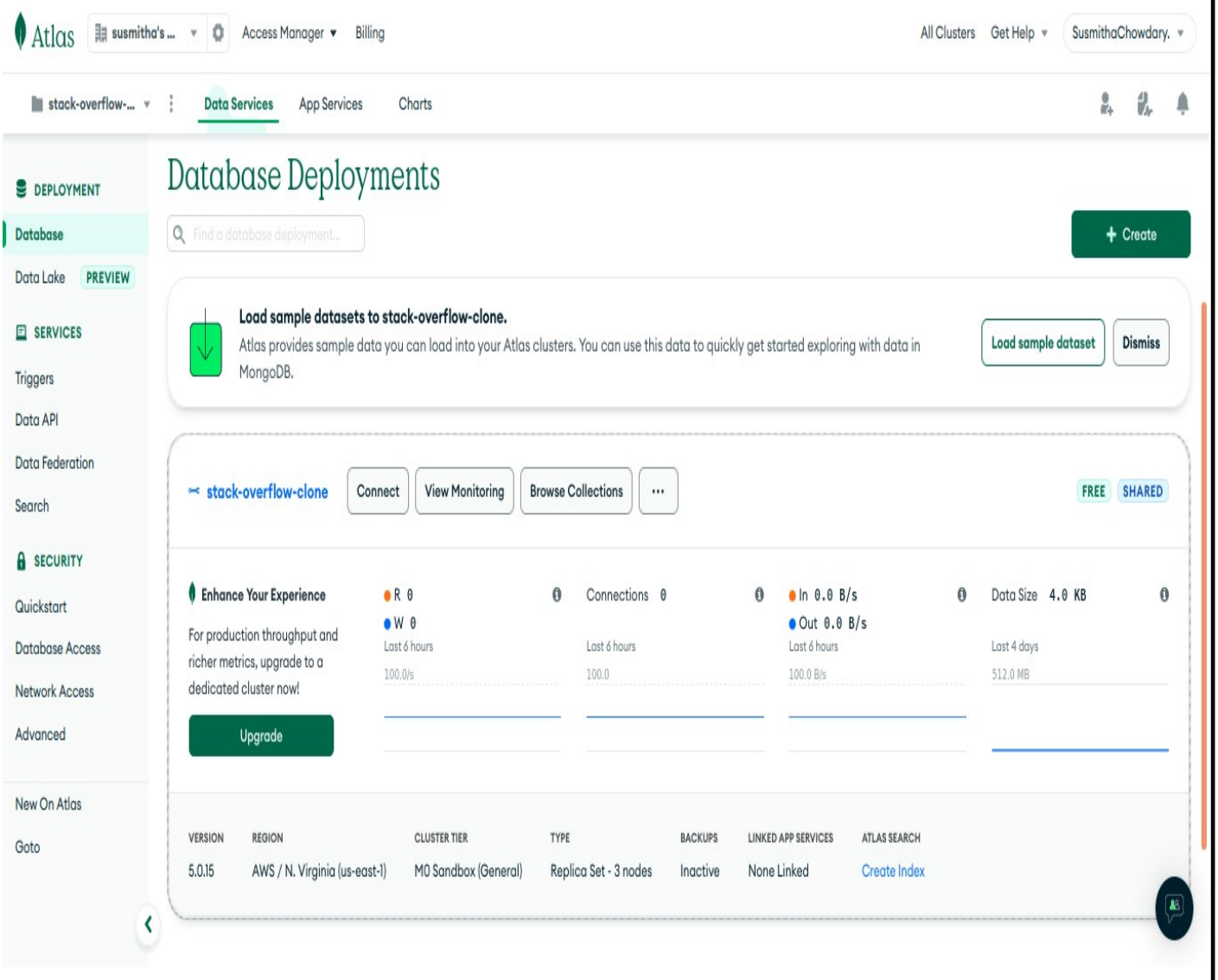
2.In the MongoDB Atlas dashboard, under "Databases", click the "Connect" button for your Cluster.

3.Next, choose "Connect with the MongoDB Shell".

4.Copy your connection string.

