# **MXIANS’ PORTAL**

SUBMITTED BY

**DHANU PRIYAN M (18MX102)**

**VAISHNA VIJAYAN (18MX220)**

**USMAN HILMI SA (19MX162)**

**DISSERTATION SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE DEGREE OF**

**MASTER OF COMPUTER APPLICATIONS**

**ANNA UNIVERSITY**

****

**APRIL 2020**

**DEPARTMENT OF COMPUTER APPLICATIONS**

**PSG COLLEGE OF TECHNOLOGY**

**(Autonomous Institution)**

**COIMBATORE – 641 004**

# **PSG COLLEGE OF TECHNOLOGY**

**(Autonomous Institution)**

**COIMBATORE-641 004**

**Mini Project I**

# **MXIANS’ PORTAL**

**Bonafide record of work done by**

**DHANU PRIYAN M (18MX102)**

**VAISHNA VIJAYAN (18MX220)**

**USMAN HILMI SA (19MX162)**

**Dissertation submitted in partial fulfilment of the requirements for the degree of**

## **MASTER OF COMPUTER APPLICATIONS**

**ANNA UNIVERSITY**

**APRIL 2020**

**Faculty guide**

**……………………………**

**MR. C. SUNDAR**

**ACKNOWLEDGEMENT**

We indulged to thank **Dr. K. PRAKASAN**, Principal In-Charge, PSG College of Technology, Coimbatore for permitting us to take up this project.

We extend our earnest gratitude to **Dr. A. CHITRA**, Professor and Head, Department of Computer Applications, PSG College of Technology, Coimbatore, for her support, encouragement and above all her ardent motivation.

We are grateful to thank **Dr. R. MANAVALAN**, Associate Professor, Program coordinator, Department of Computer Applications, for his full-fledged support and guidance.

We would like to express our gratitude to our guide **Mr. C. SUNDAR**, Assistant Professor, Department of Computer Applications, PSG College of Technology, for his valuable suggestions, encouragement and guidance in the process of completion of the project.

We would like to express our gratitude to our tutor **Dr. S. BHAMA**, Assistant Professor, Department of Computer Applications, PSG College of Technology, for her support.

Finally, we express our gratefulness to all the faculty members of Department of Computer Applications, PSG College of Technology, and our friends for their encouragement and support.

**i**

**SYNOPSIS**

**ii**

**CONTENTS**

**CHAPTER Page No**

**ACKNOWLEDGEMENT** i

**SYNOPSIS** ii

**1. INTRODUCTION**  1

**1.1** Project Overview 1

**1.2** Development Environment 2

**1.3** Technology Overview 3

**2. SYSTEM ANALYSIS**  9

**2.1** Existing System 9

**2.2** Proposed System 9

**2.3** System Requirements 10

**2.4** System Flow/UML Diagram 12

**3. SYSTEM DESIGN AND IMPLEMENTATION**  13

**3.1** MVC Architecture 13

**3.2**  System Implementation 15

**4. TESTING**  30

**5. CONCLUSION**  40

**BIBLIOGRAPHY**

**CHAPTER 1**

**INTRODUCTION**

**1.1 Project Overview**

The objective of this project is to connect the entire MCA department of PSG College of Technology. This project aims to create a portal for faculty, students and alumni to establish a communication between them.

**Faculty can perform the following tasks,**

* Each faculty can have their own profile.
* Faculty can post any information publicly and within a class also.

**Student can perform the following tasks,**

* Each student can have their own profile.
* Each Student can enter/edit their personal details, achievements, etc.
* Students can communicate with other students individually, within their class and as a whole.

**Alumni can perform the following tasks,**

* Alumni can have their own profile.Alumni can enter/edit their details.
* Alumni can communicate with the students/staff individually and with students and as a whole.

**1.2 Development Environment**

**Hardware Requirements**

Processor : Intel Core I5

Main Memory : 8GB RAM

Secondary Memory : 1TB HDD

**Software Requirements**

Operation System : Windows 10

Developing Language : HTML, CSS, JavaScript, TypeScript

Tools & Technologies : MongoDB v4.13.20 ,NodeJS

(v12.3.1), ExpressJS (v4.17.1)

Angular 9, Socket.IO ,Visual Studio Code,

Postman.

**1.3 Technology Overview**

The Technology used here is MEAN Stack. The Various tools and technologies used in project tenure are explained here.

**MEAN Stack**

* M - MongoDB
* E - ExpressJS
* A - AngularJS
* N - NodeJS

**1.3.1 MongoDB:**

MongoDB is an open-source document-based database management tool that stores data in JSON-like formats. It is a highly scalable, flexible, and distributed NoSQL database. Being a NoSQL tool means that it does not use the usual rows and columns. It is an architecture that is built on collections and documents. The basic unit of data in this database consists of a set of key–value pairs. It allows documents to have different fields and structures. This database uses a document storage format called BSON which is a binary style of [JSON](https://intellipaat.com/blog/processing-json-data-in-real-time-streaming-using-storm-kafka/) documents. The data model that MongoDB follows is a highly elastic one that allows to combine and store data of multivariate types without having to compromise on the powerful indexing options, data access, and validation rules. MongoDB provides high performance. Most of the operations in MongoDB are faster compared to relational databases.

**1.3.2 ExpressJs:**

Express.js is a Node js web application server framework, which is designed for building single-page, multi-page, and hybrid web applications. It allows to set up middlewares to respond to HTTP requests. It defines a routing table which is used to perform different actions based on HTTP method and URL. It allows to dynamically render HTML Pages based on passing arguments to templates. Its Open Source ,free, easy to extend and very performant. There are also lots and lots of pre-built packages which are used to perform all kinds of operations.

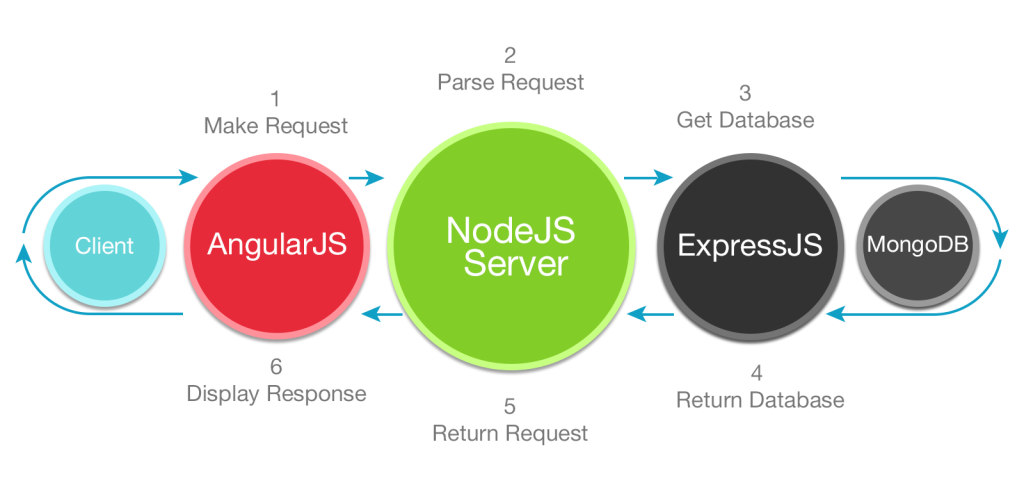
**1.3.3 Angular Js:**

It is one of the open-source, front-end, JavaScript-based frameworks widely used in creating single page applications on the client side. It is capable of transferring the entire content directly from the server to the browser, loading all the pages at the same time. The page does not reload if a link is clicked upon; instead, the sections within the page are updated instantly. AngularJS enables faster loading of pages as there is less load on the server. The two-way data binding feature saves the burden of writing numerous codes. AngularJS works on MVC architecture. Uses HTML as template language and lets to extend HTML's syntax to express your application's components clearly and succienctly.

**1.3.4 Node Js:**

Node.js is a cross-platform runtime environment and library for running JavaScript applications outside the browser. It is used for creating server-side and networking web applications. It is open source and free to use. Node.js also provides a rich library of various JavaScript modules to simplify the development of web applications. Node.js is mostly used to run real-time server applications. Over the years, most of the applications were based on a stateless request-response framework. In these sorts of applications, it is up to the developer to ensure the right code was put in place to ensure the state of web session was maintained while the user was working with the system. But with Node.js web applications, you can now work in real-time and have a 2-way communication. The state is maintained, and either the client or server can start the communication.

**MEAN Stack Workflow**.



**Fig 1.1 MEAN Stack.**

The **Fig 1.1** explains the workflow of MEAN Stack.

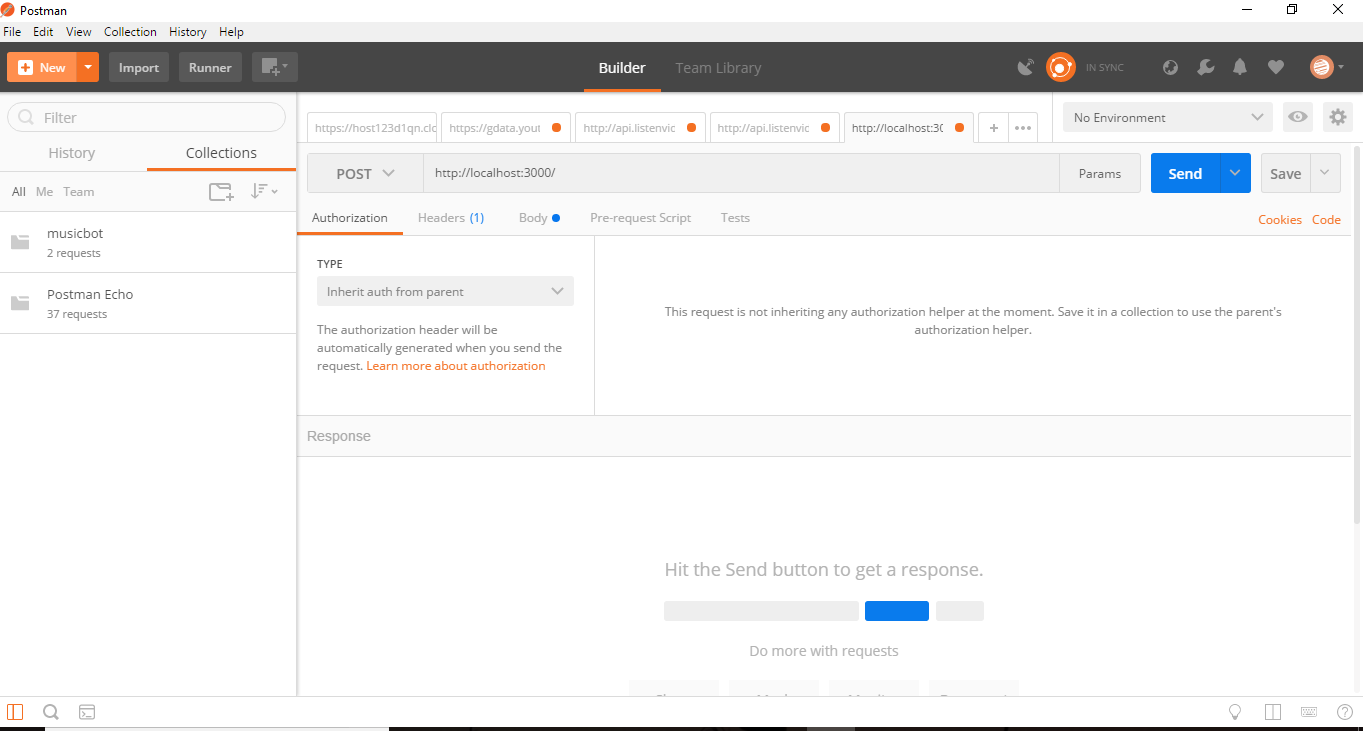
Client make request via AngularJS and the client’s request get parsed in NodeJS server and then it is sent via backend framework Express JS for getting database from MongoDB and it returns back the database to NodeJS server. Then NodeJS server returns the request as response and that response displayed to client using AngularJS.

**1.3.5 Socket.IO:**

Socket.IO is a [JavaScript](https://en.wikipedia.org/wiki/JavaScript) library for realtime [web applications](https://en.wikipedia.org/wiki/Web_application). It enables realtime, bi-directional communication between web clients and servers. It has two parts: a [client-side](https://en.wikipedia.org/wiki/Client-side) library that runs in the [browser](https://en.wikipedia.org/wiki/Web_browser), and a [server-side](https://en.wikipedia.org/wiki/Server-side) library for [Node.js](https://en.wikipedia.org/wiki/Node.js). Both components have a nearly identical [API](https://en.wikipedia.org/wiki/Application_programming_interface). Like [Node.js](https://en.wikipedia.org/wiki/Node.js), it is [event-driven](https://en.wikipedia.org/wiki/Event-driven_architecture). Socket.IO primarily uses the [WebSocket](https://en.wikipedia.org/wiki/WebSocket) protocol with polling as a fallback option, while providing the same interface. Although it can be used as simply a [wrapper](https://en.wikipedia.org/wiki/Wrapper_library) for WebSocket, it provides many more features, including broadcasting to multiple sockets, storing data associated with each client, and [asynchronous I/O](https://en.wikipedia.org/wiki/Asynchronous_I/O).

It can be installed with the [npm](https://en.wikipedia.org/wiki/Npm_(software)" \o "Npm (software)) tool.

**1.3.6 Postman:**

 Postman is an API(Application Programming Interface) development tool which helps to build, test and modify APIs. Almost any functionality that could be needed by any developer is encapsulated in this tool. It has the ability to make various types of HTTP requests(GET, POST, PUT,DELETE), saving environments for later use, converting the API to code for various languages(like JavaScript, Python).The Postman software is used to send and receive requests, POST data to the server.

**Fig 1.2 Postman.**

The above figure 1.2 contains Postman API Tool.

* The longest middle input field that looks something like a search bar is where the URL that the user want to GET or POST or DELETE,etc. is fed.
* Just to the left of it, is a drop down button which has all the various HTTP methods as options. To make POST to the URL that the user have specified, select POST.
* To the right of it is the params button. If the user click on it, a new interface will appear. Params are basically the data that the user want to send to the server with the request.
* To the left of this button is the **Send** button which is used in sending the request to the server or the app in this case.

