1. Introduction

1.1 Aim

The project ‘StuAdvisor’ aims to solve the problem of students while they are in search of good college for completing their study. Our project ‘StuAdvisor’ will help them to find college according to the needs of students like it can be in the same city, same State or in the country (India) for the students. They can easily search college and can get admission without involving the third person in the middle. Our aim to provide them best content at free of cost and easily. We will help them in their academics part by giving them study materials like Notes, Important Questions, Sample Papers and Previous Years Exam Papers to study. They can also read blogs about any particular topic they want to study for clearing their doubts.

Our goal is to create the community of students where they can help each others by posting important content for each others. In future, our goal is to provide them functionality to chat with each other on our platform where they can chat personally with each other and they can talk and give their views on a particular topic of subject through the functionality of comments.

On StuAdvisor, faculties can also post study related stuff for students to helping them out. Teachers and Students can easily maintain their profile after registration. In future, our aim is to add more functionalities to users like teacher’s profile will be identified by other users and same students profile will be identified easily. For achieving this, we can add some badges or colors to the profile of users.

Our primary goal is to save the money of students while finding out colleges because if they ask for help to the third person, there will be a chance that he/she will demand some amount of money from the students.

Students don’t need to pay for any types of notes or study material, we will provide them at free of cost and other members of the community can help to solve the particular subject related doubts.

1.2 Project Scope

Project StuAdvisor will help students in many ways and can save the money of students by providing them exact and accurate stuff related to their field of study. This type of projects can easily gain popularity in the market and after gaining popularity we can also earn by collaborating with other entrepreneurs who want to invest in our project. After gaining popularity we can invest more on the project to get more positive outputs.

In future, we can also add ‘live videos’ feature in our platform to give learning in the more easy way to the students where everyone can create and post their videos on our platform and users will be able to come live and can discuss on any doubt/problem.

On StuAdvisor, we can also add chat bot feature for the students where they can ask their questions to our chat bot. our chat bot will guide them to find out the material they searching for.

StuAdvisor can tie-up with the colleges/institutes for helping the students in the admission process so, admissions will be done by us easily for the students at affordable price. We can also use their study material to help the students available on our platform.

In future, we can also help students to get job easily by sharing jobs and vacancies information personally. We can also tie-up with industry level partners to provide internships to our students on our platform.

In future, we can also add data about foreign colleges/universities on our platform for the students who want to study abroad. Students will easily search colleges abroad and we will help them in the process of admission.

1.3 Existing System

There are some websites available on the internet for finding out colleges like –

1. Careers360.com
2. Collegeduniya.com
3. Shiksha.com
4. Studyguideindia
5. Collegedekho.com
6. Careerindia etc.

They are also doing the same but the problem is that they are not actually helping the students to get admission in the college. They are allowing students to only find the colleges and using them like customers for them. They are becoming the third person we was talking about in the process of admission and they are not helping them in their academics part.

They are not providing any type of genuine help to the students after the admission in the college. Their field of working area is limited, they are not expanding their platform to help the students more even after the admission.

These websites can help students to find colleges and students can apply for colleges on these websites but they are not providing them personalized help to the students and they are earning huge amount of money by giving admissions to the colleges and sometimes due to this, students have to pay extra fee.

They are helping to take admissions but not on affordable prices and their work area is limited like helping students to take admissions only.

No help related to study like academics support or study material is not giving by these websites to help the students. Students have to visit on the other different websites to find out the content like notes, blogs, sample papers etc.

1.4 Proposed System

For solving the problem discussed above in existing system, we have developed StuAdvisor (a web application) for solving the problems of students in a more efficient way as compared to other websites who are doing the same for students.

We will help students to find out best colleges according to their field and interest where we will interact more with students with our website. We will help students till the date of admission and we will try to provide admission to students in more affordable prices than others. We will take less amount of money from students/colleges in the process of admission. No hidden payment will be done to us. We will take less amount of money only for running our infrastructure. Our main goal is to provide admission to students at affordable price.

Along with this, StuAdvior will also help students to help in their academics part like students can easily download Notes, Important Questions, Sample Papers and Previous Years Exam Papers and many more.

On StuAdvisor, students can also read blogs about the topic they want to study and they can also read other categories rather than educational blogs to gain extra knowledge. Students can search the specific blog they want using search feature implemented by us.

Users can create account on StuAdvisor and can become registered user on StuAdvisor and then they will be able to post study material in the various formats like pdf, word, doc, txt etc. and other users on StuAdvisor can download those study materials for studying. After registration, users can post blogs to help other students for solving their problems in the particular topic or field.

Users can also help StuAdvisor by sending us data about colleges which are not available on Stuadvisor database, we will verify the data then we will add the college data to the database of StuAdvisor.

2. Project Analysis

2.1 Project Description

Our project "StuAdvisor" is made up of various trending web development technologies of JavaScript (MERN Stack), in which we used React, Express, Node and MongoDB for the development of our project. Our project is very fast as compared to other websites available on the internet due to the latest development technologies.

Our project 'StuAdvisor' is the single page web applications and due to this users don't have to reload the page for using the webapp. Our project will be loaded on the user's system in one time. This will enhance the user's experience while using the project.

Users can use our project like a mobile application on the web browser in which they don't have to wait for any kind of requested data from the server. Connectivity is done by using asynchronous JavaScript with our server for data loading and data is transferred in the JSON format which is very light weight and very fast.

We are storing user's information (data) safely in the very safe format so, there are less chances of getting data leaked. We are storing user's password in the hashed format in our database. We also can not read the passwords of users as it is stored in the encrypted format. We are also using form validation techniques when users will go for registrations on our platform. Users have to give us the real data about themselves otherwise they cannot register on the website. Our form validation features will block the users who will try to put false information in the registration form.

Front-End –

HTML, CSS, JavaScript, React

Back-End –

Express, Node

DataBase –

MongoDB & Mongoose

2.2 Functional Requirements

For running our project in the development phase, some tools/softwares are needed –

- Node JS (JavaScript run time environment)
- MongoDB and MongoDB Compass
- Visual Studio Code Editor
- And an updated web browser

For setting up the project, various project dependencies should also installed in the project which is used for running the project –

- All the list of dependencies that needed to be installed before running the project is listed in the package.json file in the project
- Terminal should be open inside the project directory to install the requirements for installing the all dependencies.
- You have to install the dependencies on client and server both sides, In the both directories, you can open terminal and can execute a single command for installing the all dependencies.
- The command can be only executed after the Node js installation and the command is – 'npm install'

For running the project on your localhost, some other commands are used to run the project in the development mode which are –

- Open terminal in the server directory and execute the command – 'node app.js'
- Then open terminal in the client directory and execute this command to start the project – 'npm start'

After executing these commands, you will be able to see the project that we developed in your browser on your localhost i.e., 127.0.0.1:3000

2.3 Non-Functional Requirements

- Any device can be used for running the project as this project is based on web development so there is no need for any specific platform for running the project. Any OS will able to run the project.

- Only browser is sufficient to run the project easily and efficiently.