Our MongoDB connection url:

mongodb+srv://Susmitha:<password>@stack-overflow-clone.rrxx50o.mongodb.net/?retryWrites=true&w=majority



Fig(vi):MongoDB database for stack-overflow clone project

code:

index.js:

```
import express from 'express'
import cors from 'cors'
import dotenv from 'dotenv'
import userRoutes from './routes/users.js'
import questionRoutes from './routes/Questions.js'
import answerRoutes from './routes/Answers.js'
import connectDB from "./connectMongoDb.js";
dotenv.config();
connectDB();
const app = express();
app.use(express.json({ limit: "30mb", extended: true }));
app.use(express.urlencoded({ limit: "30mb", extended: true }));
app.use(cors());
app.use("/user", userRoutes);
app.use("/questions", questionRoutes);
app.use("/answer", answerRoutes);
const PORT = process.env.PORT || 5000;
app.listen(PORT, () => {
  console.log(`server running on port ${PORT}`);
});
```

.env:

```
PORT = "5000"
CONNECTION_URL=mongodb+srv://Susmitha:<password>@stack-overflow-clone.rrxx50o.mongodb.net/?retryWrites=true&w=majority
JWT_SECRET = "This could be anything like test"
```

auth.js:

```
import mongoose from "mongoose";
const userSchema =mongoose.Schema({
   name: {type:String, required: true},
   email: {type:String, required: true},
   password: {type:String, required: true},
   about: {type:String },
   tags: {type:[String]},
   joinedOn: {type:Date, default:Date.now}
})
export default mongoose.model("User", userSchema)
```

connectMongoDb.js:

```
import mongoose from "mongoose";
const connectDB = async () => {
  try {
    const conn = await mongoose.connect(process.env.CONNECTION_URL, {
      useNewUrlParser: true,
      useUnifiedTopology: true,
    });
    console.log(`MongoDB connected: ${conn.connection.host}`);
  } catch (error) {
    console.error(error);
    process.exit(1);
  }
};
export default connectDB;
```

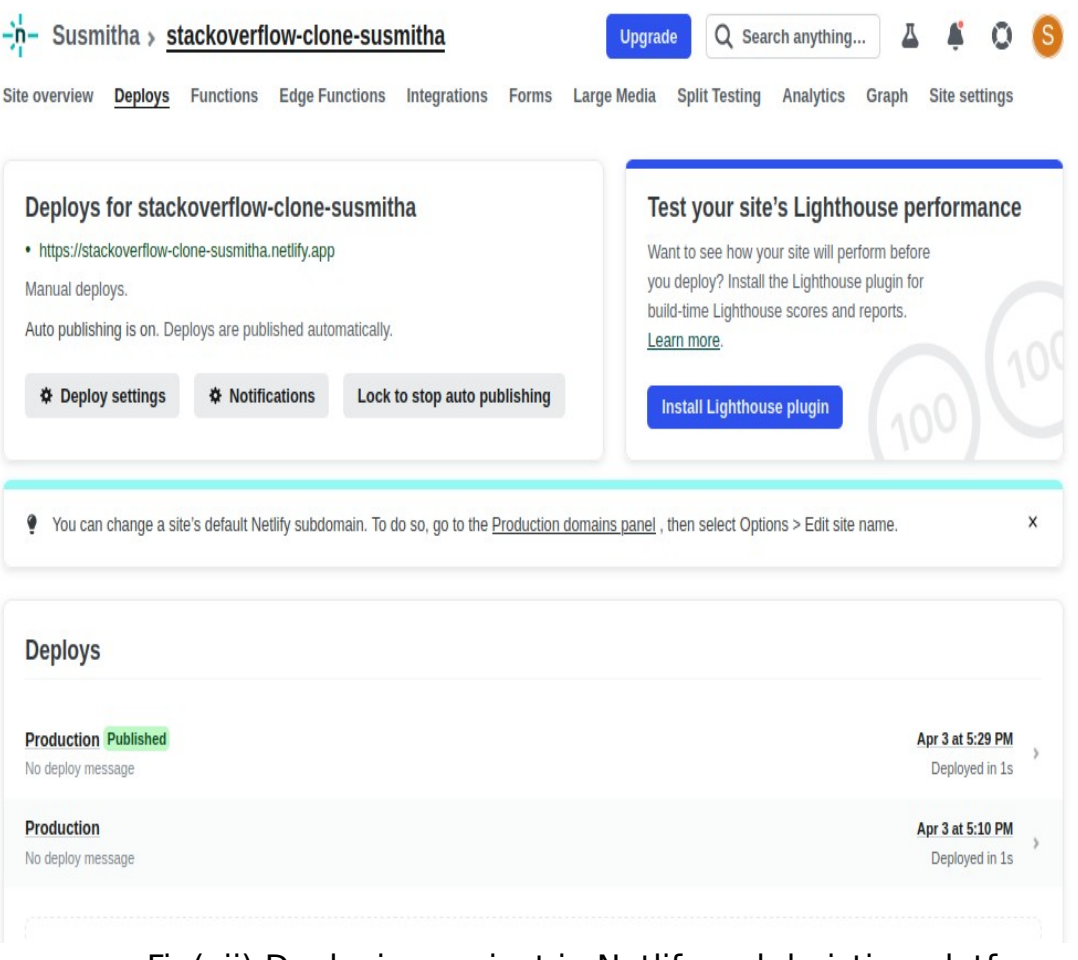
Deployment:

Netlify:

Netlify is a platform for web developers to host their sites in the cloud without managing any servers in the back-end where application logic and database works.

It can have dynamic functionalities which change constantly. Netlify builds its own kind of file storage and management system to push updates both to Git providers and Netlify simultaneously getting connected to the repository.

Every update/changes made to the content is distributed across the servers spread over in the content delivery network and pre-built as static pages with file optimization before being delivered to the users. Visitors of a website in Netlify get the pre-loaded version of website from a geographically nearest server reducing the loading time.

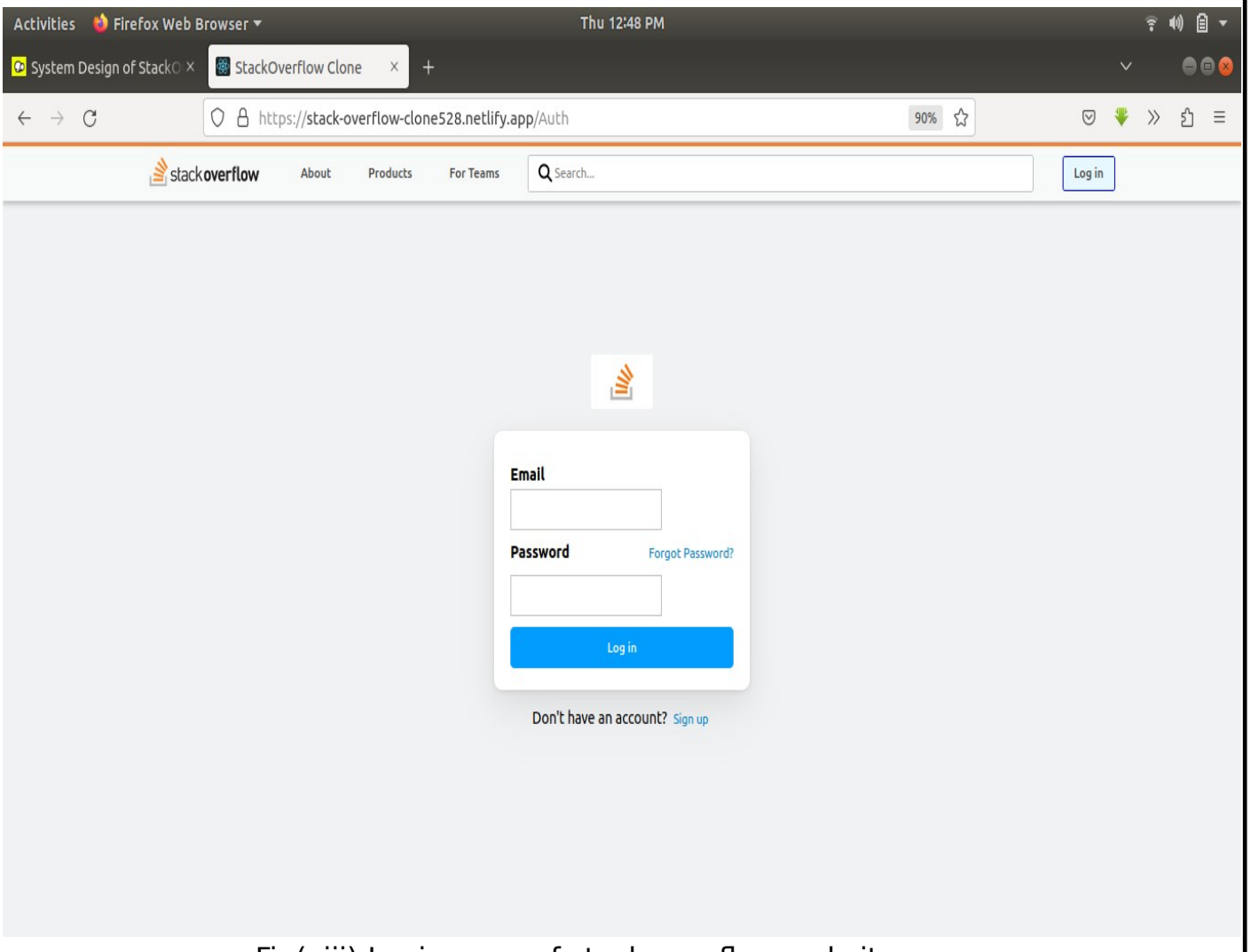


Fig(vii):Deploying project in Netlify web hoisting platform

The following is the link to stack-overflow clone project which is deployed in Netlify platform: *https://stack-overflow-clone528.netlify.app/*

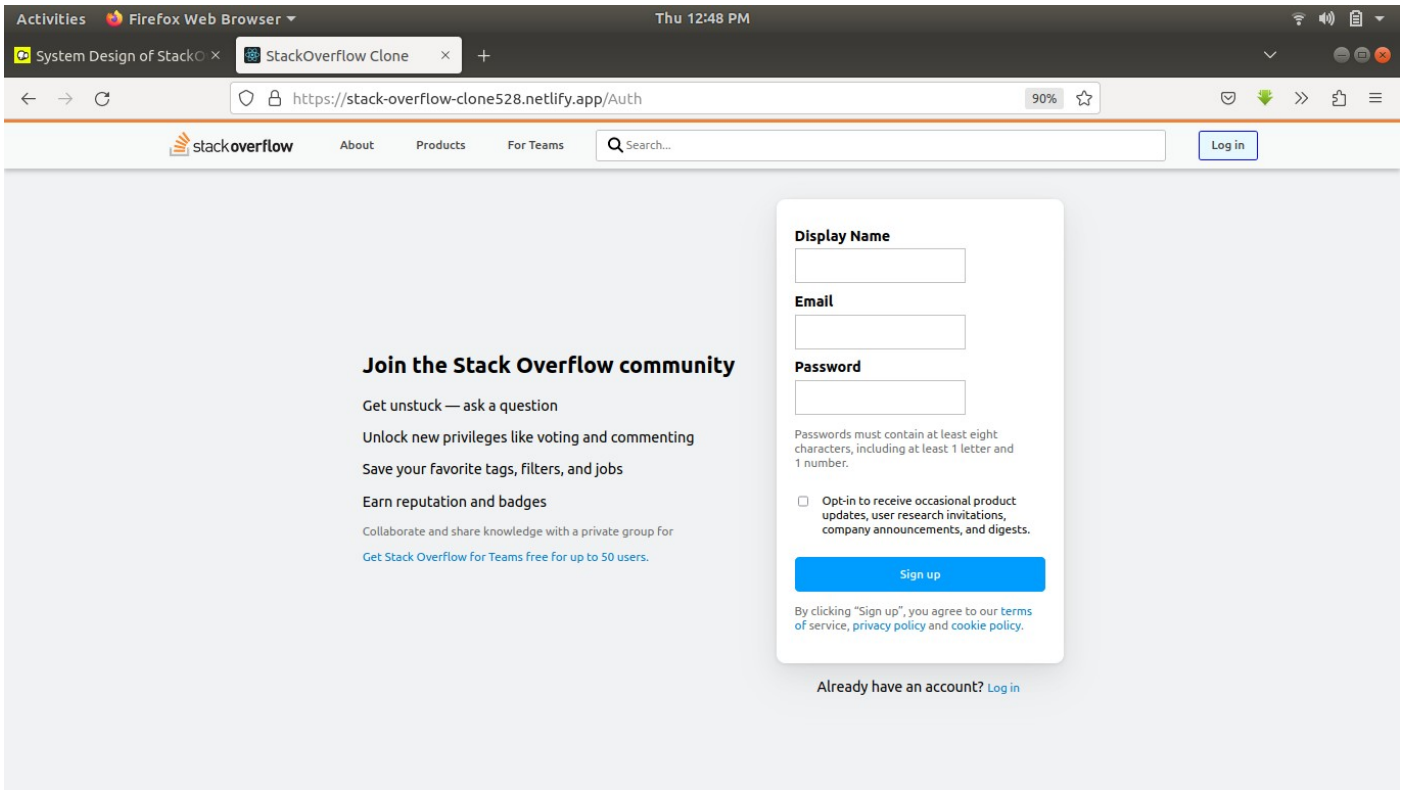
Screenshots:

Login page:



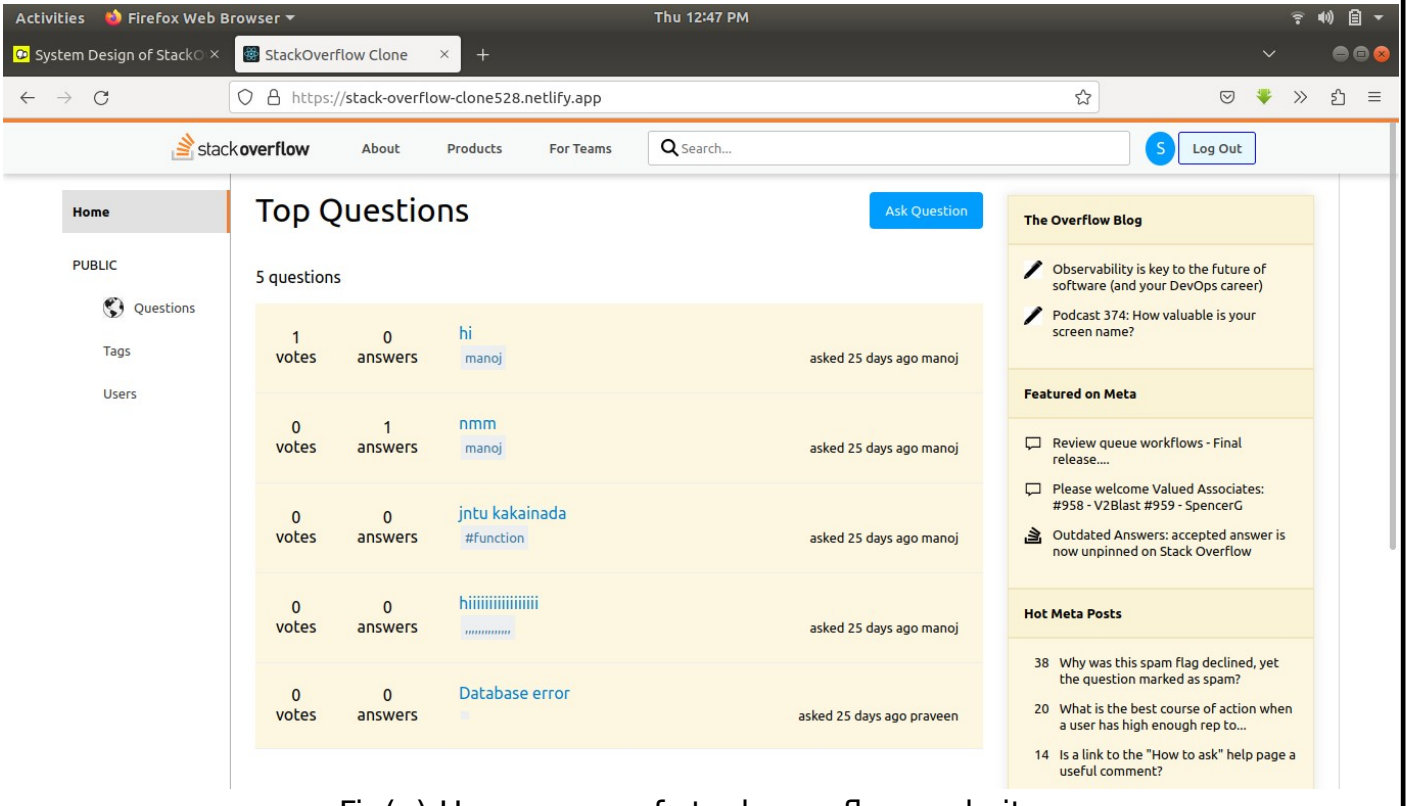
Fig(viii):Login page of stack-overflow website