**1.3.7 Visual Studio Code:** Visual Studio Code is a [source-code editor](https://en.wikipedia.org/wiki/Source-code_editor) developed by [Microsoft](https://en.wikipedia.org/wiki/Microsoft) for [Windows](https://en.wikipedia.org/wiki/Windows), [Linux](https://en.wikipedia.org/wiki/Linux) and [macOS](https://en.wikipedia.org/wiki/MacOS).It can be used with a variety of programming languages, including [Java](https://en.wikipedia.org/wiki/Java_(programming_language)), [JavaScript](https://en.wikipedia.org/wiki/JavaScript), [Go](https://en.wikipedia.org/wiki/Go_(programming_language)), [Node.js](https://en.wikipedia.org/wiki/Node.js) and [C++](https://en.wikipedia.org/wiki/C%2B%2B)It includes support for [debugging](https://en.wikipedia.org/wiki/Debugging), embedded [Git](https://en.wikipedia.org/wiki/Git) control and [GitHub](https://en.wikipedia.org/wiki/GitHub), [syntax highlighting](https://en.wikipedia.org/wiki/Syntax_highlighting), [intelligent code completion](https://en.wikipedia.org/wiki/Intelligent_code_completion), [snippets](https://en.wikipedia.org/wiki/Snippet_(programming)), and [code refactoring](https://en.wikipedia.org/wiki/Code_refactoring). It is highly customizable, allowing users to change the [theme](https://en.wikipedia.org/wiki/Theme_(computing)), [keyboard shortcuts](https://en.wikipedia.org/wiki/Keyboard_shortcut), preferences, and install [extensions](https://en.wikipedia.org/wiki/Plug-in_(computing)) that add additional functionality. The source code is [free and open source](https://en.wikipedia.org/wiki/Free_and_open_source) .

**CHAPTER 2**

**SYSTEM ANALYSIS**

**2.1 Existing System**

In the Existing System,

* There is no common portal for communication and sharing information among the students, faculty and alumni.
* There is a alumni portal, where the details of the alumni are stored.
* Tutor have to collect all the students’ information manually. This leads to waste of time and data loss my occur because it is not stored in database.

**2.2 Proposed System**

In the Proposed System,

* There is a common blog like page for each class of students batchwise. So, information sharing is done by selecting the batch’s page and posting it. All posts are posted by students, faculties and alumni.
* Students can have their dashboard. There they should be able to enter all their information which are needed and they are stored in database.

**2.3 System Requirements**

This section explains about the functional and the behavioural requirements of the

Mxians’s portal.

**2.3.1 Functional Requirements**

These functional requirements are the services that the system should provide, how

the system should react upon particular inputs and how the system should behave

in particular situations.

1. Allowing the user to login on to the app.
2. Denying access to unauthorized users.
3. Allowing new users (Alumni) to register on to the app.
4. Allowing users to update their profile details.
5. Allowing users to post on the blog with title and content.
6. Providing edit and delete access to the user who created a specific post.
7. Viewing the list of the blog posted with the post creator name, date and blog posted timing.
8. Providing access to chat for each and every user.
9. Providing access to upload multimedia files .

10.Allowing the users to download uploaded multimedia files .

11.Allowing to logout from the app.

**2.3.2 Non-Functional Requirements**

This Non Functional Requirements are those constraints on the services, development process, functions and how a system is supposed to be. They are often termed as quality attributes.

**Efficiency**

The system provides efficiency and comprehensiveness. It is quite stable and operated by all naive users without any training prior.

**Response Time**

The system is interactive in very action-response of the system. Since single-page applications don’t update the entire page but only required content, they significantly improve a website’s speed.In case of good network connection the communication will be even more faster.

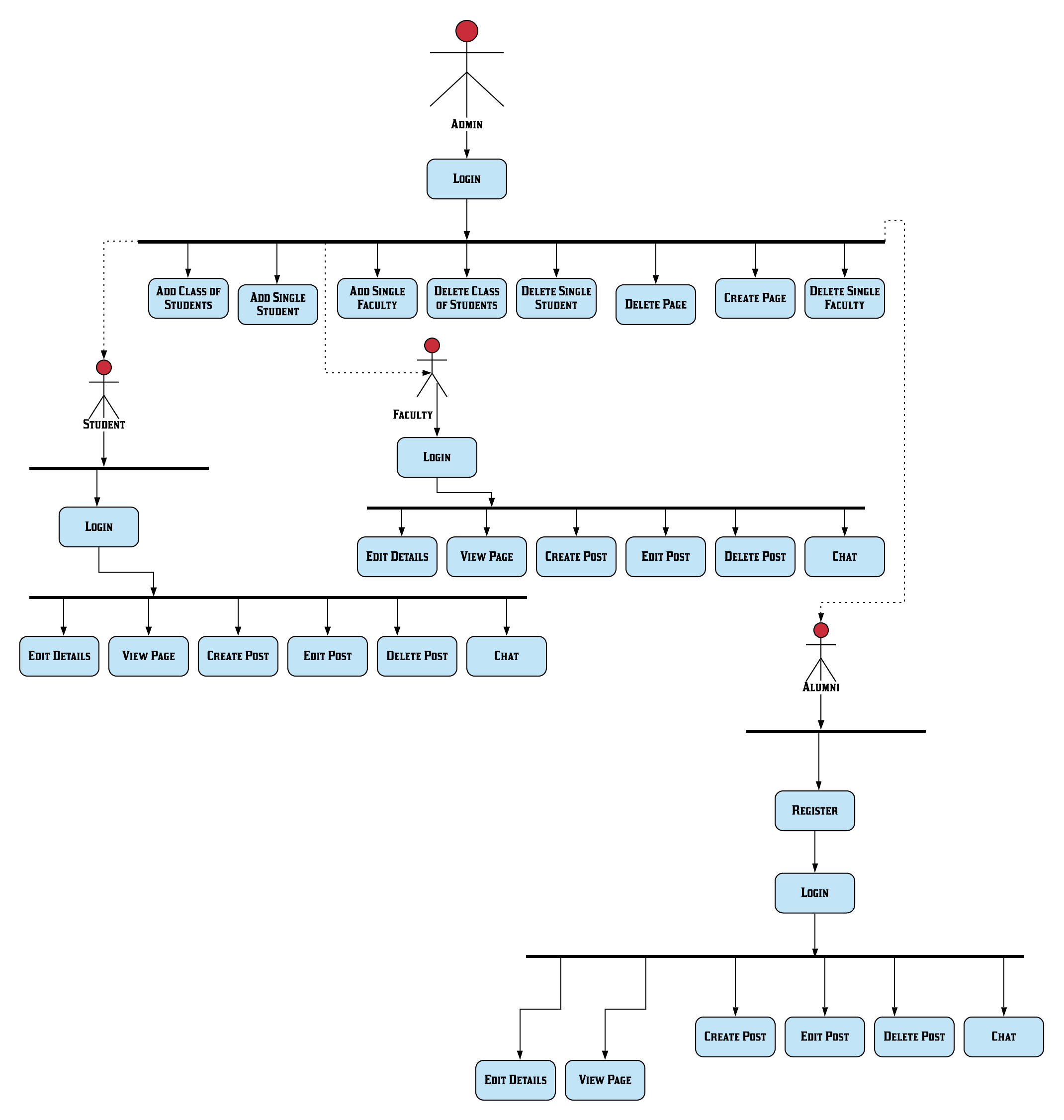
**Security**

The application is properly protected by refusing access to unauthorized users and also to illegal traversing by configuring the url .

**Availability**

The application is available all times, where the user can access it using a web browser. As the backend service is cloud storage we can retrieve anywhere from the server with the network connection.

**2.4 System Flow – Activity Diagram**

**Fig 2.1 Activity Diagram.**

The figure 2.1 is a UML Activity Diagram which explains the System Flow of this project.

Admin has the role of,

* Creating Class of Students.
* Creating Faculty.
* Creating Page.
* Deleting Class of Students.
* Deleting Faculty.
* Deleting Single Student.
* Deleting Page.

Faculty has the role of,

* Editing Details.
* Viewing the Page.
* Creating Post.
* Editing Post.
* Deleting Post..
* Chatting with Students, Faculty and Alumni.

Student has the role of,

* Editing Details.
* Viewing the Page.
* Creating Post.
* Editing Post.
* Deleting Post.
* Chatting with Students, Faculty and Alumni.

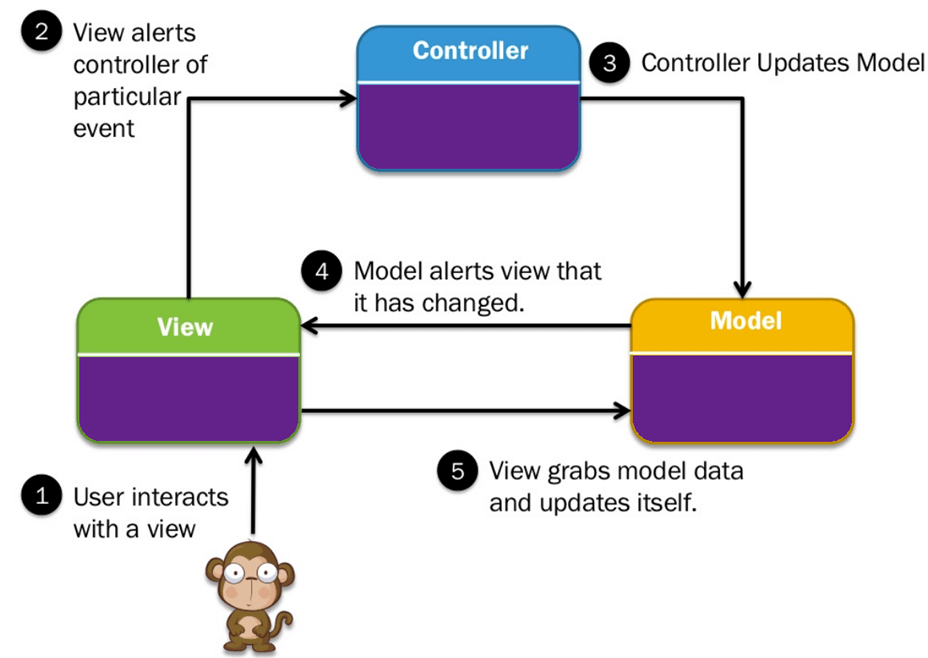
Alumni has the role of,

* Signing up.
* Editing Details.
* Viewing the Page.
* Creating Post.
* Editing Post.
* Deleting Post.
* Chatting with Students, Faculty and Alumni.

**CHAPTER 3**

**SYSTEM DESIGN AND IMPLEMENTATION**

**3.1 MVC framework**

This section tells about the design pattern which is used to develop modern user interfaces.

**Fig 3.1 MVC Framework**

The above figure 3.1 explains the working of MVC framework.

The **Model-View-Controller (MVC)** framework is an architectural pattern that separates an application into three main logical components Model, View, and Controller. Hence the abbreviation MVC. Each architecture component is built to handle specific development aspect of an application. MVC separates the business logic and presentation layer from each other.

It was traditionally used for desktop graphical user interfaces (GUIs). Nowadays, MVC architecture has become popular for designing web applications as well as mobile apps.

**Model:**

The model component stores data and its related logic. It represents data that is being transferred between controller components or any other related business logic. For example, a Controller object will retrieve the customer info from the database. It manipulates data and send back to the database or use it to render the same data.

**View:**

A View is that part of the application that represents the presentation of data. Views are created by the data collected from the model data. A view requests the model to give information so that it resents the output presentation to the user. For example, any customer view will include all the UI components like text boxes, drop downs, etc.

**Controller:**

The Controller is that part of the application that handles the user interaction. The controller interprets the mouse and keyboard inputs from the user, informing the model and the view to change as appropriate.A Controller send's commands to the model to update its state(E.g., Saving a specific document). The controller also sends commands to its associated view to change the view's presentation

**3.2 System Implementation**

**3.2.1 Database :**

**Fig 3.2** displays the list of collections in the MxiansPortal which is the Mongo Database for this project.

**Fig 3.3** displays the collection 18mxians which contains the students of the batch 18mxians.

**Fig 3.4** displays the collection 18mxiansGroup which contains all the chat messages which have been sent to the students of the batch 18mxians.

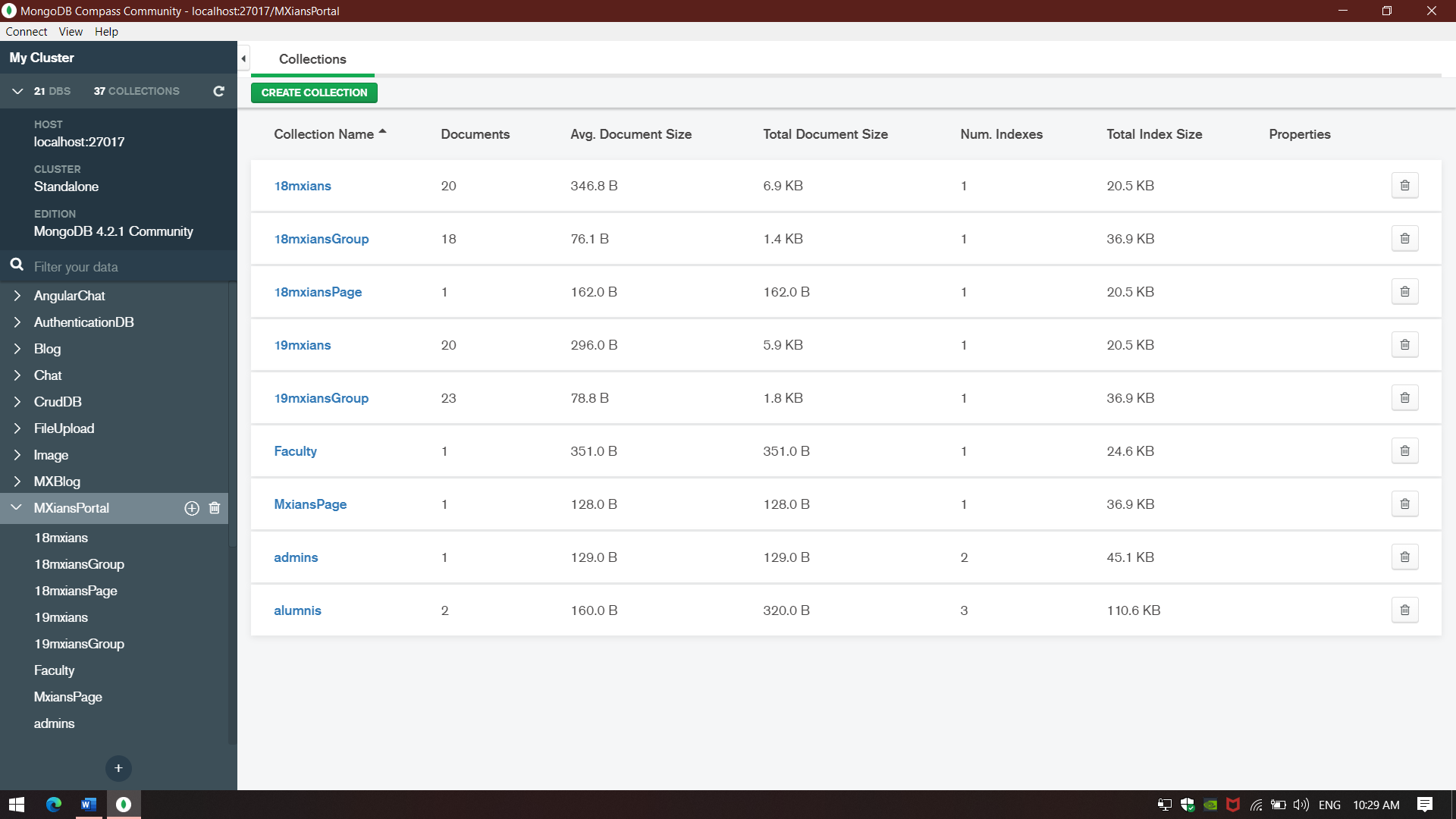
**Fig 3.5** displays the collection 18mxiansPage which contains all the posts which have been posted to the students of the batch 18mxians.

