**AMRITSAR GROUP OF COLLEGES**

**Autonomous status conferred by UGC | NAAC-A Grade,**

**Six Week Training Report**

**On**

# “STUDENT HELP WEBSITE”

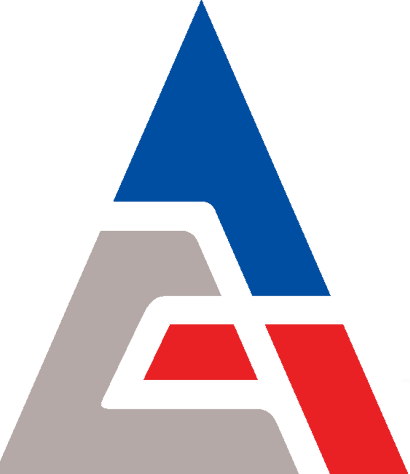
Submitted in the Partial fulfilment of the requirement for the Award of Degree of

**Bachelor of Technology**

**In**

## COMPUTER SCIENCE & ENGINEERING

**Batch (2020-24)**



**Submitted To: Submitted By:**

Dr. Sandeep Kad Kanish Chhabra (2000120)

*Head of Department* Karandeep Singh Gill (2000123)

*(CSE)* Kartik Arora (2132015)

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**Amritsar Group of Colleges, Amritsar**

# DECLARATION

We undersigned solemnly declare that the report on project **Student help website**is based on our own work carried out during the course of our study.

We assert the statements made and conclusions drawn are an outcome of our research work. We further certify that

1. The work contained in the report is original and has been done by us.
2. The work has not been submitted to any other Institution for any other degree/diploma/certificate in this college or any other of India or abroad.
3. We followed the guidelines provided by the college in writing the report.
4. Whenever We have used materials (data, theoretical analysis and text) from other sources, We have given due credit to them in the text of the report and giving their details in the bibliography.

**Kanish Chhabra**

**University Roll no.: 2000120**

**Karandeep Singh Gill**

**University Roll no.: 2000123**

**Kartik Arora**

**University Roll no.: 2132015**

**ACKNOWLEDGEMENT**

We would like to express our heartfelt gratitude to VMM (Veenus Mind Media) Education and **Mr. Shalinder** for giving us the opportunity to work under their guidance and help us gain immensely enriching professional experience in our training. Our sincere thanks to project guide, **Mr. Subhrato** for giving us valuable inputs and ideas right from the selection of topic for project till its successful completion.

We are extremely thankful to the **Dr. Sandeep Kad** Head of Department and all faculty members of CSE Department at Amritsar Group of Colleges, Amritsar for their co-ordination and for their kind guidance and encouragement.

The successful completion of our project would not have been possible without the dedicated support from all our mentor, family and friends.

This project completion has indeed helped us explore more knowledge avenues related to **Web Developement** and we are sure it will help us in future too.

**INDEX PAGE**

|  |  |  |
| --- | --- | --- |
| **S. No.** | **Tittle** | **Page No.** |
| 1 | Training Objective | 1 |
| 2 | Organization Brief | 2-3 |
| 3 | Technology Used | 4-8 |
| 4 | Software Model | 9-17 |
| 5 | Project Details | 18-22 |
| 6 | Project Screen Shots with Explanations | 23-27 |
| 7 | Code of the Project | 28-67 |
| 8 | Bibliography | 68 |

## TRAINING OBJECTIVE

* Understand the principles of creating an effective Full stack web page using HTML, CSS, Bootstrap, JavaScript & NodeJS
* Understand how to plan and conduct user research related to web usability.
* Learn techniques of responsive web design, including user queries.
* Become familiar with graphic design principles that relate to web design and learn how to implement theories into practice.
* Learn CSS grid layout and flexbox.
* Learn how to create own localhost using XAMPP server.
* Understand how to make good web page which feels comfortable and secure to the users.
* Learn Communication Skills, Presentation Skills &Teamwork.
* Learn how to connect Front End with Back End.

## ORGANIZATION PROFILE

The Institute where we are pursuing our six weeks Industrial Training is VMM (Veenus Mind Media), at Amritsar*.*

### **DETAILS OF VMM**

VMM Education’s journey started in January 2005 with a vision of bringing computer education of global standard in the holy city of Amritsar. To turn this dream into reality we needed to create a talent pool of bright young minds who would power the engines of growth of the global economy, today seven years later VMM Education or VMM, as it is popularly known as, is the largest and the most trusted computer center of the region, with annual turn out of more than 1000 students each year. The reason for the success of VMM is simply the “Hard work” that our team has put in these seven years.