- There is no need of high computation in the project so it is okay to have a nomal descent specifications in the system for running the project.

- No active internet connection is required to run the project in the development phase, it can run on your localhost even without any active internet connection.

- You don't have to add extra data to the database initially to run the project as dummy data and information about colleges, doc files, blogs are already provided in the project to check the functionality of the project.

2.4 System Requirements

2.4.1 Software Requirements –

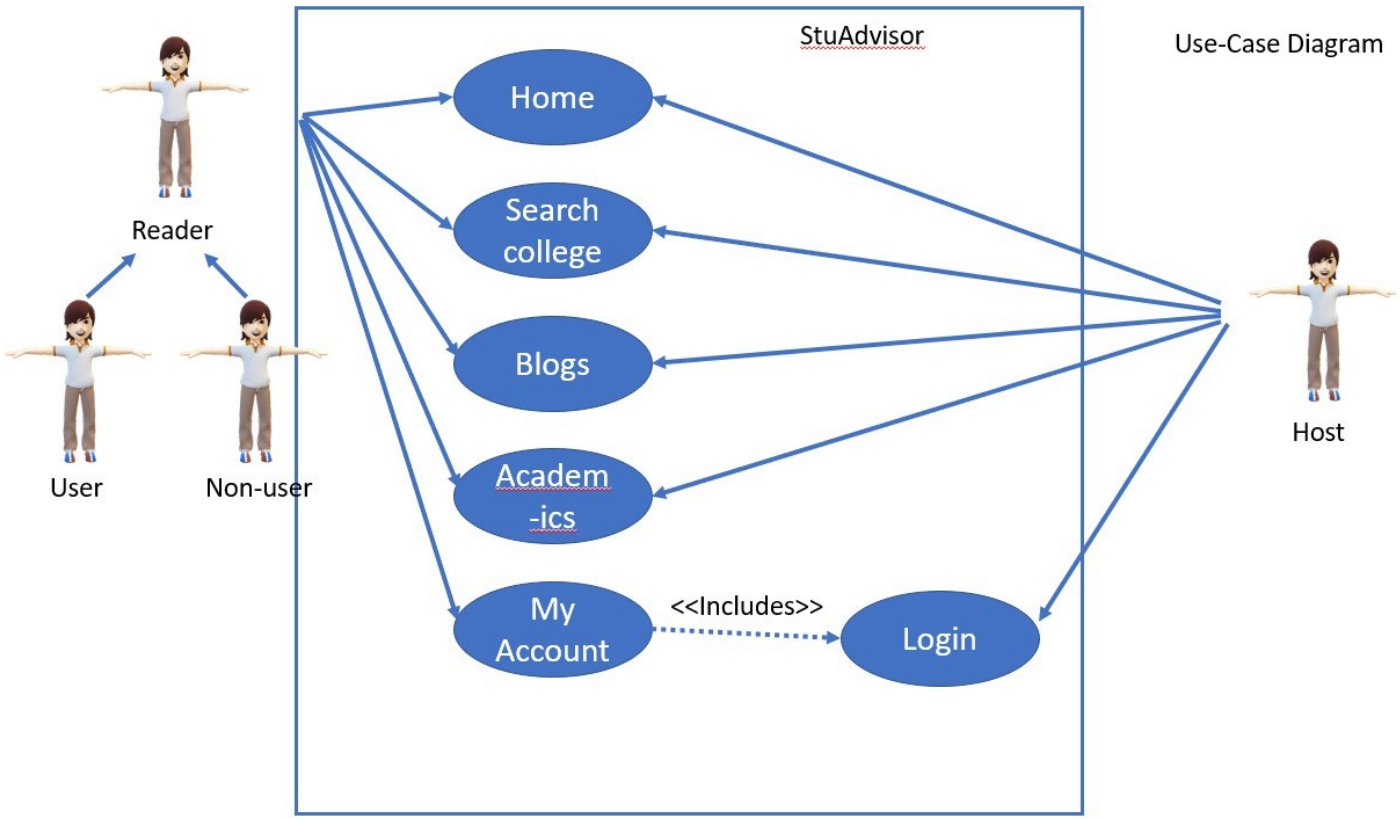
- Operating System can be windows XP, windows 7 or more.
- Active internet connection
- An updated web browser

2.4.2 Hardware Requirements –

- Processor windows – intel dual core
- Minimum RAM – 500 MB
- Recommended RAM – 1 GB
- Minimum Disk Space – 500 MB
- Recommended Disk Space – approx 1 GB or above