**Fig 3.6** displays the collection MxiansPage which contains all the posts which have be posted to the common page MxiansPage which is used as a public page for all the batches of Mxians.

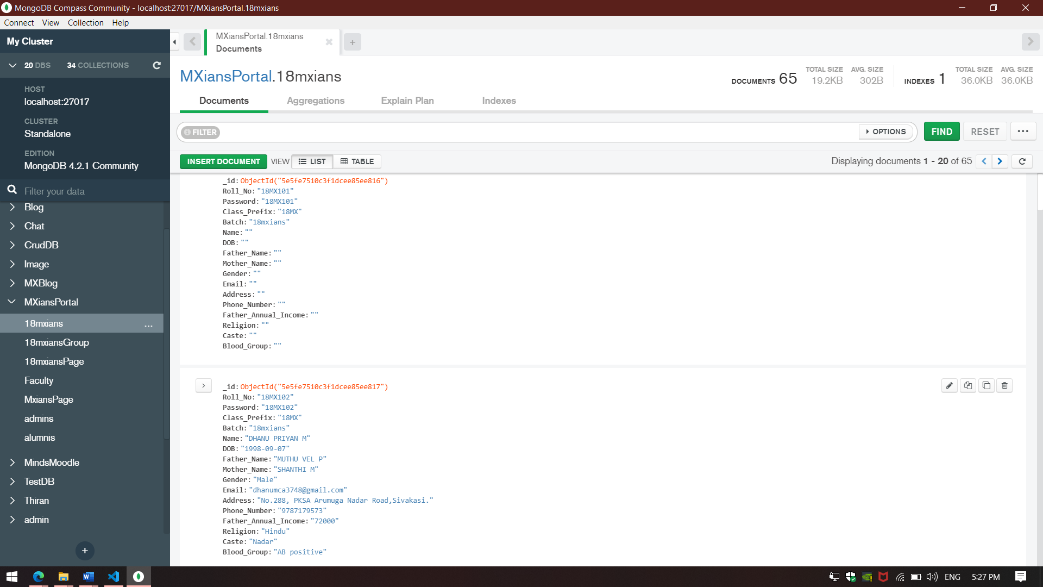
**Fig 3.7** displays the collection Faculty which contains the faculty of the department of MCA of PSG College of Technology.

**Fig 3.8** displays the collection alumnis which contains all the alumni who are registered in this website.

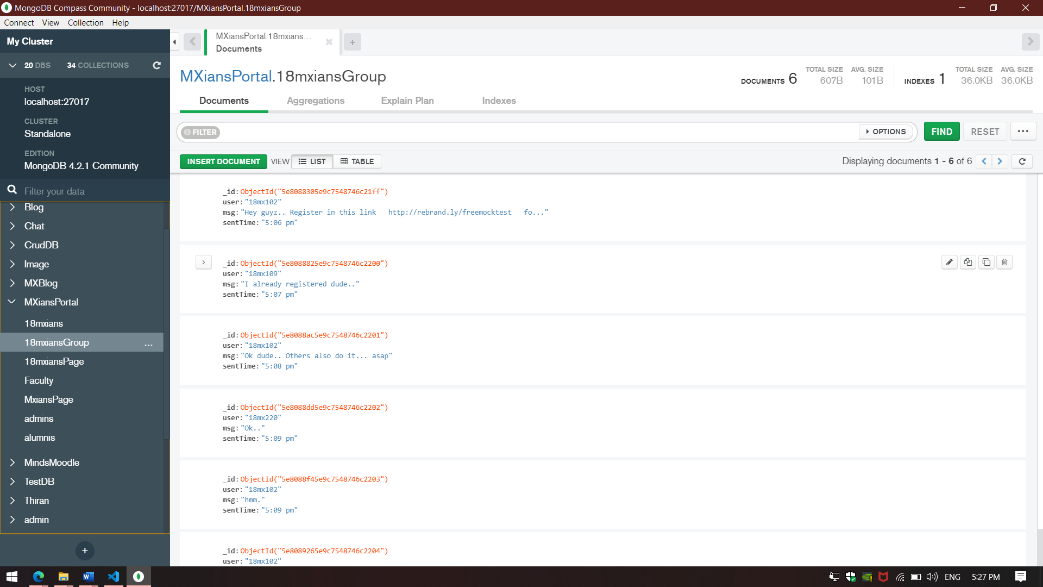
**Fig 3.9** displays the collection admin which contain all the admins for this website.