Signup page:



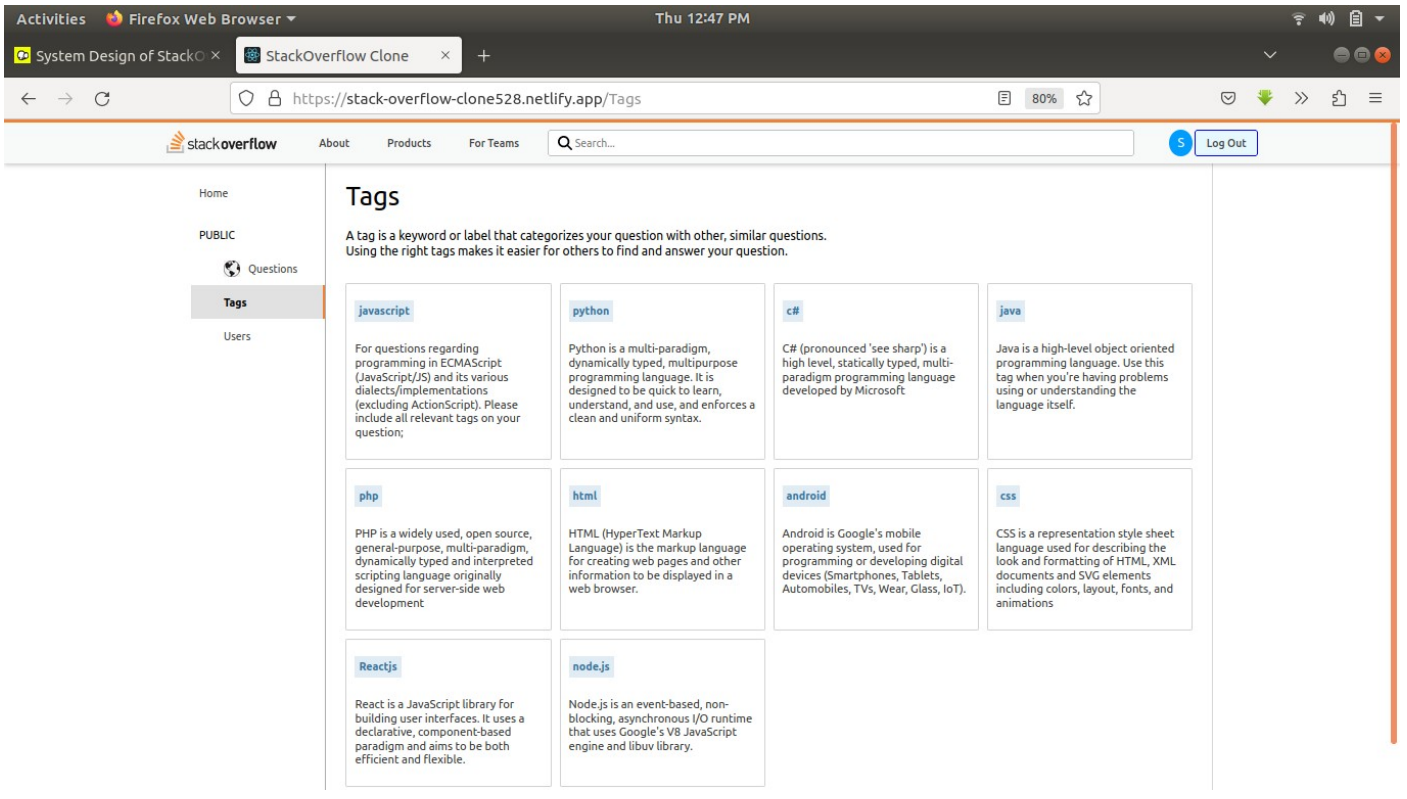
Fig(ix):Sign-up page of stack-overflow website

Home page:



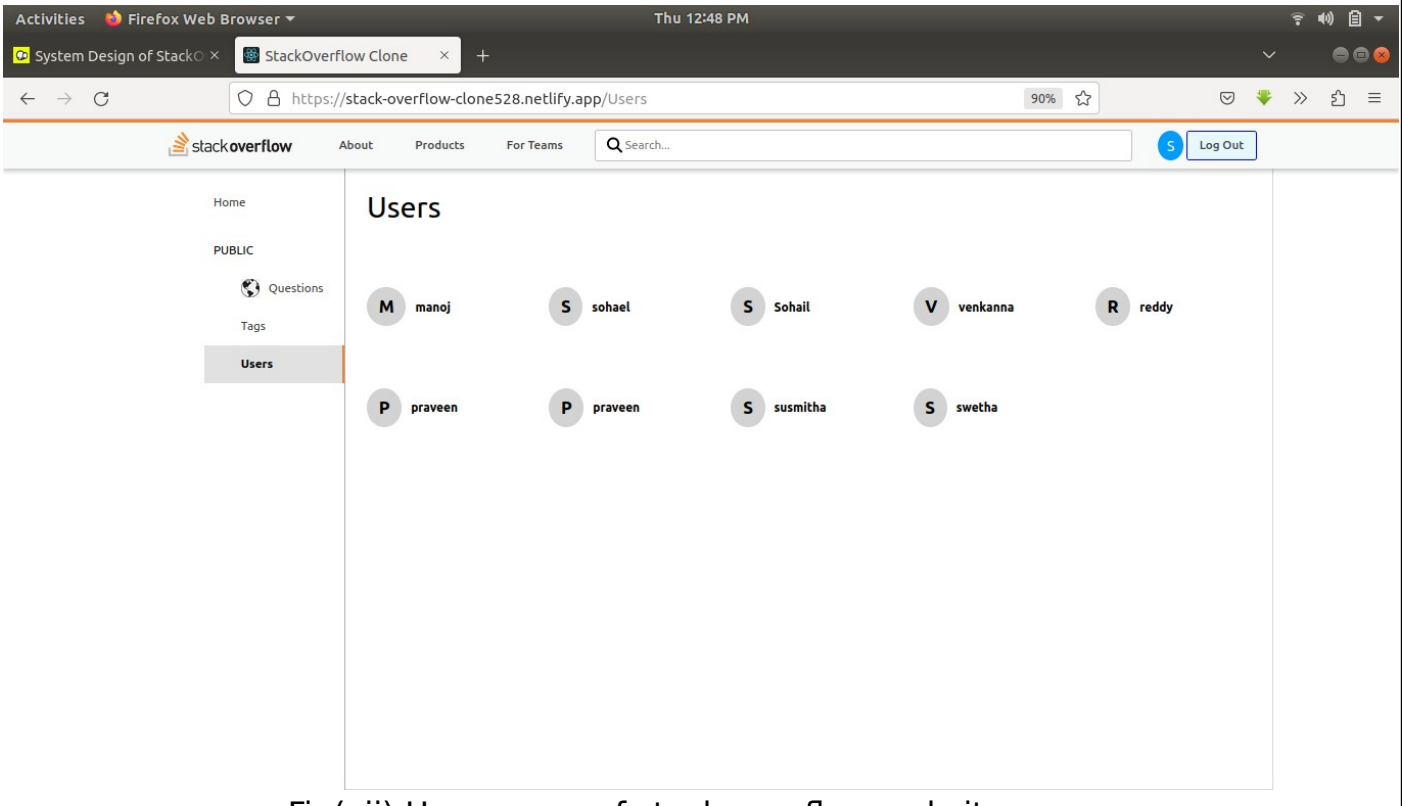
Fig(x):Home page of stack-overflow website

Tags page:



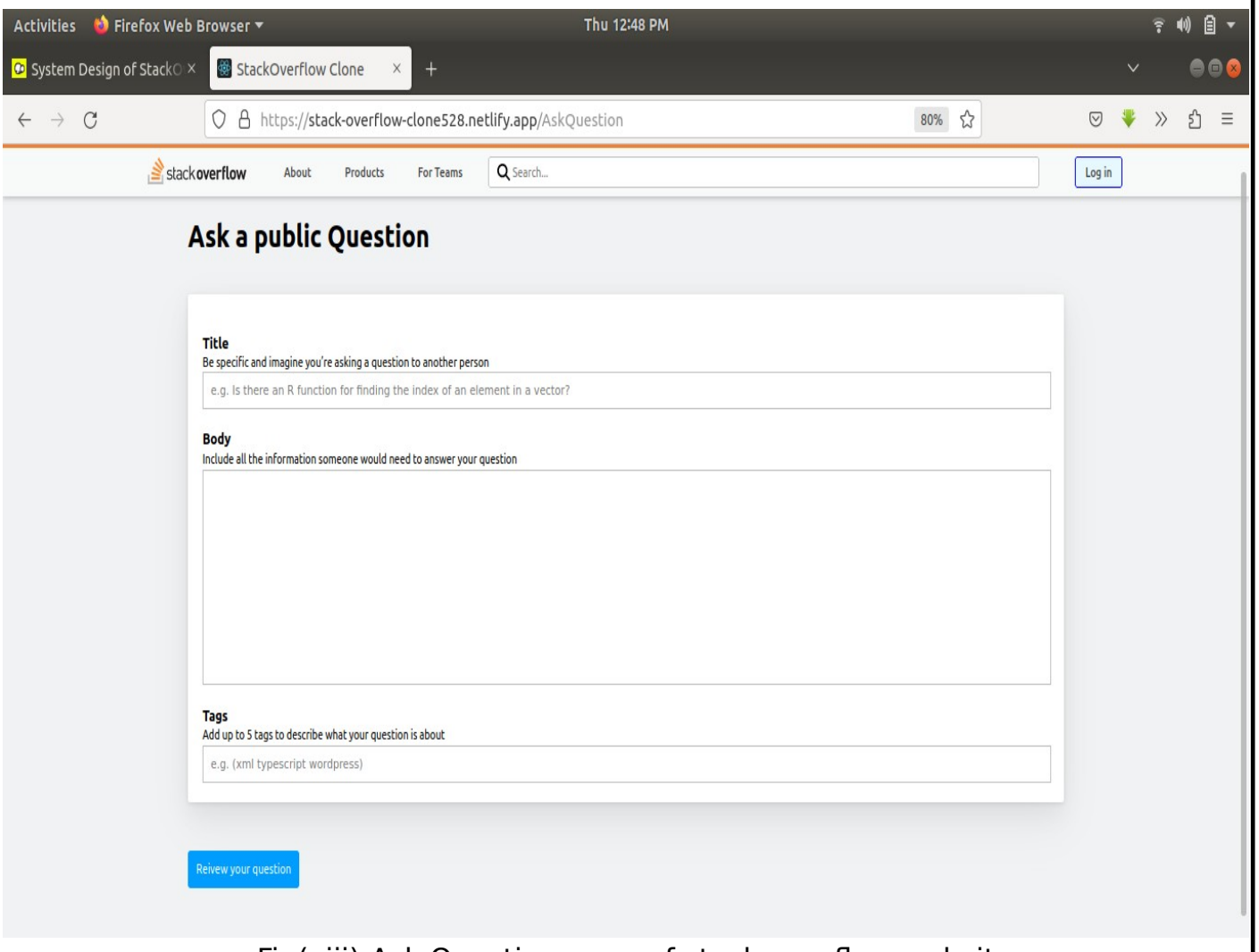
Fig(xi):Tags page of stack-overflow website

Users page:



Fig(xii):Users page of stack-overflow website

Ask Question page:



Fig(xiii):Ask Question page of stack-overflow website

CONCLUSION

Stack Overflow is an incredible collection of programming and software knowledge. Thousands of students and developers use it daily to lookup answers for questions they can’t solve easily themselves.Stack Overflow is not a discussion site or a forum. It’s a knowledge base. That means for whatever questions are asked, any answer should be accurate, specific, and useful to the highest degree. It’s not a place to post partial answers or to ask questions on questions. Answers are upvoted and downvoted to determine the communities opinion on them.

Stack Overflow is a valuable resource for programmers around the world. Its question and answer format, tagging system, and community guidelines make it a useful tool for solving technical problems and learning new skills.By following the website's guidelines and contributing to the community, users can help ensure that Stack Overflow remains a useful and reliable resource for years to come.

References:

https://www.w3schools.com/REACT/DEFAULT.ASP

https://www.w3schools.com/nodejs/

https://www.w3schools.com/mongodb/

https://nullclass.com/

https://iq.opengenus.org/system-design-of-stackoverflow/

https://blog.logrocket.com/getting-started-with-create-react-app-d93147444a27/

**********THANK YOU**********