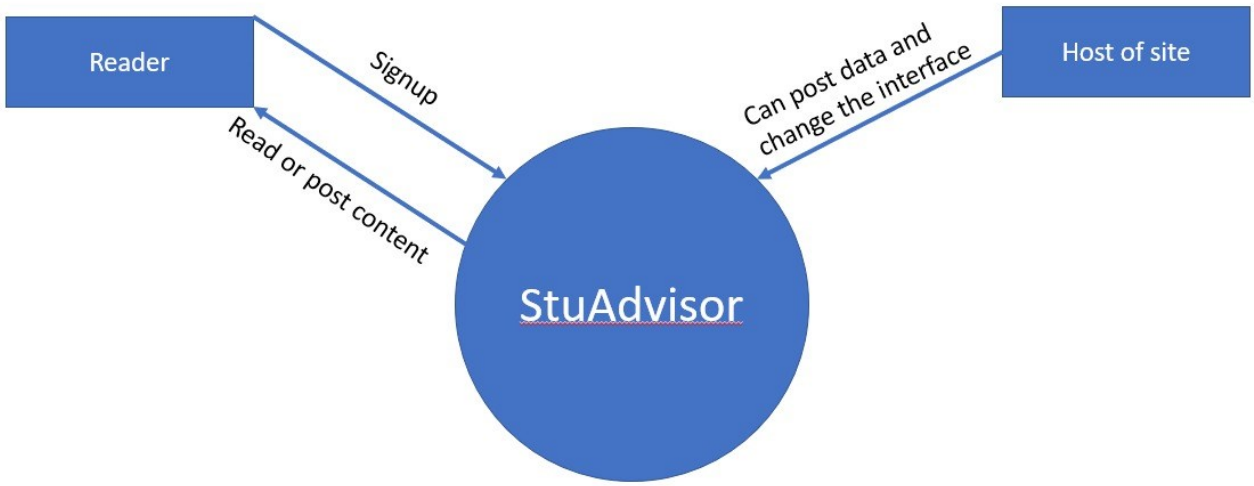
3. Project Diagram

3.1 Use Case Diagram

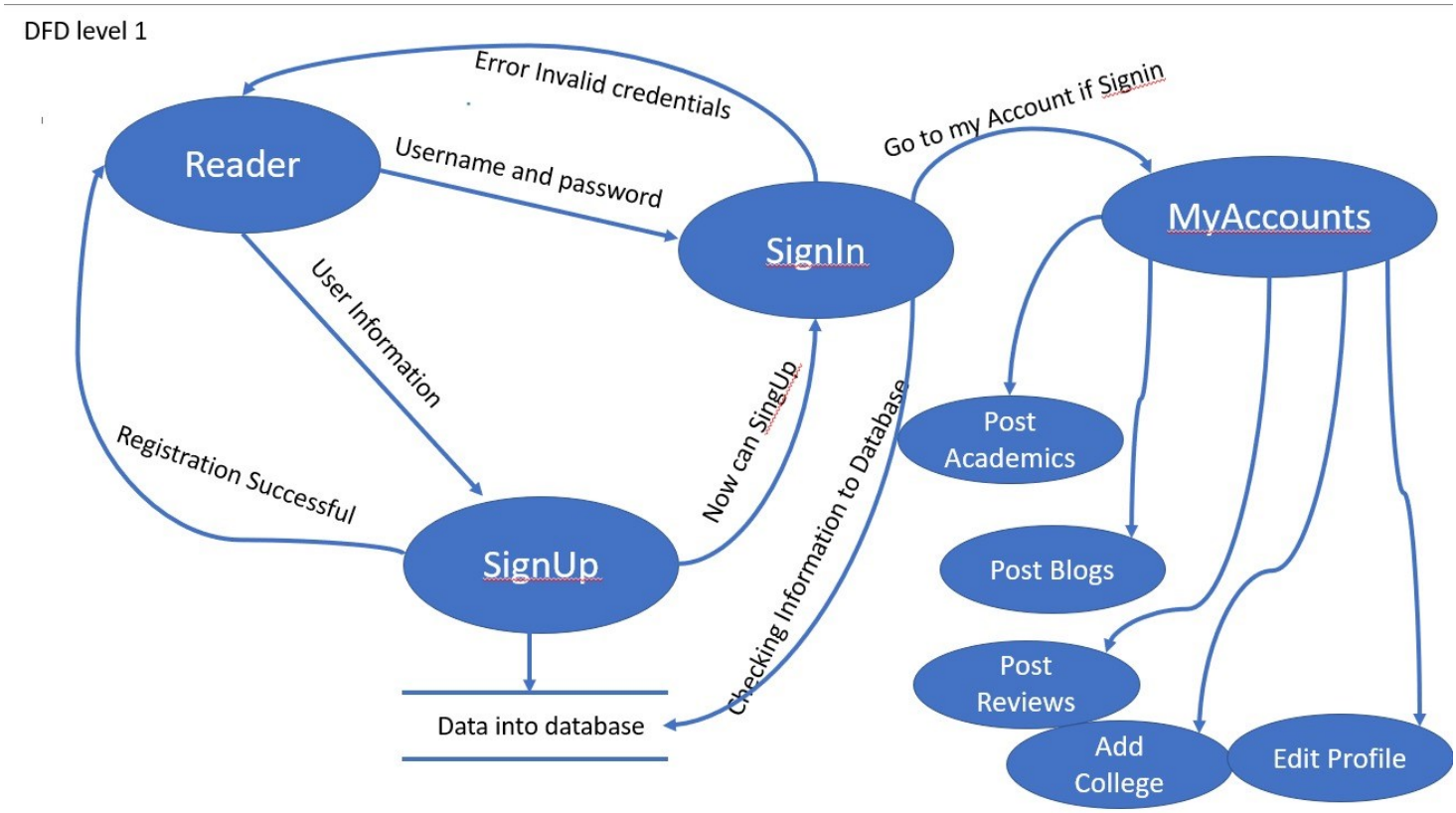


3.2 Data Flow Diagram (Level 0)

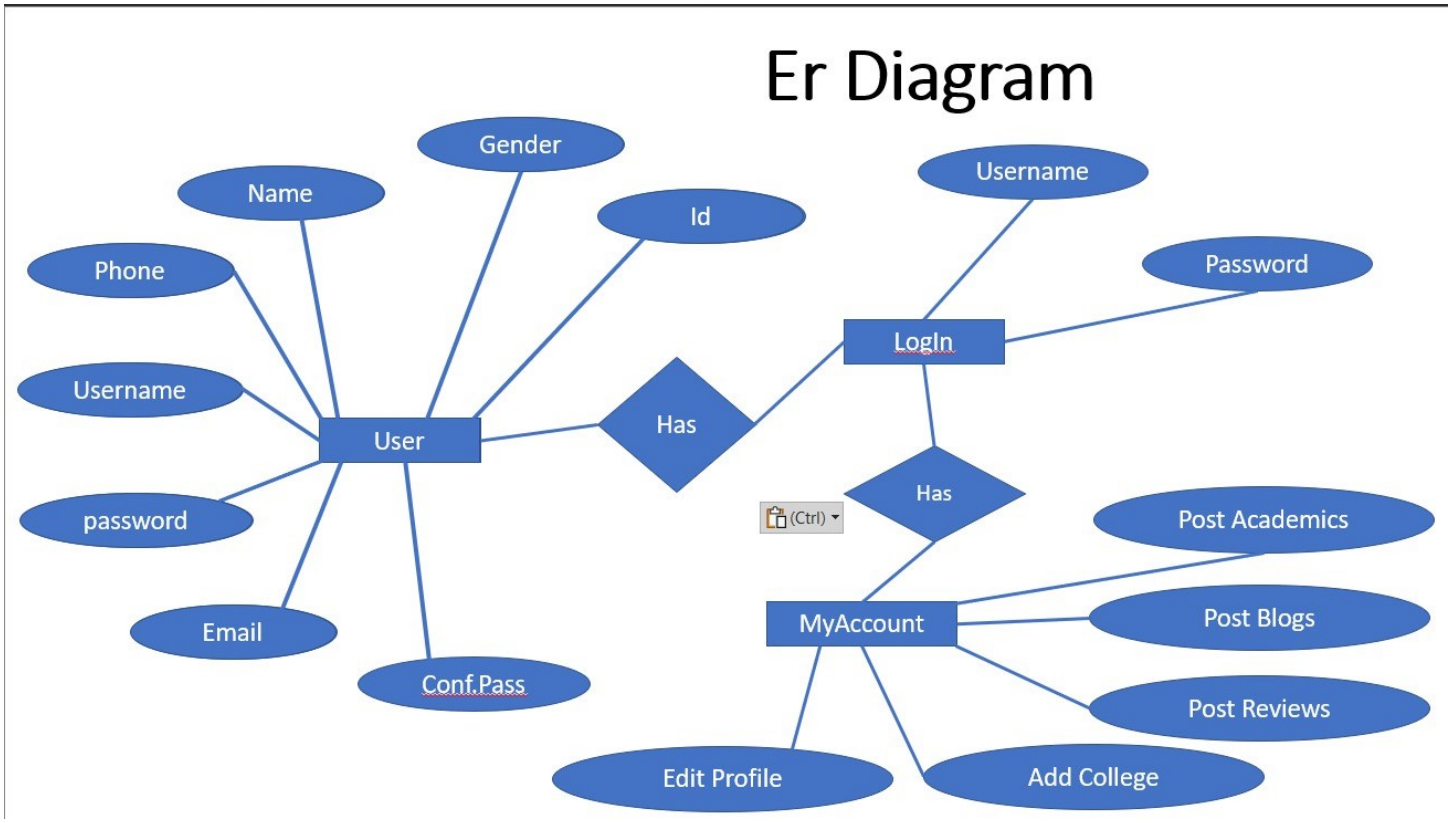
Data flow diagram level 0



3.3 Data Flow Diagram (Level 1)



3.4 ER Diagram



4. Implementation

⇨ index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import { BrowserRouter as Router } from 'react-router-dom';
import 'bootstrap/dist/css/bootstrap.min.css';
import App from './App';
import './layout.css';
import './pages.css';
import './mediaqueries.css';
import '../src/newspage.css';

const root = ReactDOM.createRoot(document.getElementById("root"));
root.render(
  <React.StrictMode>
    <Router>
      <App />
    </Router>
  </React.StrictMode>
);
```

⇨ App.js