In this past 7 years, VMM Education has provided world class training in global technologies such as Java, .Net, Oracle and Linux, while keeping our syllabus up to date with the current industry standard. We have managed to successfully train more than 7000 engineers who are currently working in Global Multinationals like TCS, Tech Mahindra, Infosys, and CSC etc.

VMM is today the favorite choice of students of various engineering college for pursuing their six months or six weeks industrial training. A unique “industry-endorsed curriculum,” crafted by professionals of VMM enhances the job-readiness and employability of learners and equips them for the IT Industry.

To provide IT education which can match with the global IT standards, VMM also undertake industrial projects under the banner of Veenus Software Solutions (VSS as the Veenus Software solution is popularly known as undertake projects from UK & USA example **im4schools.co.uk**) this also allows our students to work on the live projects and make projects for the various industries. Some of our products include Point of sale software for Super Markets with barcode reader support, Finger print attendance Management System that can work for school’s colleges and other institutes, Remote Lan Controller which can be used to view remote desktops on LAN or WAN.VMM has the world class computer labs that are equipped with the latest Hardware and software so that the practicals of the students must always remain for that and we continuously upgrade our hardware and software every 6 months.

Attendance of students is one of the major features of VMM, students are required to record their day-to-day attendance using Biometric Finger print recognition device this allows us to provide accurate attendance of students to their parents and college during their training time.This course includes two languages C and C++ that allows us to build a strong foundation of programming for the beginner and First year & Second year engineering graduates

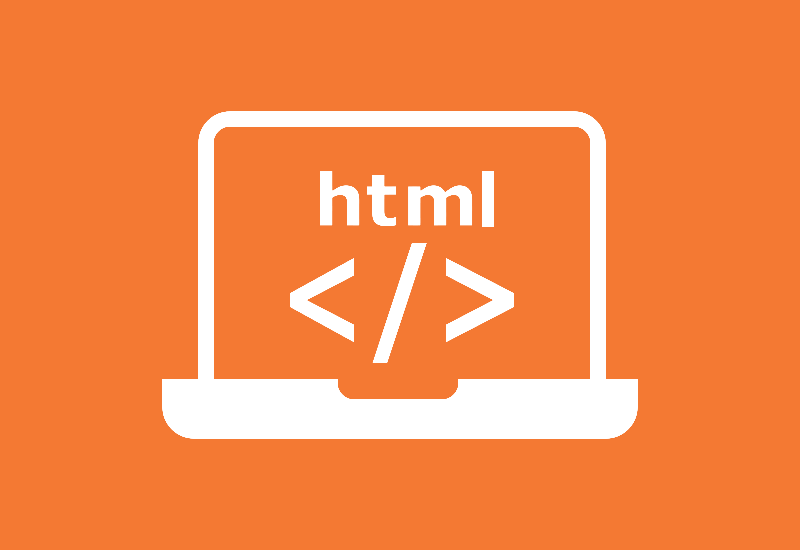
Apart from above training program VMM provides a bouquet of courses in windows application development using VB.Net/C#.Net/Java, Web Development using ASP.Net or J2EE, Mobile Software development in Android. Veenus Mind Media or VMM is a Software Solutions Provider. The company specializes in providing industry focused solutions and customized development. The development work ranges from management-oriented solutions utilizing the latest of databases, design and programming tools to web development designing and Network resources. Veenus Mind Media is a software arm of Hotel Veenus International (Amritsar), it is one of the fastest growing software solution providers extending its arms in diversified activities such as School & College Network AMC’s, Wi-Fi Internet Management solutions in school/colleges/shopping malls & Hotels, customized software development, Web-Site Development & Web Marketing (SEO) The total Turnover of entire group is more than 5 million (INR) VMM commenced its operations in the year 2004 as 100% software consultants. VMM was started from one home computer and today VMM has got infrastructure of more than one million (INR). VMM operates it two wings one is of the software consultancy services and other is **VMM Education** i.e., the Education wing which deals with providing the world class computer education to its students.

# TECHNOLOGIES AND TOOLS USED

**FRONT END**

### **HTML**

### 



The Hypertext Markup Language, or HTML is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets and scripting languages such as JavaScript.