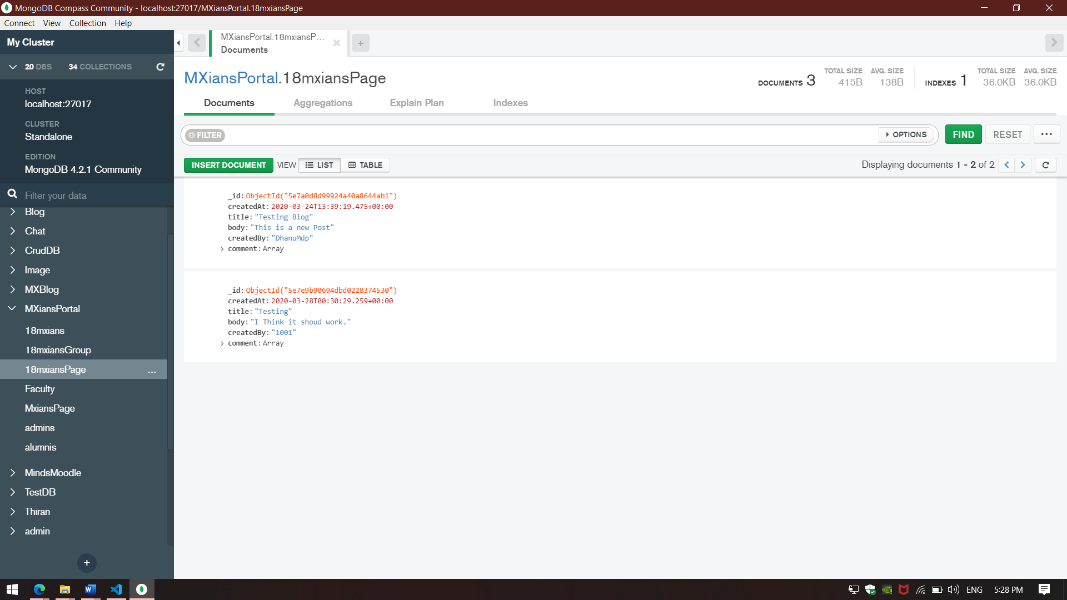
**Fig 3.2 Database.**



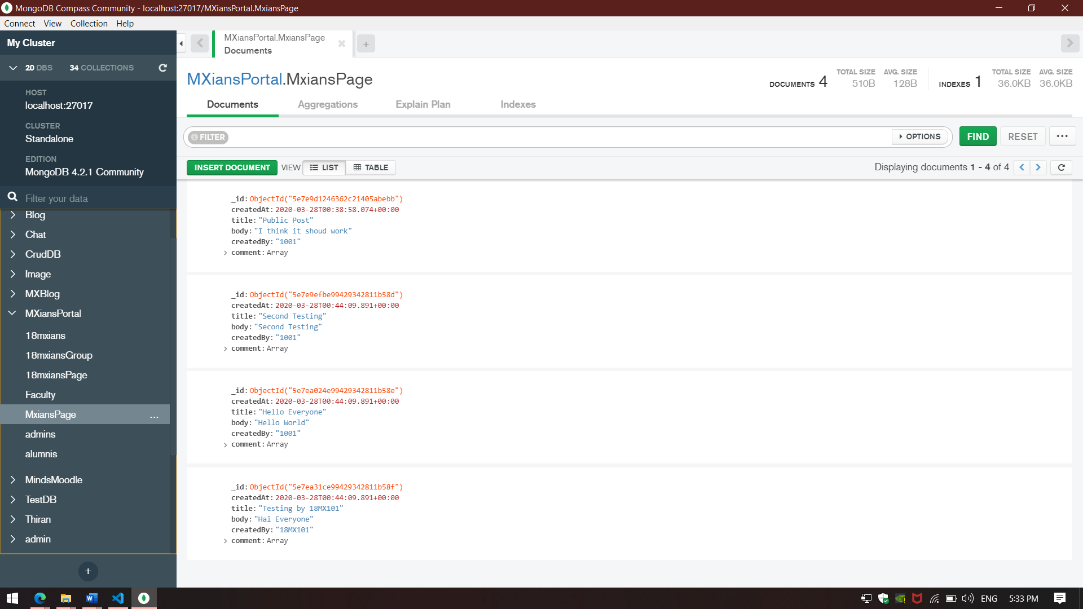
**Fig 3.3 18mxians Collection.**



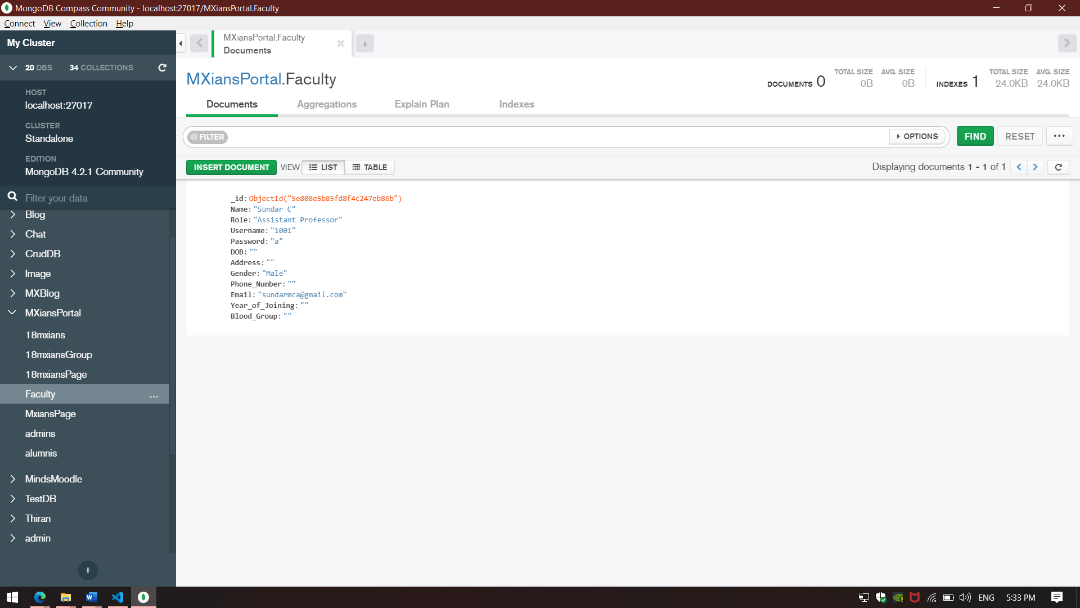
**Fig 3.4 18mxiansGroup Colleciton.**



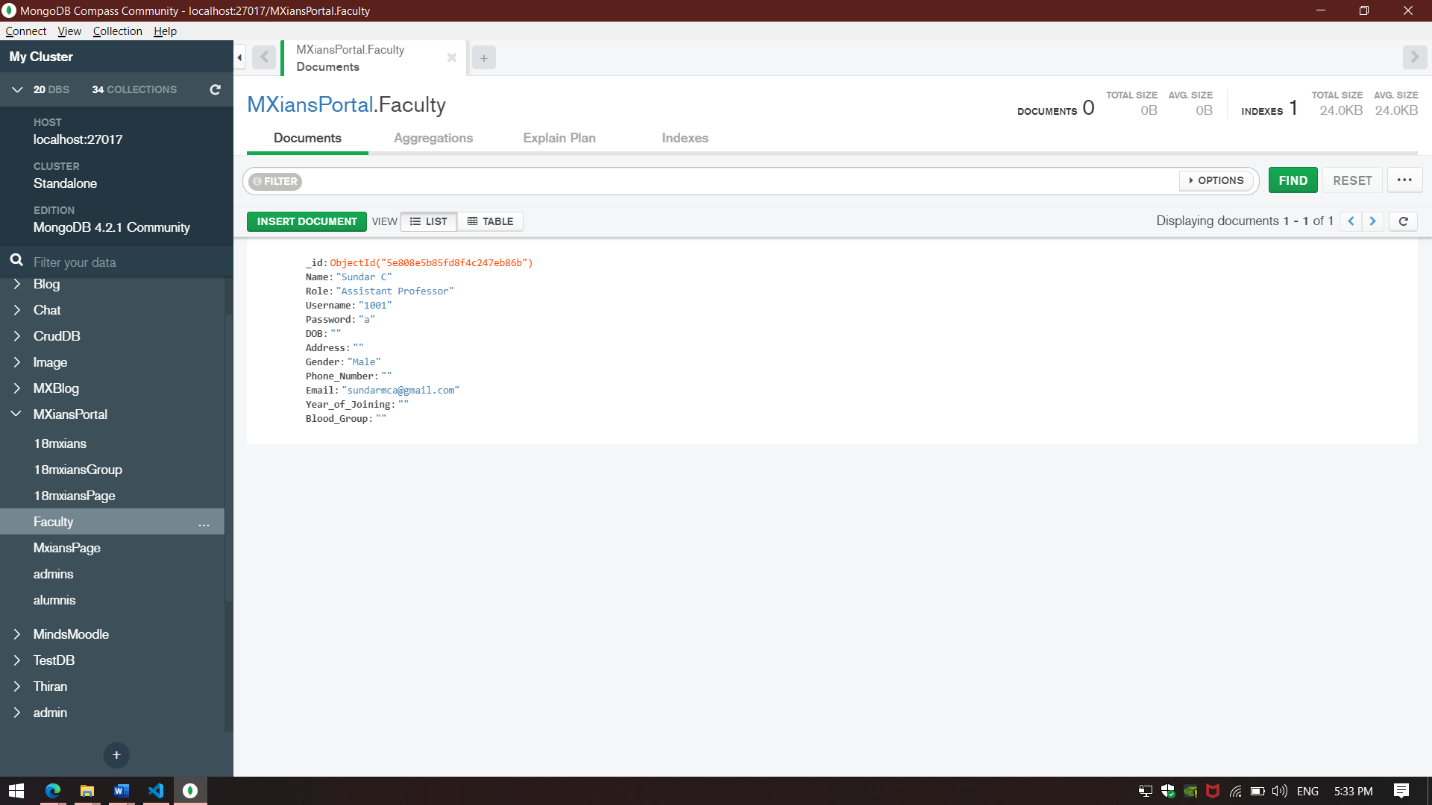
**Fig 3.5 18mxiansPage Collection.**

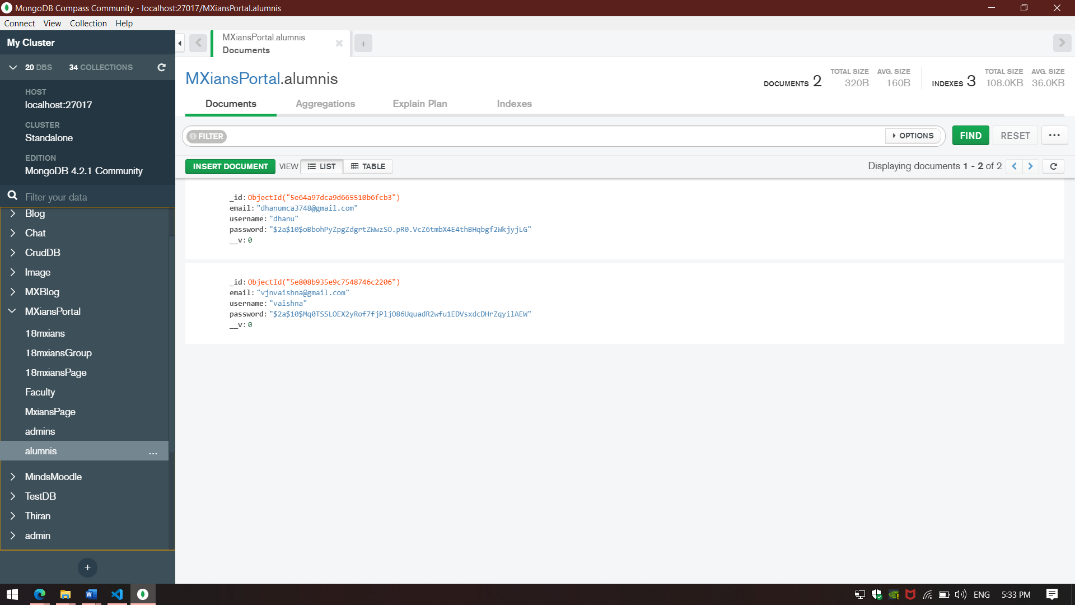


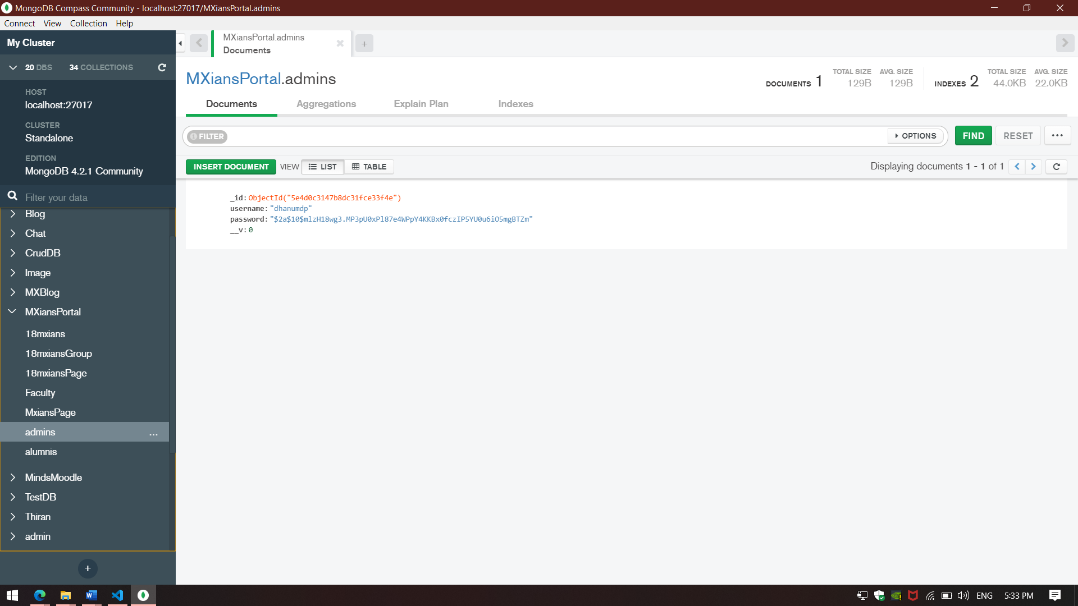
**Fig 3.6 MxiansPage Collection.**



**Fig 3.7 Faculty Collection.**





**Fig 3.8 Alumn Collection.**

**Fig 3.9 Admins Collection.**

As mentioned in Chapter 2.4, this project has four users.

* Admin
* Student
* Faculty
* Alumni

**3.2.2 Admin**

**Fig 3.10** displays the login page for the admin.

**Fig 3.11** displays the process of creating class by the admin.

**Fig 3.12** displays the process of creating faculty by the admin.

**Fig 3.13** displays the process of creating page by the admin.

**Fig 3.14** displays the process of creating group by the admin.

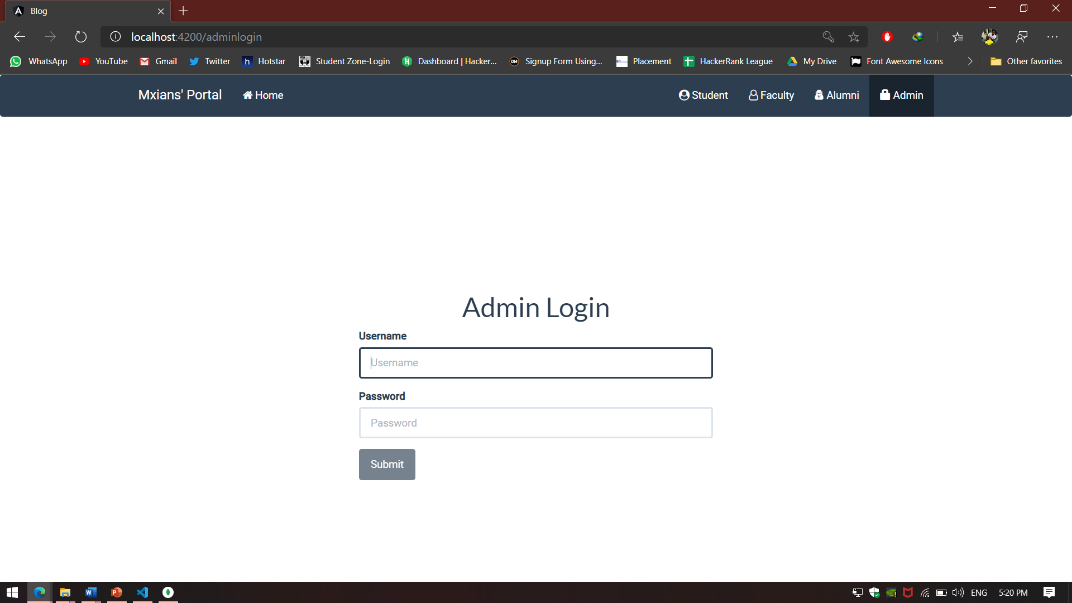
**Fig 3.15** displays the process of deleting student by the admin.

**Fig 3.16** displays the process of deleting class by the admin

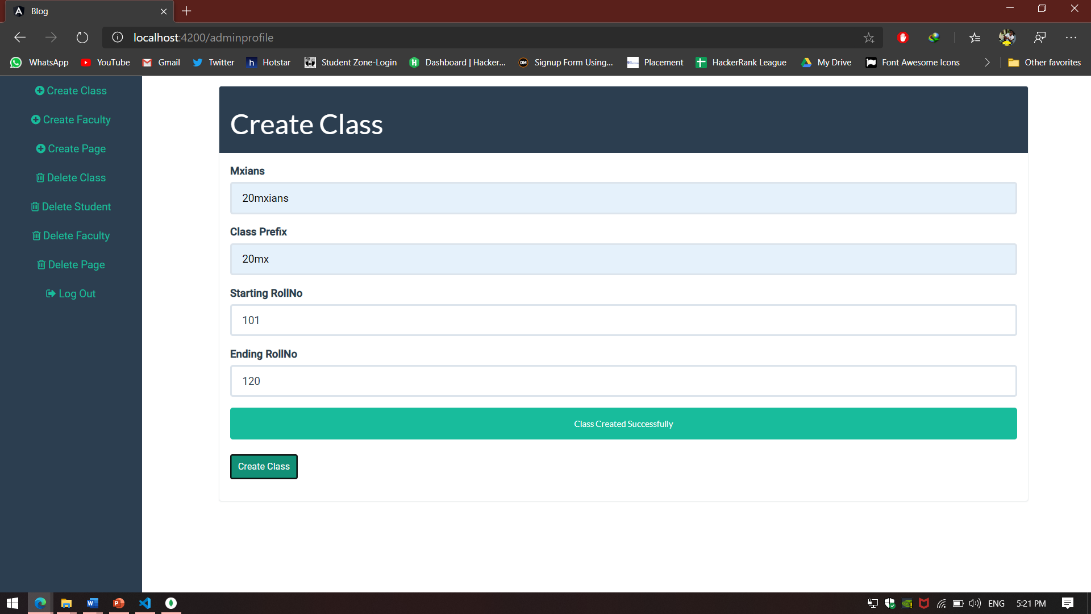
**Fig 3.17** displays the process of deleting faculty by the admin.

**Fig 3.18** displays the process of deleting page by the admin.

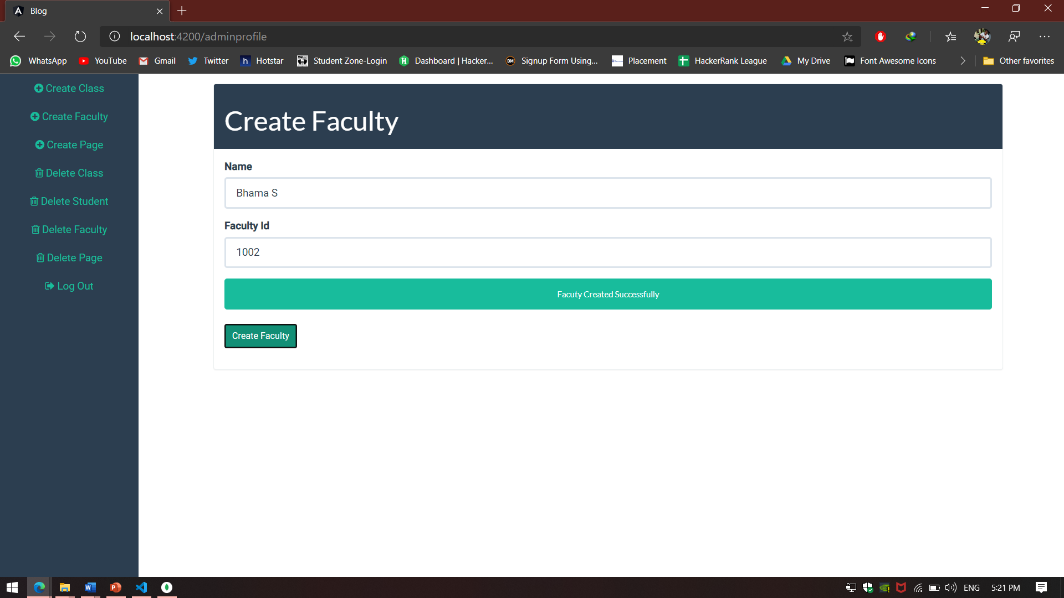
**Fig 3.19** displays the process of deleting group by the admin.