```
import React, { useReducer, useEffect } from 'react';
import { Routes, Route, useNavigate, useLocation } from 'react-router-dom';
import { reducer, initialState } from './reducer/Reducer';
import axios from 'axios';
import { ToastContainer, toast } from 'react-toastify';
import 'react-toastify/dist/ReactToastify.css';
import Template from './components/Template';
import Home from './components/Home';
import SearchCollege from './components/SearchCollege';
import Academics from './components/Academics';
import Blogs from './components/Blogs';
import UserAccount from './components/UserAccount';
import Error from './components/Error';
const API = 'http://127.0.0.1:8000';
export const SiteContext = React.createContext();
export const SignOut = React.createContext();

let path;
const getPath = data => {
  path = data;
}

const Routing = () => {
  return (
```

```
    <Routes>
      <Route path='/' element={<Template data={getPath} />}>
        <Route index element={<Home />} />
        <Route path='searchcolleges' element={<SearchCollege />} />
        <Route path='academics' element={<Academics />} />
        <Route path='blogs' element={<Blogs />} />
        <Route path='myaccount' element={<UserAccount />} />
      </Route>
      <Route path='*' element={<Error />} />
    </Routes>
  )
}

const App = () => {
  const [state, dispatch] = useReducer(reducer, initialState);
  const navigate = useNavigate();
  const { pathname } = useLocation();

  const handleSignOut = () => {
    localStorage.removeItem('jwtoken');
    setTimeout(() => {
      dispatch({ type: 'SWITCH', payload: false });
      if (pathname === '/') {
        navigate('/searchcolleges', { replace: true });
        path('Search College');
      }
      else {
        navigate('/', { replace: true });
        path('Home');
      }
    }, 2000);
    toast.success('Signed out', {
      position: "top-center",
      autoClose: 2000,
      hideProgressBar: false,
      closeOnClick: true,
      pauseOnHover: true,
      draggable: true,
      progress: undefined,
      theme: "light",
    });
  }

  const isAuthenticate = async url => {
    const token = localStorage.getItem('jwtoken');
    try {
      if (token != null) {
        const res = await axios.post(url, { token });
        if (res) dispatch({ type: 'SWITCH', payload: true });
      }
    } catch (err) {
      console.log(err.response.data);
```

```
    }
  }

  useEffect(() => {
    isAuthenticate(`${API}/confidential`);
  })

  return (
    <>
      <SiteContext.Provider value={{ state, dispatch }}>
        <SignOut.Provider value={handleSignOut}>
          <Routing />
        </SignOut.Provider>
      </SiteContext.Provider>
      <ToastContainer
        position="top-center"
        autoClose={2000}
        hideProgressBar={false}
        newestOnTop={false}
        closeOnClick
        rtl={false}
        pauseOnFocusLoss
        draggable
        pauseOnHover
        theme="light"
      />
    </>
  )
}

export default App;
```

⇨ Template.jsx

```
import React, { useState } from 'react';
import { Outlet, useLocation } from 'react-router-dom';
import NavBar from './NavBar';
import SideBar from './SideBar';
import SideBarIcon from './SideBarIcon';
import Footer from './Footer';
const Nav2Data = React.createContext();

const Template = ({ data }) => {
  const [sidebarData, setSidebarData] = useState([]);
  const [value, setValue] = useState(false);
  const { pathname } = useLocation();
  const [nav2Val, setNav2Val] = useState(() => {
    if (pathname === '/') return 'Home';
    else if (pathname === '/searchcolleges') return 'Search College'
    else if (pathname === '/academics') return pathname.charAt(1).toUpperCase() +
pathname.slice(2);
```

```
    else if (pathname === '/blogs') return pathname.charAt(1).toUpperCase() +
pathname.slice(2);
    else if (pathname === '/myaccount') return 'My Account';
  });

  const getData = data => {
    setSidebarData(data);
  }

  const setVal = () => {
    if (value === false) setValue(true);
    else setValue(false);
  }

  const nav2Data = data => {
    setNav2Val(data);
  }

  data(nav2Data);

  return (
    <>
      <div className="main-container">
        <Nav2Data.Provider value={nav2Data}>
          <NavBar />
          <div className="main-content">
            <SideBar setData={sidebarData} val={value} setVal={setVal} />
            <SideBarIcon setData={setVal} val={value} navValue={nav2Val} />
            <Outlet context={{ key: getData }} />
          </div>
          <Footer />
        </Nav2Data.Provider>
      </div>
    </>
  )
}