* HTML stands for Hyper Text Markup Language
* HTML describes the structure of a Webpage
* HTML consists of a series of elements
* HTML elements tell the browser how to display the content
* HTML elements are represented by tags

**II.CSS**

Stands for "Cascading Style Sheet." Cascading style sheets are used to format the layout of Web pages. They can be used to define text styles, table sizes, and other aspects of Web pages that previously could only be defined in a page's HTML.CSS helps Web developers



create a uniform look across several pages of a Web site. Instead of defining the style of each table and each block of text within a page's HTML, commonly used styles need to be defined only once in a CSS document. Once the style is defined in cascading style sheet, it can be used by any page that references the CSS file. Plus, CSS makes it easy to change styles across several pages at once. For example, a Web developer may want to increase the default text size from 10pt to 12pt for fifty pages of a Web site. If the pages all reference the same style sheet, the text size only needs to be changed on the style sheet and all the pages will show the larger text.

While CSS is great for creating text styles, it is helpful for formatting other aspects of Web page layout as well. For example, CSS can be used to define the cell padding of table cells, the style, thickness, and color of a table's border, and the padding around images or other objects. CSS gives Web developers more exact control over how Web pages will look than HTML does. This is why most Web pages today incorporate cascading style sheets.

**III. BOOTSTRAP**

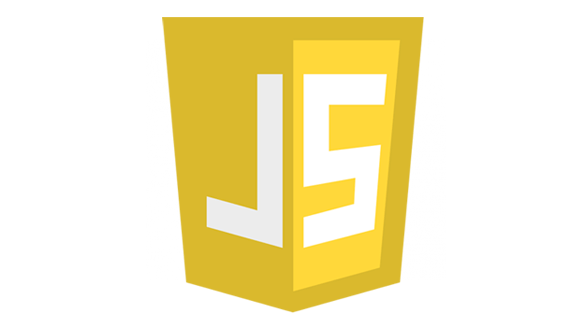


Bootstrap is a free and open-source frontend development framework for the creation of websites and web apps. The Bootstrap framework is built on HTML, CSS, and JavaScript (JS) to facilitate the development of responsive, mobile-first sites and apps. Responsive design makes it possible for a web page or app to detect the visitor’s screen size and orientation and automatically adapt the display accordingly; the mobile first approach assumes that smartphones, tablets and task-specific mobile apps are employees 'primary tools for getting work done and addresses the requirements of those technologies in design. Bootstrap includes user interface components, layouts and JS tools a long with the frame work for implementation. The software is available precompiled or as source code.

Mark Otto and Jacob Thornton developed Bootstrap at Twitter as a means of improving the consistency of tools used on the site and reducing maintenance. The software was formerly known as Twitter Blueprint and is sometimes referred to as Twitter Bootstrap.

In computers, the word bootstrap means to boot: to load a program into a computer using a much smaller initial program to load in the desired program (which is usually an operating system). In the physical world,a bootstrap is a small strap or loop at the back of a leather boot that enables you to pull the entire boot on and in general usage, bootstrap in the leveraging of a small initial effort into something larger and more significant.

#### **IV. JAVASCRIPT**

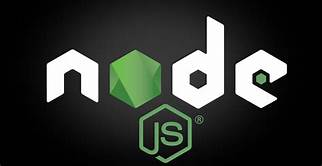


JavaScript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities. JavaScript was first known as Live Script, but Netscape changed its name to JavaScript, possibly because of the excitement being generated by Java. JavaScript made its first appearance in Netscape 2.0 in 1995 with the name Live Script. The general-purpose core of the language has been embedded in Netscape, Internet Explorer, and other web browsers.

Client-side JavaScript is the most common form of the language. The script should be included in or referenced by an HTML document for the code to be interpreted by the browser. It means that a web page need not be a static HTML, but can include programs that interact with the user, control the browser, and dynamically create HTML content. The JavaScript client-side mechanism provides many advantages over traditional CGI server-side scripts. For example, you might use JavaScript to check if the user has entered valid-mail address in a form field.

**BACK END**

1. **NODE JS**



**Node.js** is an open-source, cross-platform, JavaScript runtime environment that executes JavaScript code outside of a web browser. Node.js is a popular, lightweight web framework for beginners, and it is used by many big companies.

Node.js lets developers use JavaScript to write command line tools and for [server-side scripting—](https://en.wikipedia.org/wiki/Server-side_scripting) running scripts server-side to produce [dynamic web page c](https://en.wikipedia.org/wiki/Dynamic_web_page)ontent before the page is sent to the user's web browser.