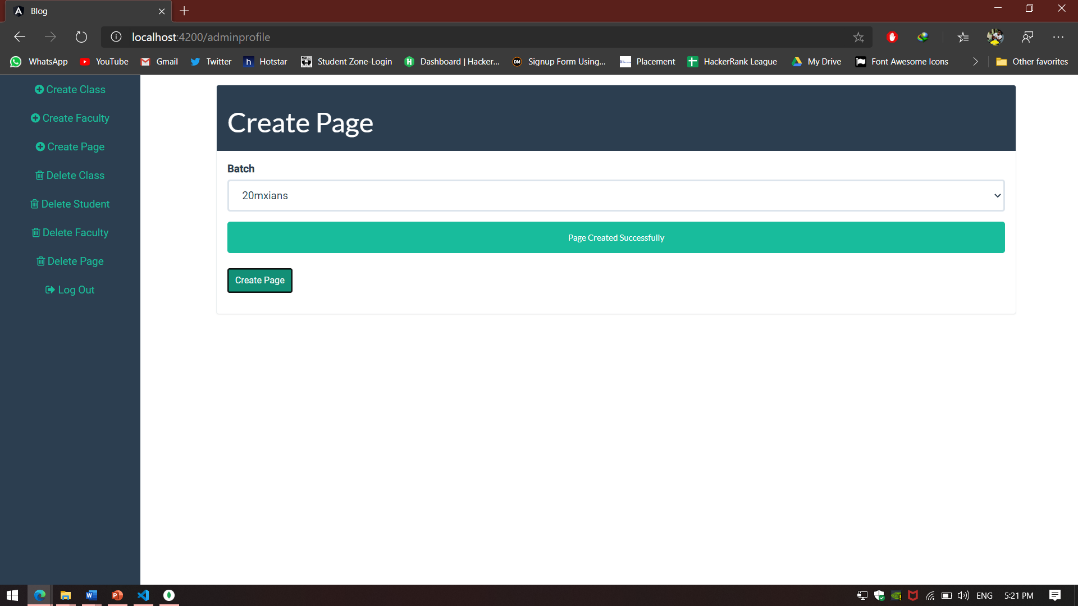
**Fig 3.10 Admin Login.**



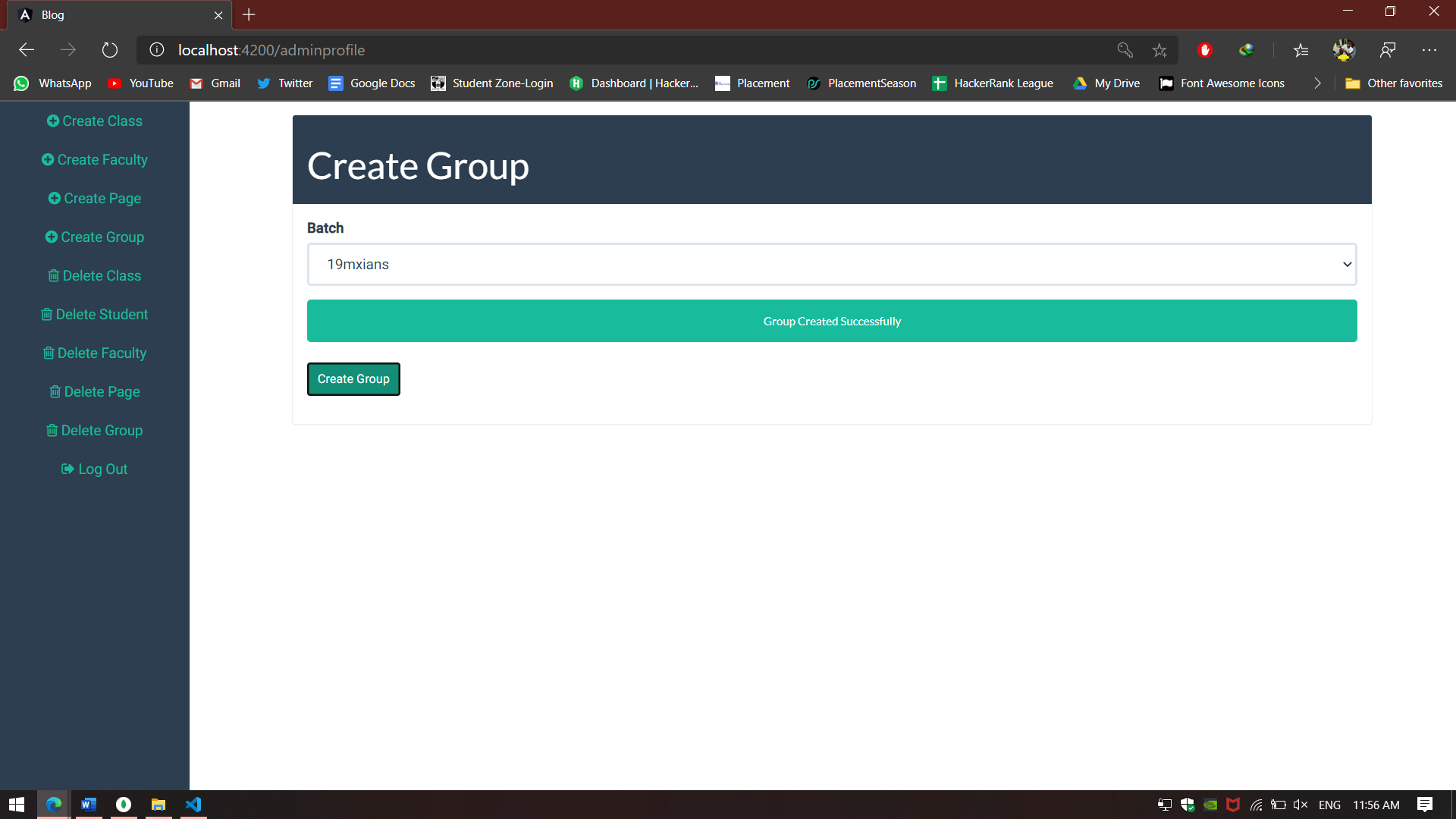
**Fig 3.11 Class Creation by Admin.**



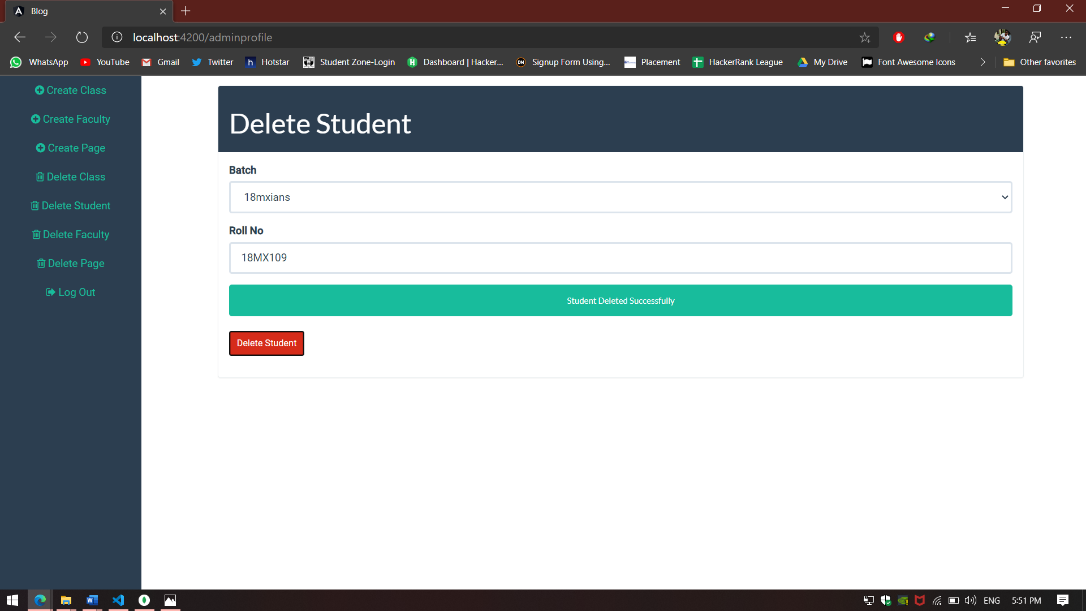
**Fig 3.12 Faculty Creation by Admin.**



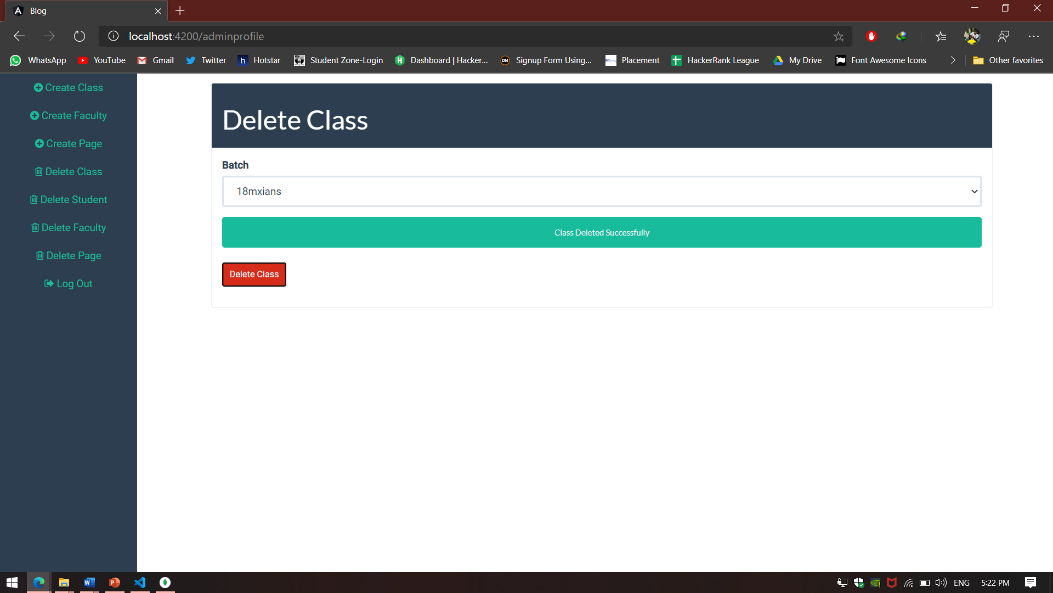
**Fig 3.13 Page Creation by Admin.**



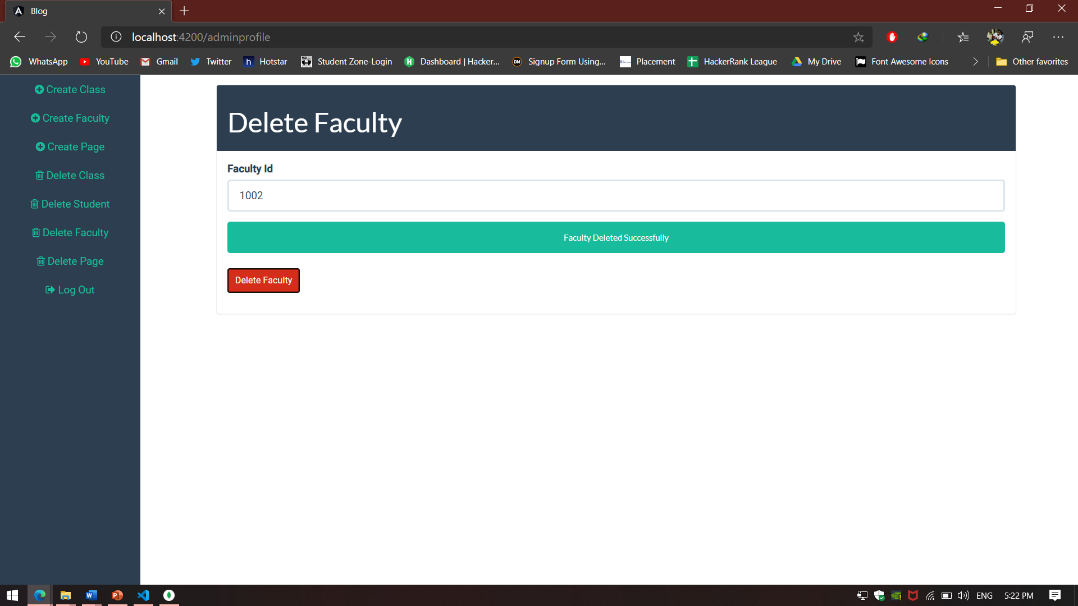
**Fig 3.14 Group Creation by Admin.**



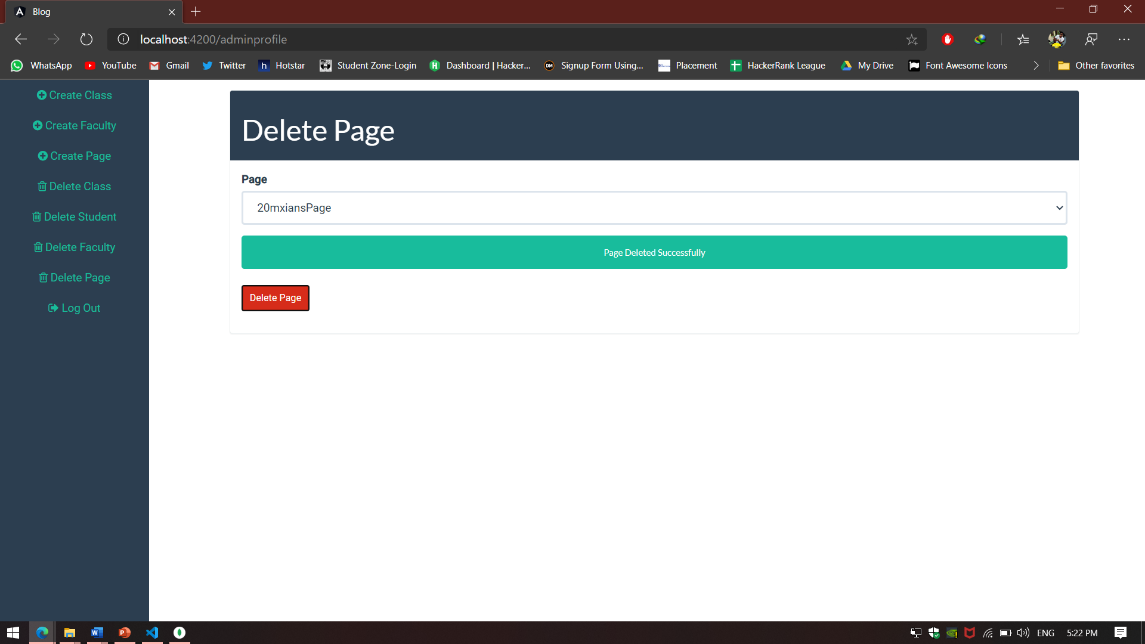
**Fig 3.15 Student Deletion by Admin.**



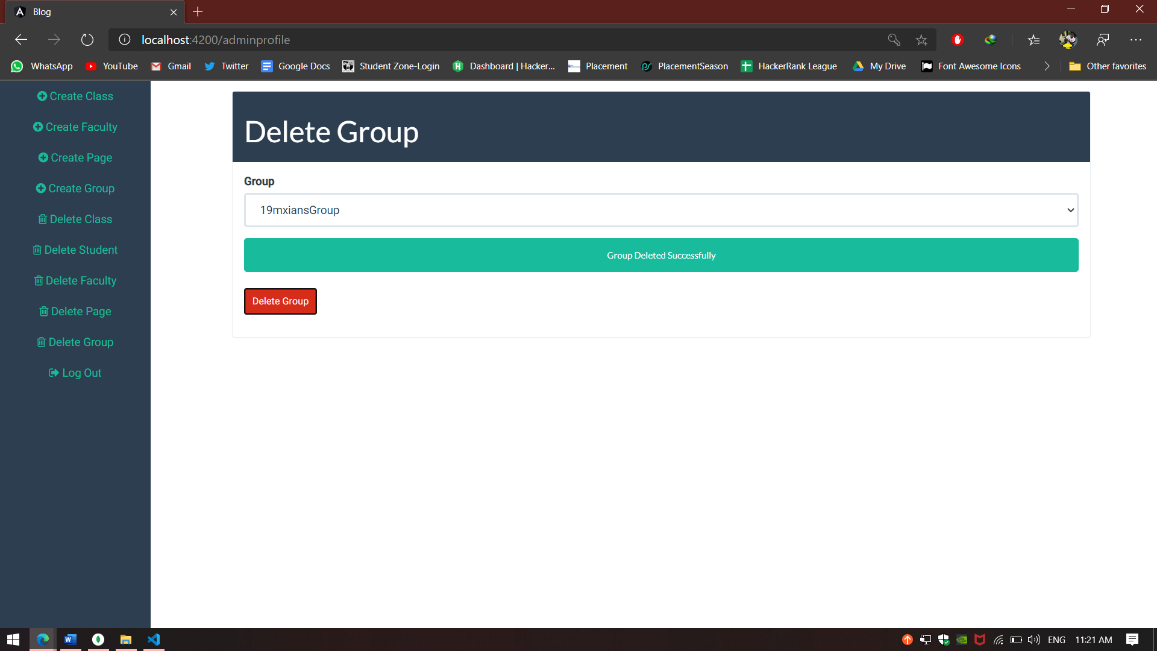