export default Template;
export { Nav2Data };
```

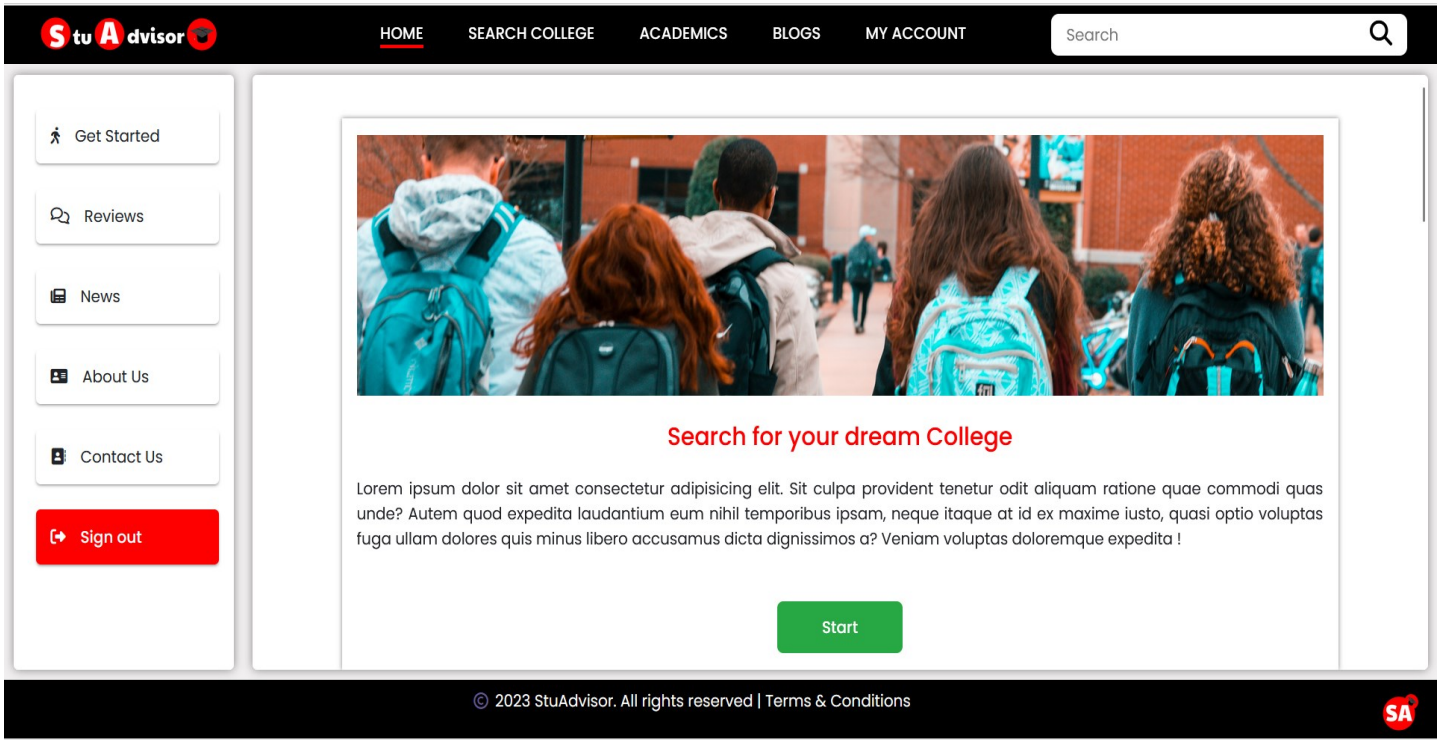
⇨ Home.jsx

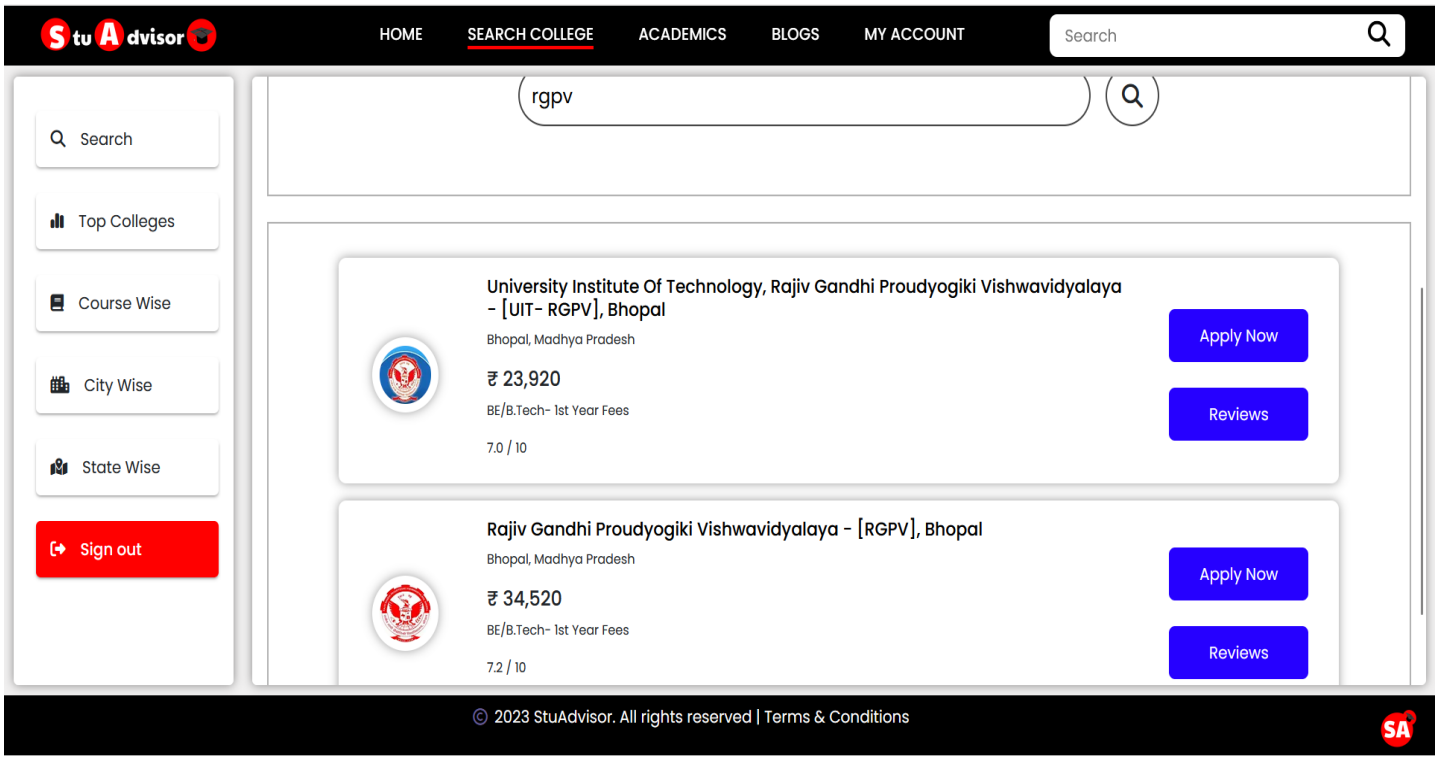
```
import React, { useEffect, useState, useContext } from 'react';
import { useOutletContext } from 'react-router-dom';
import axios from 'axios';
import { SignOut, SiteContext } from '../App';
import SignIn from './SignIn';
import SignUp from './SignUp';
import GetStarted from './GetStarted';
import News from './News';
import AboutUs from './AboutUs';
```

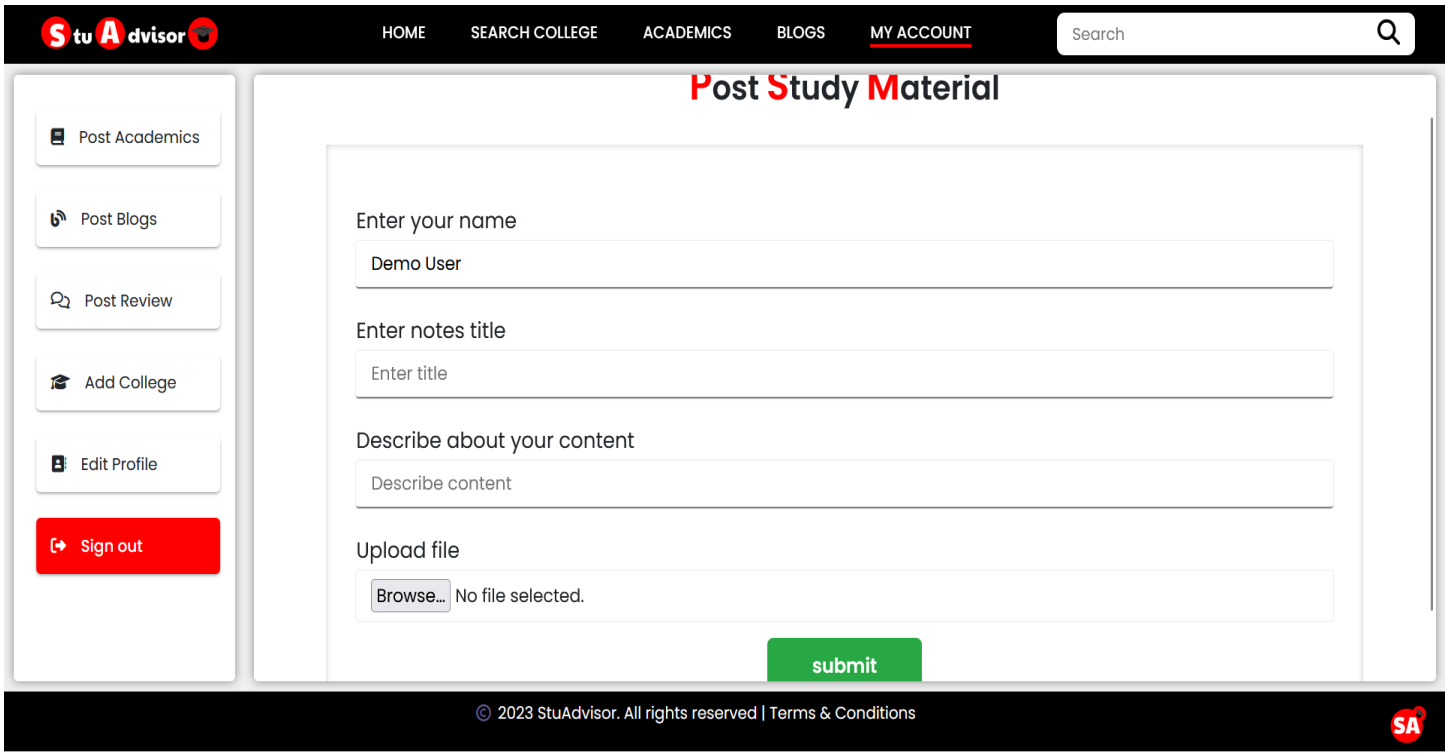
```
import ContactUs from './ContactUs';
import Reviews from './Reviews';
const API = 'http://127.0.0.1:8000';