When we typically think of JavaScript, our mind tends to go to web development. Until Node.js came along, there was really no way to run JavaScript outside of a browser. When we write a backend server and database, Node.js is a popular choice because we can run our code as a standalone application rather than something that can only be evaluated in a browser environment.

Node.js is an important tool for any JavaScript developer to understand. So, today, we’ll introduce you to Node.js and show you how to get started with a project.

## II. MySQL



MySQL is an open-source relational database management system (RDBMS). Its name is combinations of “My”, the name of co-founder Michael Widenius' daughter, and "SQL", the abbreviation for Structured Query Language. The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements. MySQL was owned and sponsored by a single for-profit firm, the Swedish company MySQL AB, now owned by Oracle Corporation.

MySQL is a central component of the [LAMP](https://en.wikipedia.org/wiki/LAMP_(software_bundle)) open-source web application software stack (and other "[AMP](https://en.wikipedia.org/wiki/List_of_AMP_packages)" stacks). LAMP is an acronym for "[Linux](https://en.wikipedia.org/wiki/Linux), [Apache](https://en.wikipedia.org/wiki/Apache_HTTP_Server), MySQL, and [Perl](https://en.wikipedia.org/wiki/Perl)/[PP](https://en.wikipedia.org/wiki/PHP)/[Python](https://en.wikipedia.org/wiki/Python_(programming_language))". Applications that use the MySQL database include: [TYPO3](https://en.wikipedia.org/wiki/TYPO3), [MODx](https://en.wikipedia.org/wiki/MODx), [Joomla](https://en.wikipedia.org/wiki/Joomla), [Word Press](https://en.wikipedia.org/wiki/WordPress), [phpBB](https://en.wikipedia.org/wiki/PhpBB), [MyBB](https://en.wikipedia.org/wiki/MyBB), and [Drupal](https://en.wikipedia.org/wiki/Drupal). MySQL is also used in many high-profile, large-scale [websites](https://en.wikipedia.org/wiki/Website), including [Google](https://en.wikipedia.org/wiki/Google) (though not for searches), [Facebook](https://en.wikipedia.org/wiki/Facebook), [Twitter](https://en.wikipedia.org/wiki/Twitter), [Flickr](https://en.wikipedia.org/wiki/Flickr), and [YouTube](https://en.wikipedia.org/wiki/YouTube).

MySQL was created by a Swedish company, [MySQL AB](https://en.wikipedia.org/wiki/MySQL_AB), founded by [David Axmark](https://en.wikipedia.org/wiki/David_Axmark), Allan Larsson and [Michael "Monty" Widenius](https://en.wikipedia.org/wiki/Michael_(Monty)_Widenius). Original development of MySQL by Widenius and Axmark began in 1994. The first version of MySQL appeared on 23 May 1995. It was initially created for personal usage from [MySQL](https://en.wikipedia.org/wiki/MSQL) based on the low-level language [ISAM](https://en.wikipedia.org/wiki/ISAM), which the creators considered too slow and inflexible.

**SOFTWARE MODEL**

**SDLC**

SDLC is the acronym of Software Development Life Cycle. SDLC is a framework defining tasks performed at each step in the software development process.

SDLC is a process followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process.

The following figure is a graphical representation of the various stages of a typical SDLC.



**Fig-1: SDLC**

A typical Software Development Life Cycle consists of the following stages –

* **Planning and Requirement Analysis** - Requirement analysis is the most important and fundamental stage in SDLC. It is performed by the senior members of the team with inputs from the customer, the sales department, market surveys and domain experts in the industry
* **Defining Requirements** - Once the requirement analysis is done the next step is to clearly define and document the product requirements and get them approved from the customer or the market analysts.
* **Designing the Product Architecture** - SRS is the reference for product architects to come out with the best architecture for the product to be developed.

### **Advantages:**

1. Improved Data Integrity
2. High Degree of Flexibility in deployment platform and configurations
3. Improved security
4. High Performance and persistent objects
5. Architecture is scalable, adding users and resources in future would be easy
6. Maintenance and modifications can be done effectively
7. Code and data reusability can be achieved

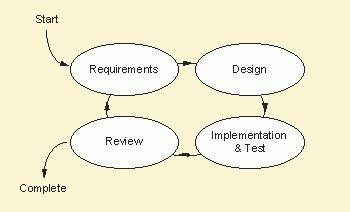
* **Building or Developing the Product** - In this stage of SDLC the actual development starts and the product is built. The programming code is generated as per DDS during this stage. If the design is performed in a detailed and organized manner, code generation can be accomplished without much hassle.
* **Testing the Product** - This stage is usually a subset of all the stages as in the modern SDLC models, the testing activities are mostly involved in all the stages of SDLC.
* **Deployment in the Market and Maintenance** - Once the product is tested and ready to be deployed it is released formally in the appropriate market.