**Fig 3.16 Class Deletion by Admin.**



**Fig 3.17 Faculty Deletion by Admin.**



**Fig 3.18 Page Deletion by Admin.**



**Fig 3.19**

**Fig 3.19 Group Deletion by Admin.**

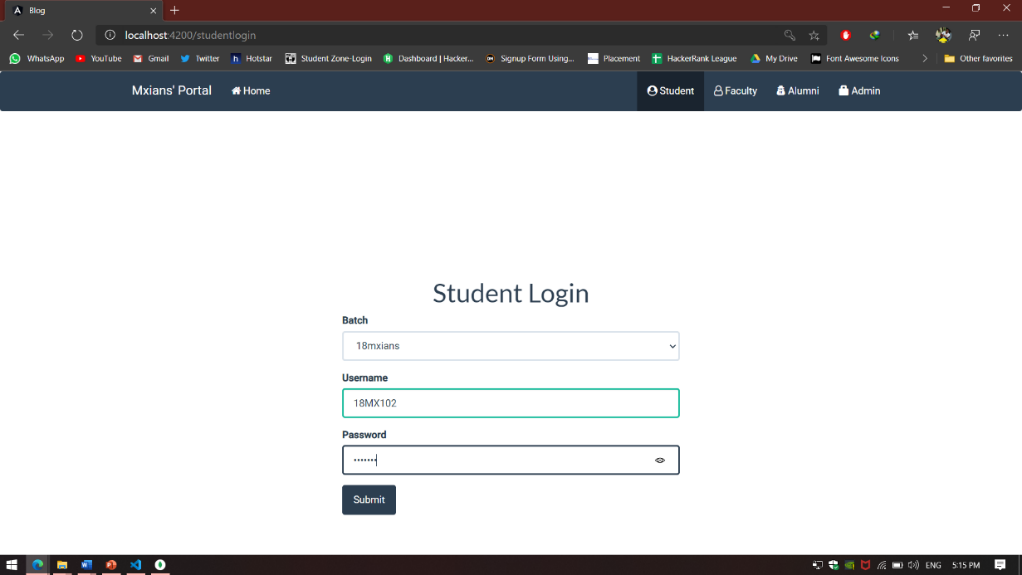
**3.2.3 Student**

**Fig 3.20** displays the login page for the student.

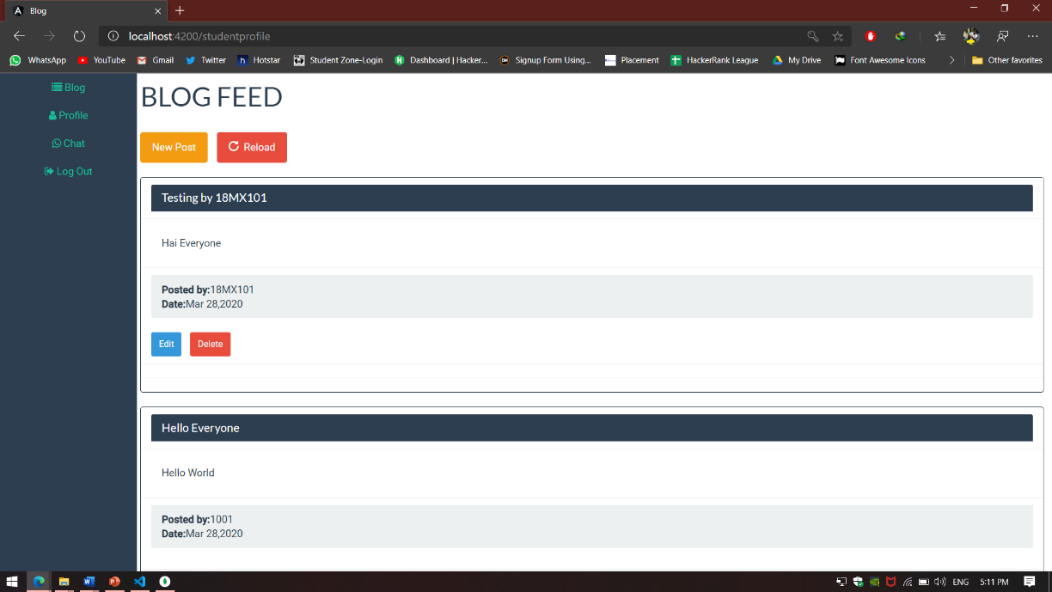
**Fig 3.21** displays the blog page for the student.

**Fig 3.22** displays the profile page of the student.

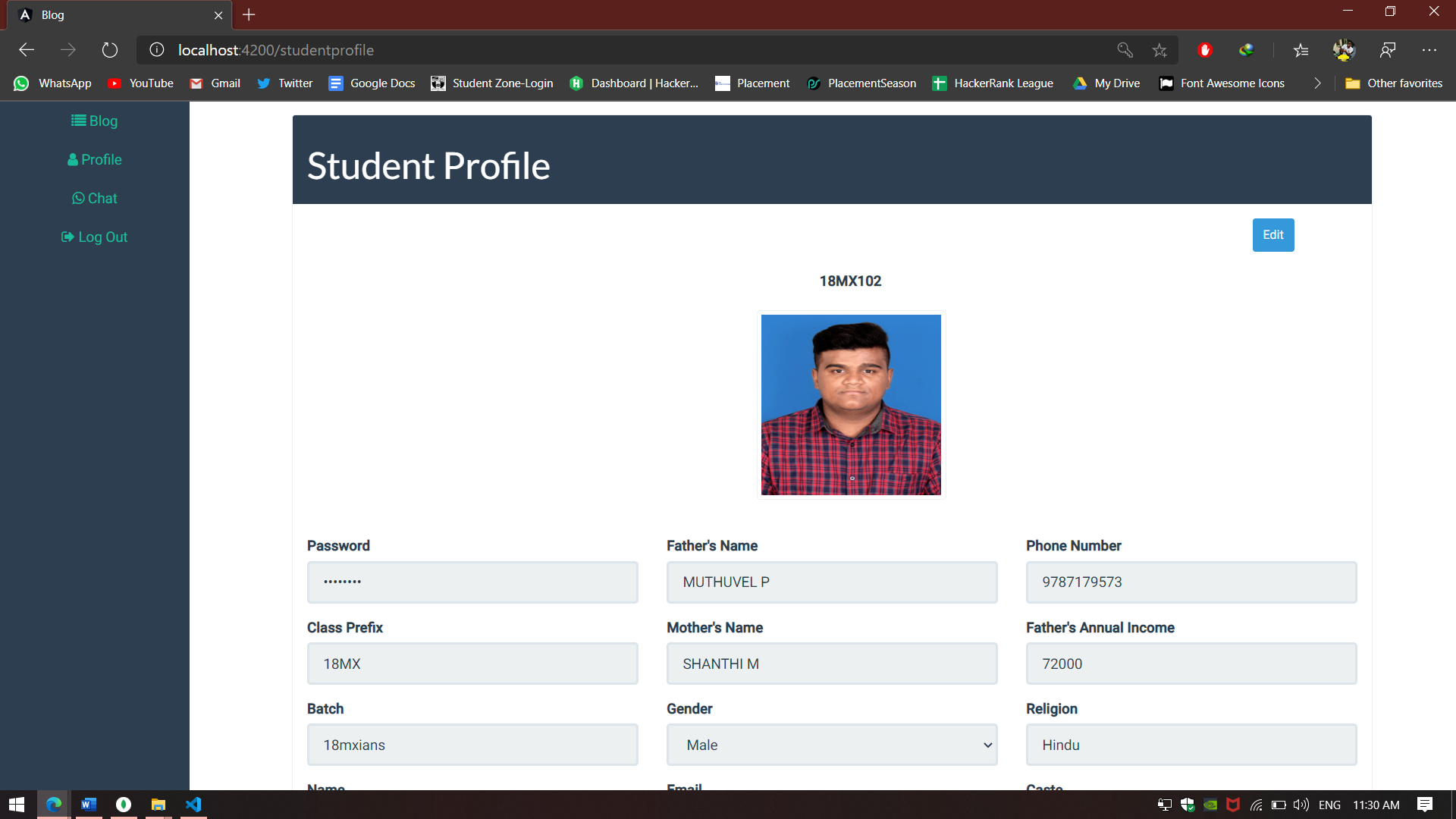
**Fig 3.23** displays the chat page of the student.