const Home = () => {
  const { state } = useContext(SiteContext);
  const handleSignOut = useContext(SignOut);
  const [data, setData] = useState(0);
  let { key } = useOutletContext();
  const [review, setReview] = useState([]);
  const [news, setNews] = useState([]);

  const menuClick = id => {
    setData(id);
  }

  const getReview = async url => {
    try {
      const res = await axios.post(url, { type: 'get' });
      setReview(res.data.reviews);
    } catch (err) {
      console.log(err.response.data.message);
    }
  }

  const getNews = async url => {
    try {
      const res = await axios.get(url);
      setNews(res.data);
    } catch (err) {
      console.log(err.response.data.message);
    }
  }

  const homeData = [{ text: 'Get Started', icon: <i className="i-tag fa-solid fa-person-walking"></i>, click: menuClick }, { text: 'Reviews', icon: <i className="i-tag fa-regular fa-comments"></i>, click: menuClick }, { text: 'News', icon: <i className="i-tag fa-solid fa-newspaper"></i>, click: menuClick }, { text: 'About Us', icon: <i className="i-tag fa-solid fa-address-card"></i>, click: menuClick }, { text: 'Contact Us', icon: <i className="i-tag fa-solid fa-address-book"></i>, click: menuClick }]

  const homeMenu = [...homeData, { text: 'Sign in', icon: <i className="i-tag fa-solid fa-door-open"></i>, click: menuClick }, { text: 'Sign up', icon: <i className="i-tag fa-solid fa-user-plus"></i>, click: menuClick }];

  const homeMenu2 = [...homeData, { text: 'Sign out', icon: <i className="i-tag fa-solid fa-right-from-bracket"></i>, click: handleSignOut }];

  useEffect(() => {
    document.title = 'Home';
    if (state) key(homeMenu2);
    else key(homeMenu);
```

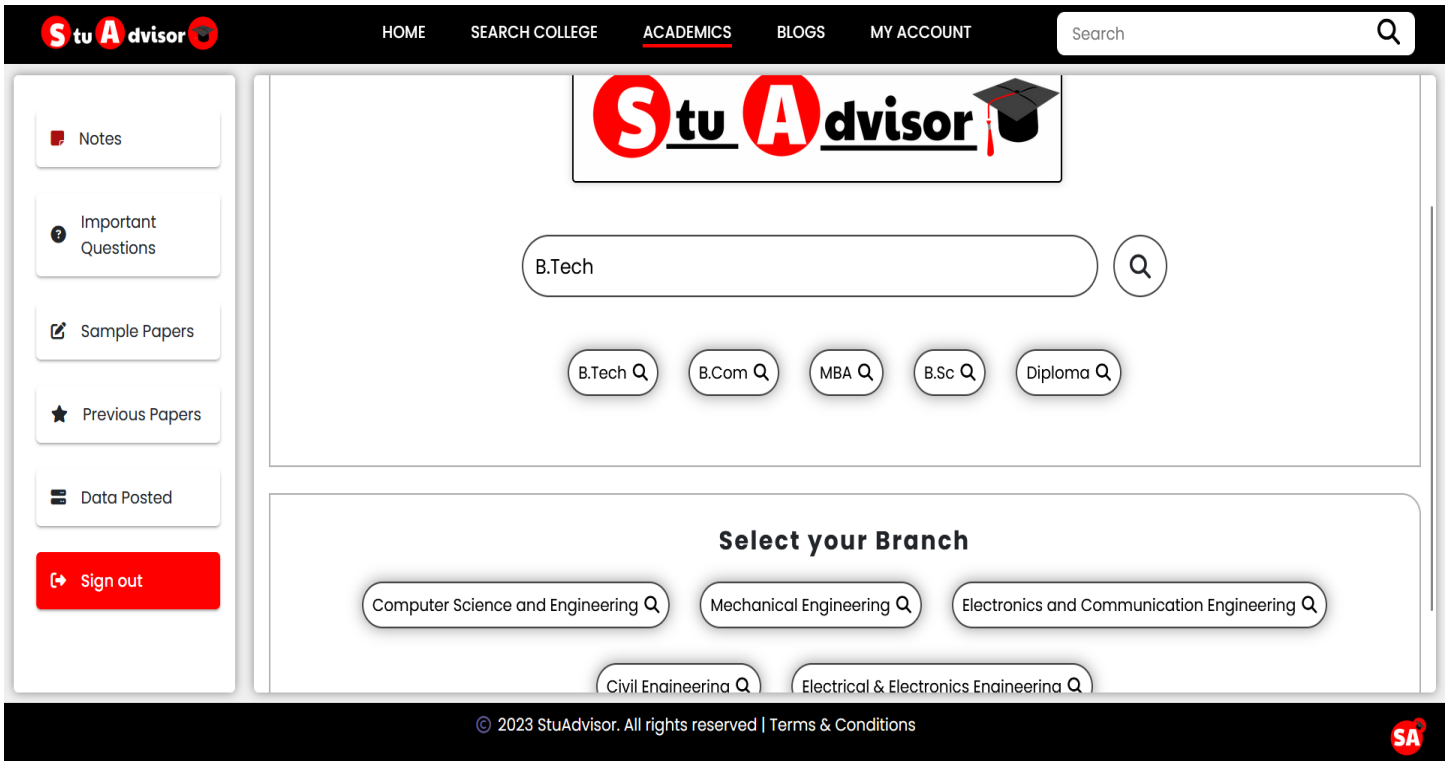
```
    getReview(`${API}/reviews`);
    getNews(`${API}/news`);
    // eslint-disable-next-line
  }, [])

  return (
    <>
      <div className="main-item main-right">
        {data === 0 ? <GetStarted /> : null}
        {data === 1 ? <Reviews data={review} /> : null}
        {data === 2 ? <News data={news} /> : null}
        {data === 3 ? <AboutUs /> : null}
        {data === 4 ? <ContactUs /> : null}
        {data === 5 ? <SignIn /> : null}
        {data === 6 ? <SignUp /> : null}
      </div>
    </>
  )
}

export default Home;
```

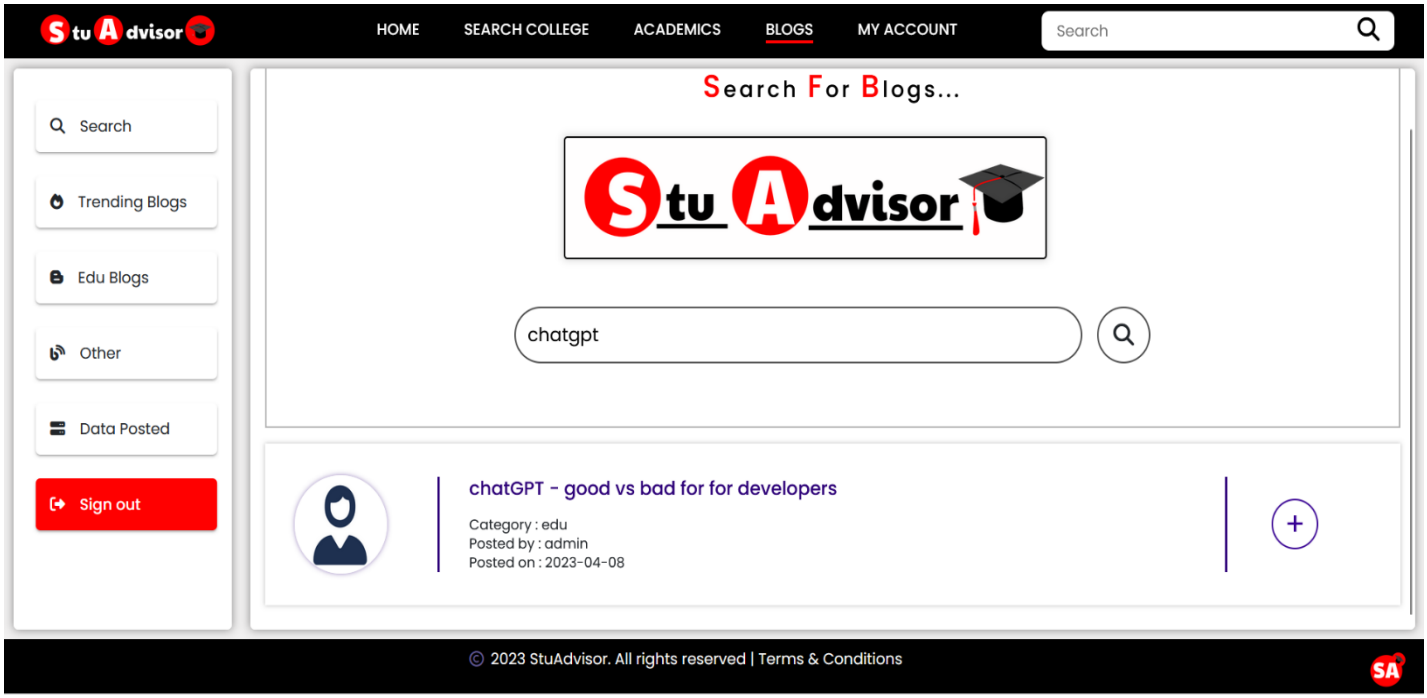
5. Snapshots

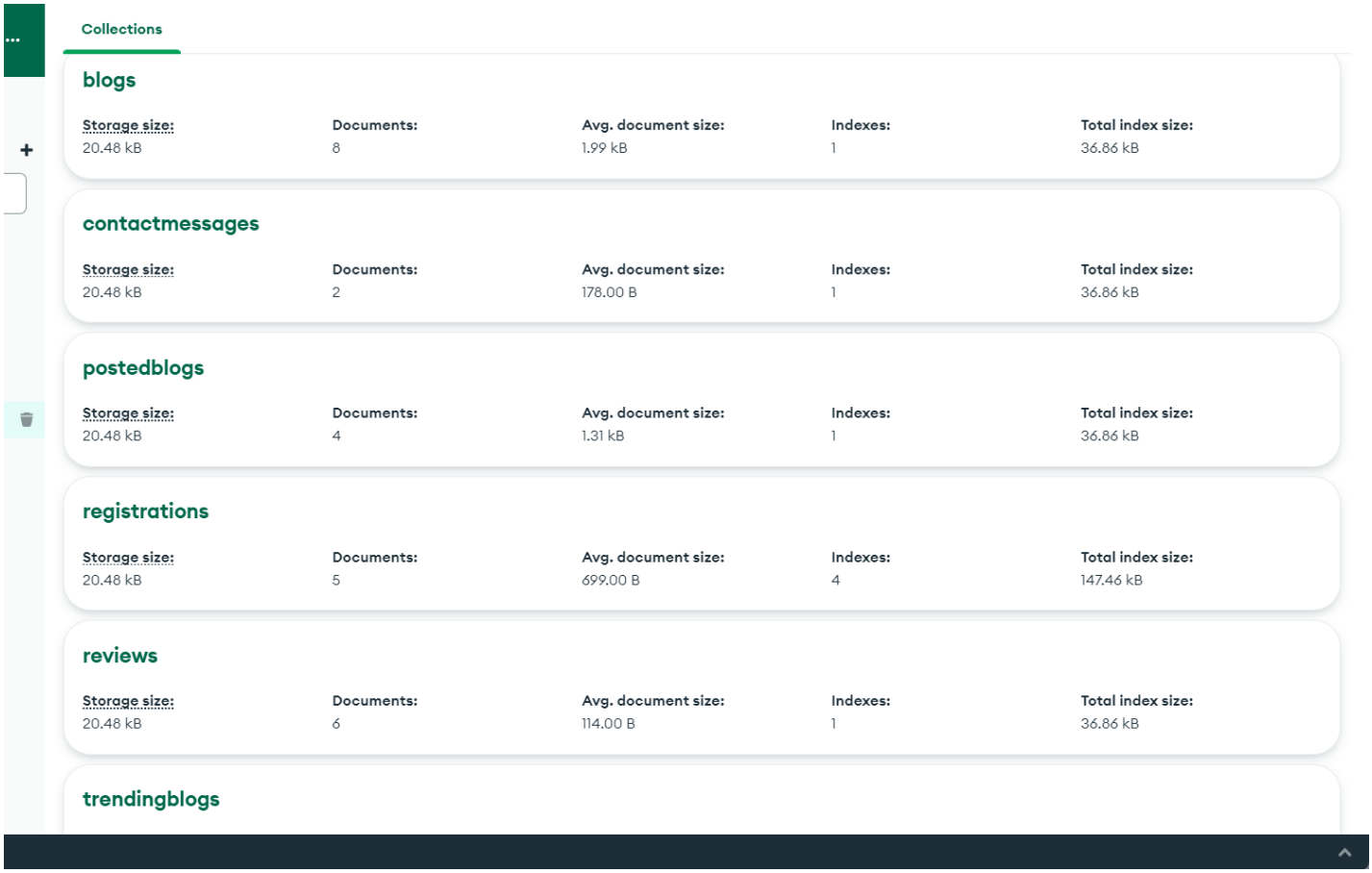

StuAdvisor
HOME
SEARCH COLLEGE
ACADEMICS
BLOGS
MY ACCOUNT
Search
Get Started
Reviews
News
About Us
Contact Us
Sign out
Search for your dream College
Lorem ipsum dolor sit amet consectetur adipisicing elit. Sit culpa provident tenetur odit aliquam ratione quae commodi quas unde? Autem quod expedita laudantium eum nihil temporibus ipsam, neque itaque at id ex maxime iusto, quasi optio voluptas fuga ullam dolores quis minus libero accusamus dicta dignissimos a? Veniam voluptas doloremque expedita !
Start
© 2023 StuAdvisor. All rights reserved | Terms & Conditions


StuAdvisor
HOME
SEARCH COLLEGE
ACADEMICS
BLOGS
MY ACCOUNT
Search
Search
Top Colleges
Course Wise
City Wise
State Wise
Sign out
rgpv
University Institute Of Technology, Rajiv Gandhi Proudyogiki Vishwavidyalaya - [UIT- RGPV], Bhopal
Bhopal, Madhya Pradesh
₹ 23,920
BE/B.Tech- 1st Year Fees
7.0 / 10
Apply Now
Reviews
Rajiv Gandhi Proudyogiki Vishwavidyalaya - [RGPV], Bhopal
Bhopal, Madhya Pradesh
₹ 34,520
BE/B.Tech- 1st Year Fees
7.2 / 10
Apply Now
Reviews
© 2023 StuAdvisor. All rights reserved | Terms & Conditions


StuAdvisor
HOME
SEARCH COLLEGE
ACADEMICS
BLOGS
MY ACCOUNT
Search
Post Academics
Post Blogs
Post Review
Add College
Edit Profile
Sign out
Post Study Material
Enter your name
Demo User
Enter notes title
Enter title
Describe about your content
Describe content
Upload file
Browse...
No file selected.
submit
© 2023 StuAdvisor. All rights reserved | Terms & Conditions


StuAdvisor
HOME
SEARCH COLLEGE
ACADEMICS
BLOGS
MY ACCOUNT
Search
Notes
Important Questions
Sample Papers
Previous Papers
Data Posted
Sign out
StuAdvisor
B.Tech
B.Tech
B.Com
MBA
B.Sc
Diploma
Select your Branch
Computer Science and Engineering
Mechanical Engineering
Electronics and Communication Engineering
Civil Engineering
Electrical & Electronics Engineering
© 2023 StuAdvisor. All rights reserved | Terms & Conditions


HOME
SEARCH COLLEGE
ACADEMICS
BLOGS
MY ACCOUNT
Search
Search
Trending Blogs
Edu Blogs
Other
Data Posted
Sign out
Search For Blogs...
chatgpt
chatGPT - good vs bad for for developers
Category : edu
Posted by : admin
Posted on : 2023-04-08
© 2023 StuAdvisor. All rights reserved | Terms & Conditions


Collections
blogs
Storage size: 20.48 kB
Documents: 8
Avg. document size: 1.99 kB
Indexes: 1
Total index size: 36.86 kB
contactmessages
Storage size: 20.48 kB
Documents: 2
Avg. document size: 178.00 B
Indexes: 1
Total index size: 36.86 kB
postedblogs
Storage size: 20.48 kB
Documents: 4
Avg. document size: 1.31 kB
Indexes: 1