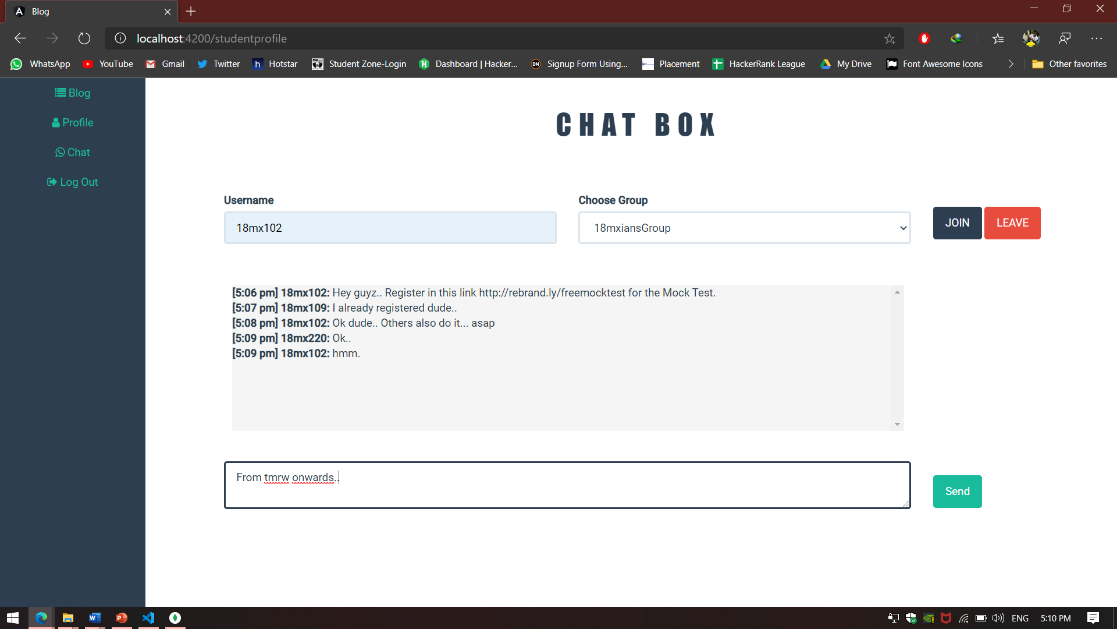
**Fig 3.20 Student Login.**



**Fig 3.21 Blog Page.**



**Fig 3.22 Student Profile.**



**Fig 3.23**

**Fig 3.23 Chat page.**

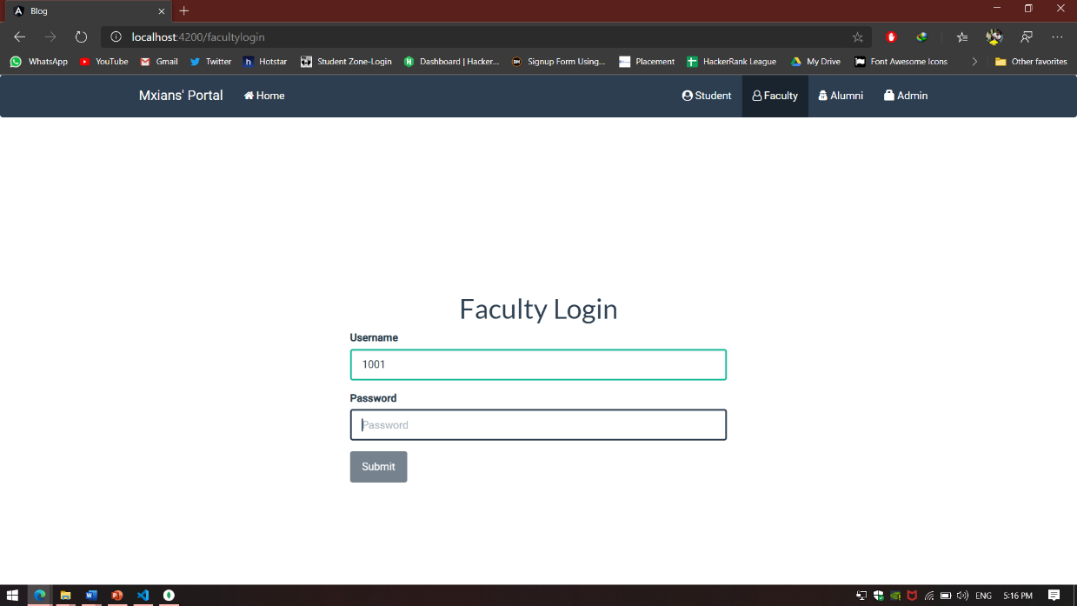
**3.2.4 Faculty :**

**Fig 3.24** displays the login page for the faculty.

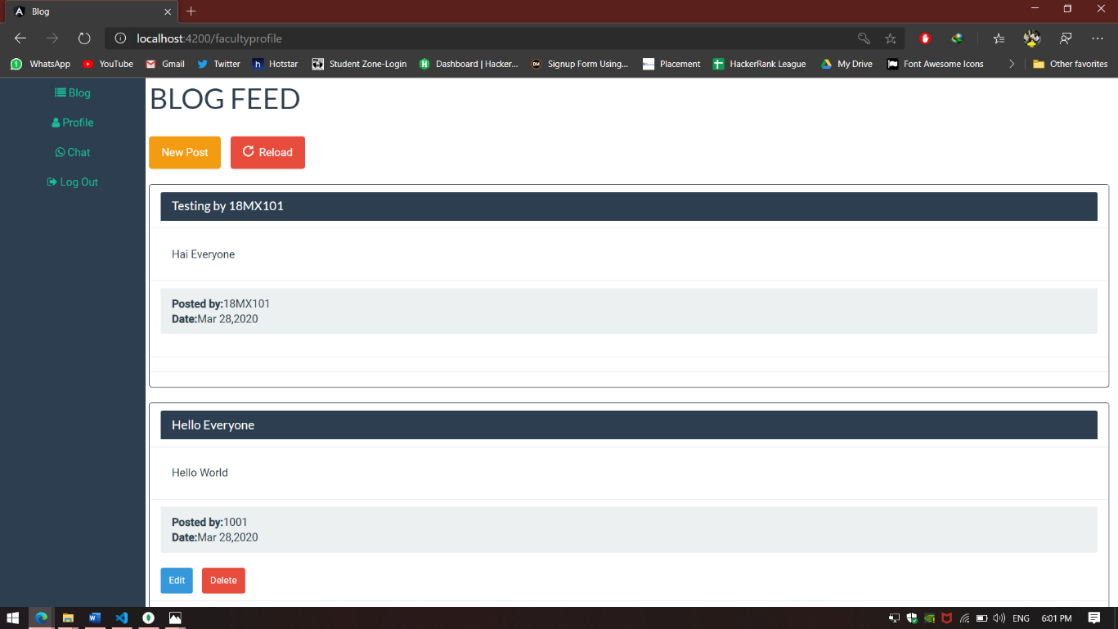
**Fig 3.25** displays the blog page for the faculty.

**Fig 3.26** displays the profile page of the faculty.

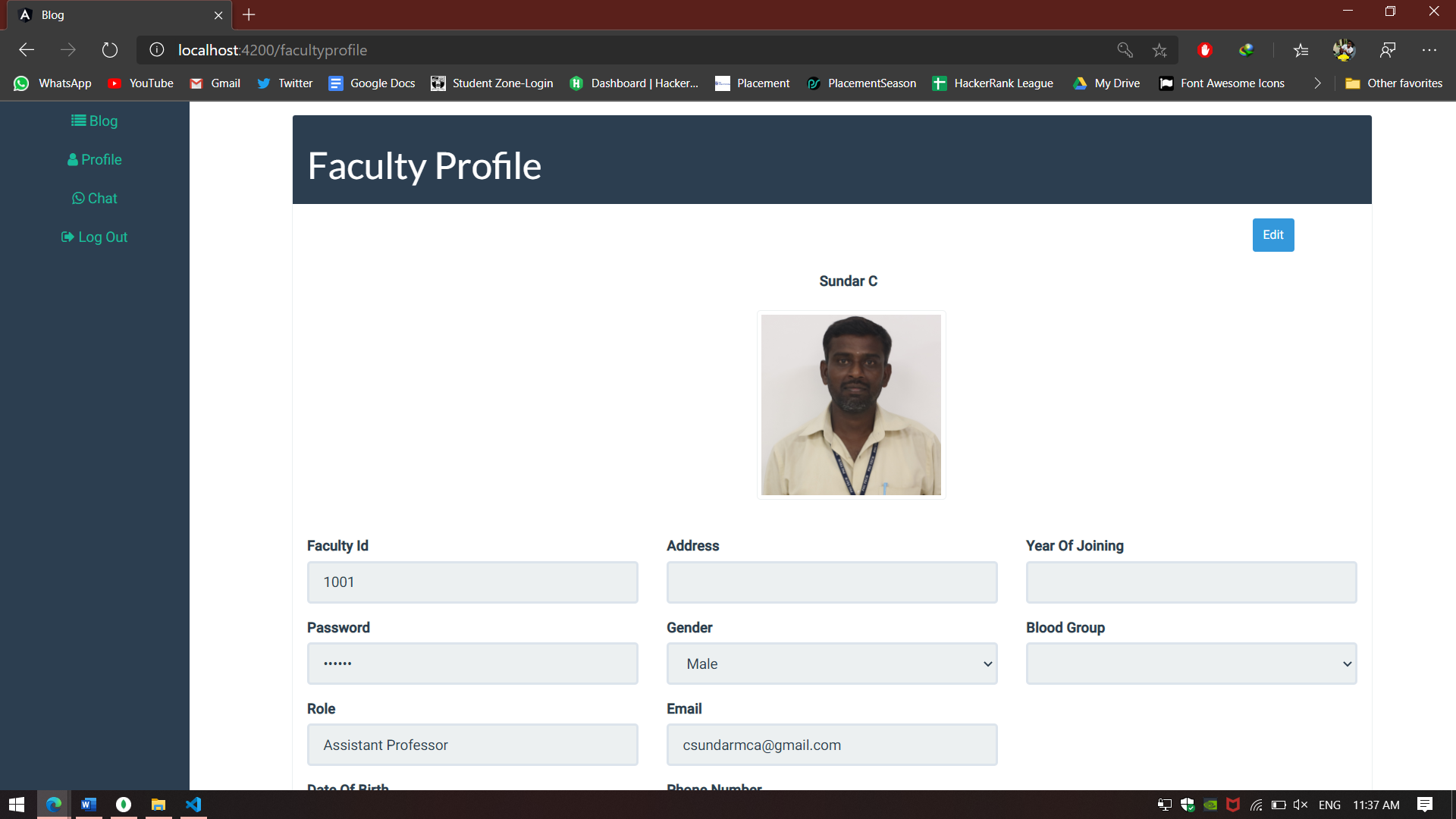
**Fig 3.27** displays the chat page of the faculty.



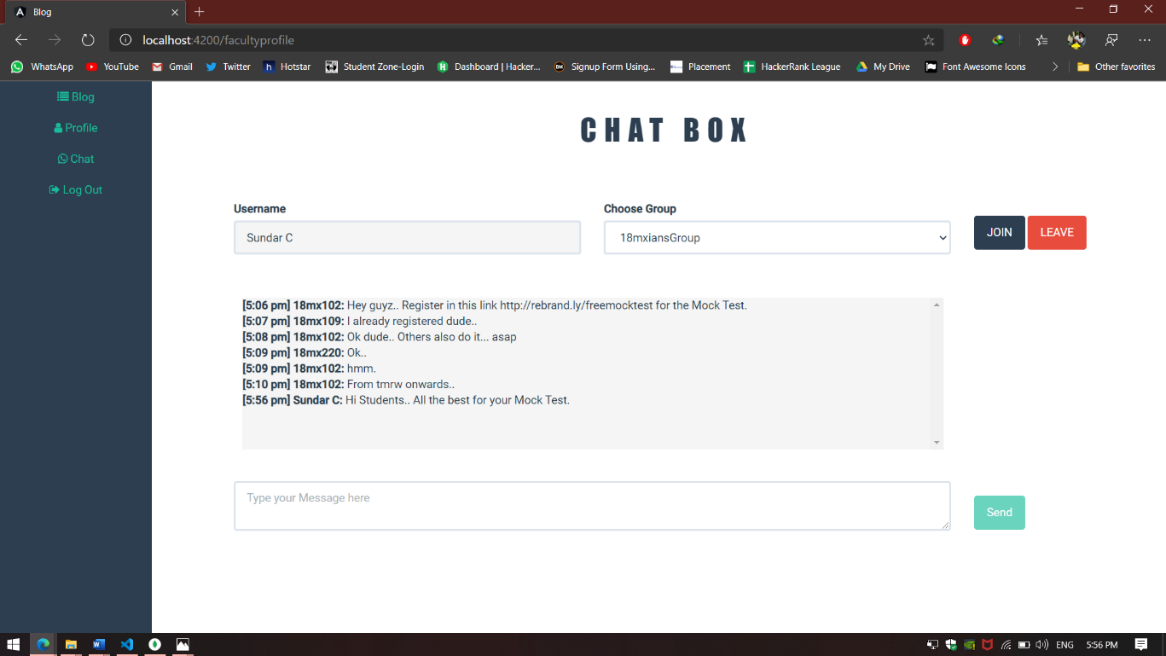
**Fig 3.24 Faculy Login.**



**Fig 3.25 Blog Page.**



**Fig 3.26 Faculty Profile.**



**Fig 3.27 Chat page.**

**3.2.5 Alumni :**

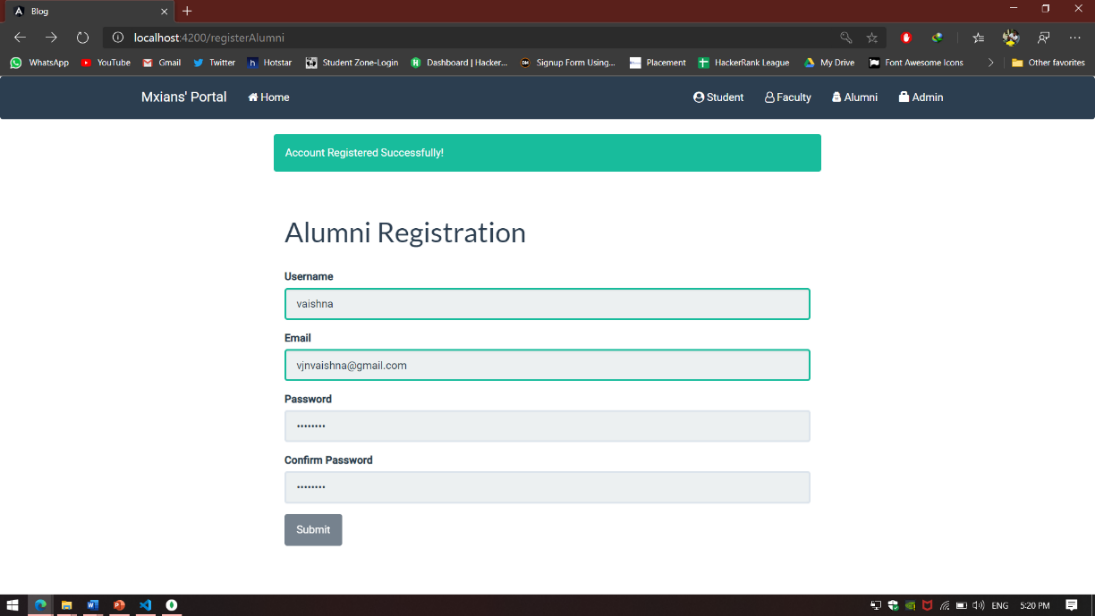
**Fig 3.28** displays the registration page of the alumni.

**Fig 3.29** displays the login page of the alumni.

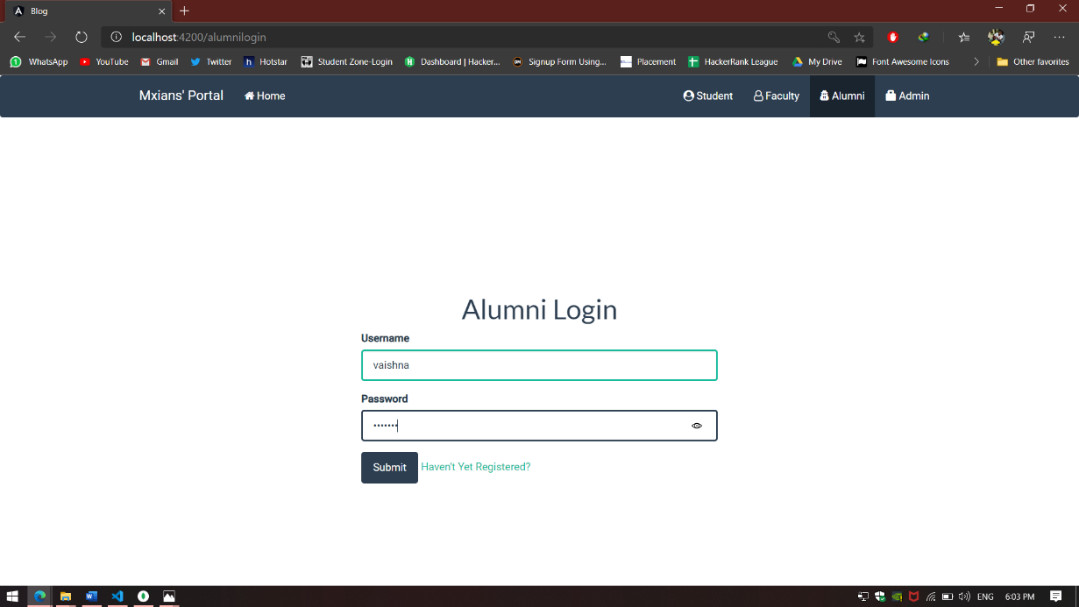
**Fig 3.20** displays the blog page of the alumni.