Total index size: 36.86 kB
registrations
Storage size: 20.48 kB
Documents: 5
Avg. document size: 699.00 B
Indexes: 4
Total index size: 147.46 kB
reviews
Storage size: 20.48 kB
Documents: 6
Avg. document size: 114.00 B
Indexes: 1
Total index size: 36.86 kB
trendingblogs

6. Testing

6.1 Software Testing

We have used ‘Black Box Testing’ for checking the working of the project after every module development. After performing Black Box Testing, we got some loopholes in the project like user can fill too much extra information in the input field which will be dangerous for our database in the future and then for fixing this, we used validation on the input fields and set the limits of the input fields according to the data provided by the user. We also get some lack of features in the functionalities while performing black box testing and then later we fixed them all.

We also used ‘White Box Testing’ inside our project while developing the modules. We checked proper working status of our code by tracing the error and bugs we got in our project. While using ‘White Box Testing’, we fixed our project internally and also we have checked optimizations of our code while performing ‘White Box Testing’.

And at the end, we used ‘Integration Testing’ for combining the modules with each other and for checking the proper communication between them all. In ‘Integration Testing’, we solved the problems we was facing to create communication between all the modules.

7. Conclusion

Our project “StuAdvisor’ is going to help students in many ways. We will provide them genuine guide for their studies. Students don’t need to pay anywhere to get study related stuff and we will help them to find good colleges for them without the involvement of any third person. This will save their money, they don’t need to pay anything to other people for getting admission.

We will provide them blogs for reading and clearing their subject related doubts. They can also post blogs to help others. Teachers can also register on our web application and can post material for the students easily.

Users can register on our website and can maintain their own profile for posting on ‘StuAdvisor’. Users can also help us by sending the data about colleges which is not available on our platform. After verification of that data and contacting to that institute/college, we will show them on our website.

In future, we are can improve the project by adding more functionalities like users can share study related videos to other members and others will be able to comment on that video and many more. In future, we can give more help to the students by providing them internship/jobs details personally according to their field of interest. We can also tie up with the colleges for proving more help to the students and students will get admission at very affordable price and they can also get study materials of other colleges on StuAdvisor.

8. Bibliography

1. google.com

2. stackoverflow.com

3. medium.com

4. freecodecamp.com

5. youtube.com