**Fig 3.31** displays the profile page of the alumni.

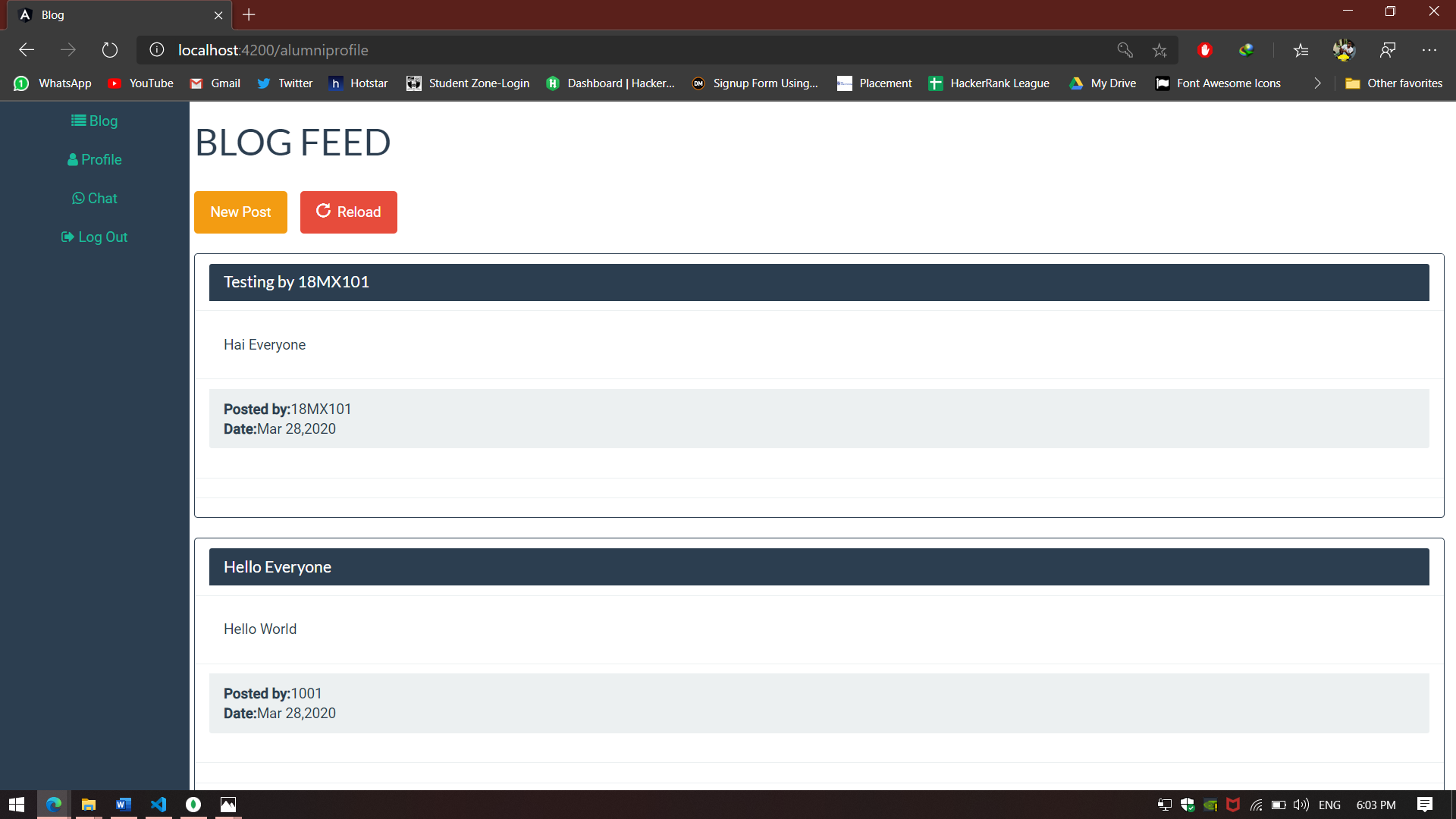
**Fig 3.32** displays the chat page of the alumni.



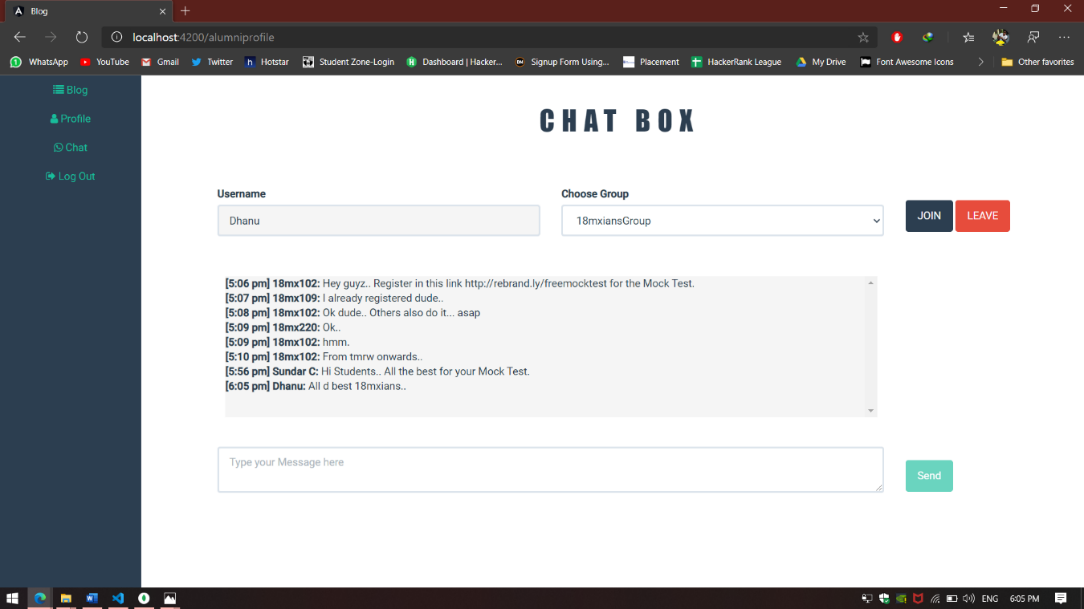
**Fig 3.28 Alumni Registration page.**



**Fig 3.29 Alumni Login page.**



**Fig 3.30 Blog page.**



**Fig 3.32 Chat page.**

**CHAPTER 4**

**TESTING**

Testing is a stage of implementation which aims to ensure whether the system works correctly and efficiently before live operation commences. System testing makes a good logical assumption that if all the paths of the system are correct then the goal will be achieved. This application is subjected to a variety of tests. A series of testing are performed for the proposed system.

* Unit Testing
* Integration Testing

**Unit Testing**

In Unit testing each module has been evaluated separately. This project handles various modules Admin, Student, Faculty and Alumni. Test case consists of conditions, expected Results,Actual Results and the Status. The listed below are test cases of unit testing.

* **Admin**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | CONDITIONS | EXPECTED RESULTS | ACTUAL RESULTS | STATUS |
| 1. | Login to authorised users | Should promote to admin process  (Class Creation & Deletion,Faculty Creation & Deletion,Blog Page Creation & Deletion) | Promotes to admin process (Class Creation & Deletion,Faculty Creation & Deletion,Blog Page Creation & Deletion) | PASS |
| 2. | Login to unauthorised users | Username not found | Username not found | PASS |
| 3. | Class Creation | Batch should be created successfully by providing access rights to each and every student | Batch created successfully by providing access rights to each and every student | PASS |
| 4. | Faculty Creation | Faculty should be created successfully by providing access  facility to each and everyone | Faculty Created Successfully.  Access facility is provided each and everyone | PASS |
| 5. | Blog Page Creation | Blog page should be created for a particular batch | Blog page created for particular batch | PASS |
| 6. | Class Deletion | Batch should be deleted by removing access rights to each and every student | Batch deleted successfully by removing access rights to each and every student | PASS |
| 7. | Student Deletion | Particular student must be removed from a batch and access privileges should be denied | Particular student removed from a batch and access privilege is denied | PASS |
| 8. | Faculty Deletion | Particular faculty must be removed and access privileges should be denied | Particular faculty is removed and access privilege is denied | PASS |
| 9. | Blog Page Deletion | Blog page should be deleted for particular batch | Blog page deleted for particular batch | PASS |

* **Alumni**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | CONDITIONS | EXPECTED RESULTS | ACTUAL RESULTS | STATUS |
| 1. | Login to authorised users | Should promote to blog page | Promotes to blog page | PASS |
| 2. | Login to unauthorised users | Username not found | Username not found | PASS |
| 3. | Clicking  “Haven’t yet registered”  (i.e)New Users | Should promote to registration page | Promotes to registration page | PASS |
| 4. | View Profile | Should promote to alumni profile page | Promotes to alumni profile page | PASS |
| 5. | Edit Profile | It should provide access to edit default details and update details | Providing access to edit default details and update details | PASS |
| 6. | Blog Creation | It should provide access to post a blog with blog title and content | Providing access to post a blog with blog title and content | PASS |
| 7. | Edit/Delete Blog | Should Provide access to edit a blog or delete a blog with a confirmation msg | Providing access to edit a blog or delete a blog with a confirmation msg | PASS |
| 8. | Chat | It should Provide access to group chat by joining a particular chat room | Providing access to group chat by joining a particular chat room | PASS |
| 9. | Join Chat Room | Should provide a user to join a chat room with a message of joining to other participants of the room and to view all the messages available on a particular room | A user can join a chat room with a message of joining to other participants of the room and to view all the messages available on a particular room | PASS |
| 10. | Leave Chat Room | It should provide a user to leave a chat room with a message of leaving to other participants of a particular room | Should provide a user to leave a chat room with a message of leaving to other participants of a particular room | PASS |
| 11. | Logout | It should enable a user to logout of the app. | Enabling a user to logout of the app. | PASS |

* **Faculty**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | CONDITIONS | EXPECTED RESULTS | ACTUAL RESULTS | STATUS |
| 1. | Login to authorised users | Should promote to blog page | Promotes to blog page | PASS |
| 2. | Login to unauthorised users | Username not found | Username not found | PASS |
| 3. | View Profile | Should promote to alumni profile page | Promotes to alumni profile page | PASS |
| 4. | Edit Profile | It should provide access to edit default details and update details | Providing access to edit default details and update details | PASS |
| 5. | Download Files | It should enable a faculty to view all the uploadedfiles by the student and download it whenever needed | Enables a faculty to view all the uploaded files by the student and download it whenever needed | PASS |
| 6. | Blog Creation | It should provide access to post a blog with blog title and content | Providing access to post a blog with blog title and content | PASS |
| 7. | Edit/Delete Blog | Should Provide access to edit a blog or delete a blog with a confirmation msg | Providing access to edit a blog or delete a blog with a confirmation msg | PASS |
| 8. | Chat | It should Provide access to group chat by joining a particular chat room | Providing access to group chat by joining a particular chat room | PASS |
| 9. | Join Chat Room | Should provide a user to join a chat room with a message of joining to other participants of the room and to view all the messages available on a particular room | A user can join a chat room with a message of joining to other participants of the room and to view all the messages available on a particular room | PASS |
| 10. | Leave Chat Room | It should provide a user to leave a chat room with a message of leaving to other participants of a particular room | Should provide a user to leave a chat room with a message of leaving to other participants of a particular room | PASS |
| 11. | Logout | It should enable a user to logout of the app. | Enabling a user to logout of the app. | PASS |

* **Student**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | CONDITIONS | EXPECTED RESULTS | ACTUAL RESULTS | STATUS |
| 1. | Login to authorised users | Should promote to blog page | Promotes to blog page | PASS |
| 2. | Login to unauthorised users | Username not found | Username not found | PASS |
| 3. | Upload Files | Should enable a student to upload all kinds of required files | Enables a student to upload all kinds of required files | PASS |
| 4. | View Profile | Should promote to alumni profile page | Promotes to alumni profile page | PASS |
| 5. | Edit Profile | It should provide access to edit default details and update details | Providing access to edit default details and update details | PASS |
| 6. | Blog Creation | It should provide access to post a blog with blog title and content | Providing access to post a blog with blog title and content | PASS |
| 7. | Edit/Delete Blog | Should Provide access to edit a blog or delete a blog with a confirmation msg | Providing access to edit a blog or delete a blog with a confirmation msg | PASS |
| 8. | Chat | It should Provide access to group chat by joining a particular chat room | Providing access to group chat by joining a particular chat room | PASS |
| 9. | Join Chat Room | Should provide a user to join a chat room with a message of joining to other participants of the room and to view all the messages available on a particular room | A user can join a chat room with a message of joining to other participants of the room and to view all the messages available on a particular room | PASS |
| 10. | Leave Chat Room | It should provide a user to leave a chat room with a message of leaving to other participants of a particular room | Should provide a user to leave a chat room with a message of leaving to other participants of a particular room | PASS |
| 11. | Logout | It should enable the user to logout. | Enabling the user to logout. | PASS |

**Integration Testing**

Integration testing is the process of testing the application by integrating all modules as single applications. The code interacts well with the framework and accepts the external dependencies.

For all the test cases mentioned above, after integration of this application it gives the same result as in the unit testing. So after integration there is no negative impact on the application.

**CHAPTER 5**

**CONCLUSION**

The project “MXIANS’ PORTAL” is a web portal for the department of Computer Applications, PSG College of Technology. This application is useful for building communication between the students, faculty and alumni and helps to know about the student and the alumni details. Eventhough many platforms are available for portal creation, use of MEAN stack makes this project effective in all aspects and the system is very flexible and extensible.

Currently this project is running in the local server machine, in future it will be uploaded in the private cloud of the department of Computer Applications, PSG College of Technology. Availability of this project in cloud enables the users to access it in anywhere at anytime.

**BIBLIOGRAPHY**

* <https://docs.mongodb.com/>
* <https://expressjs.com/>
* <https://angular.io/docs>
* <https://nodejs.org/en/docs/>
* <https://socket.io/>
* <https://www.youtube.com/>
* <https://stackoverflow.com/>
* <https://github.com/>