## SDLC Models

There are various software development life cycle models defined and designed which are followed during the software development process. These models are also referred as Software Development Process Models".

### **Iterative Model**

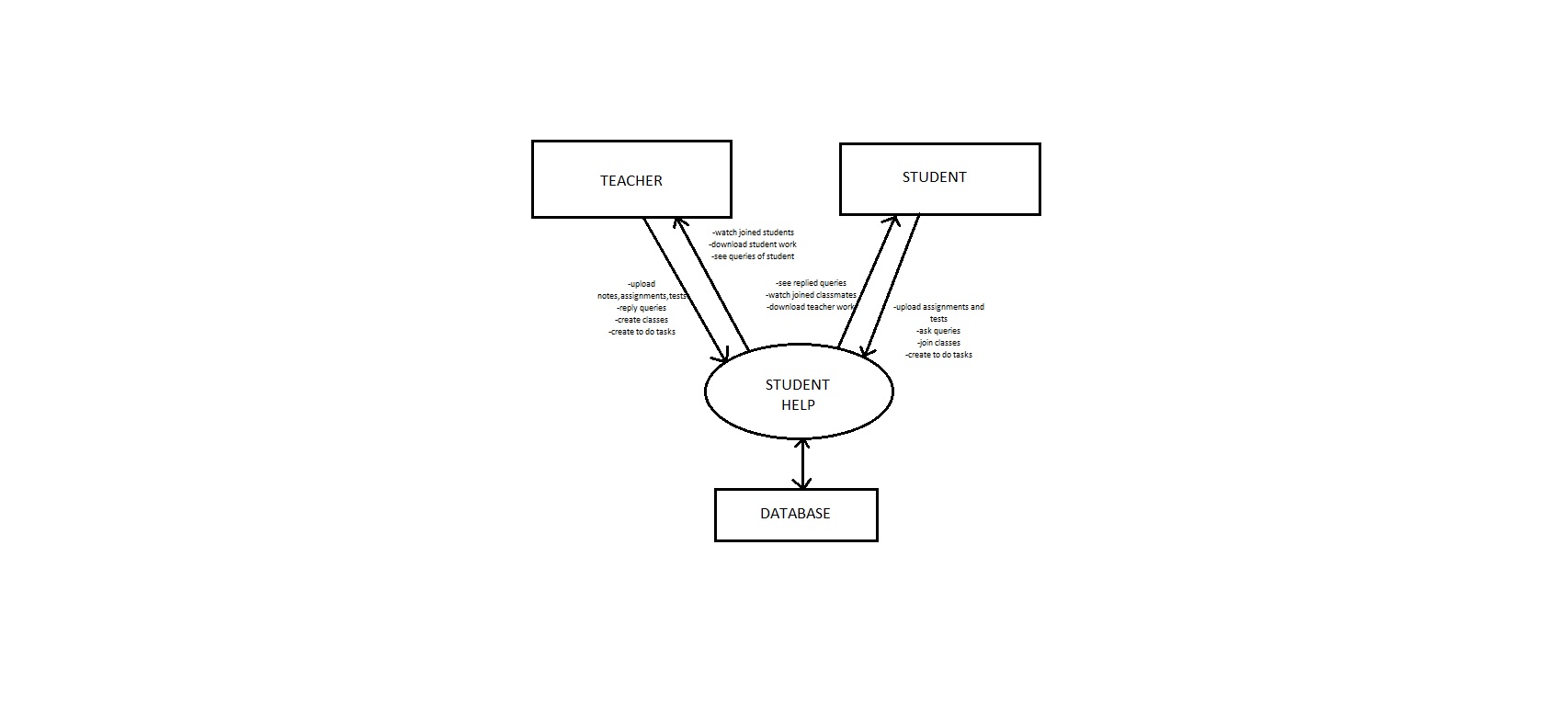
An iterative lifecycle model does not attempt to start with a full specification of requirements. Instead, development begins by specifying and implementing just part of the software, which can then be reviewed in order to identify further requirements. This process is then repeated, producing a new version of the software for each cycle of the model. Consider an iterative lifecycle model which consists of repeating the following four phases in sequence:



**Fig-2: Iterative Model**

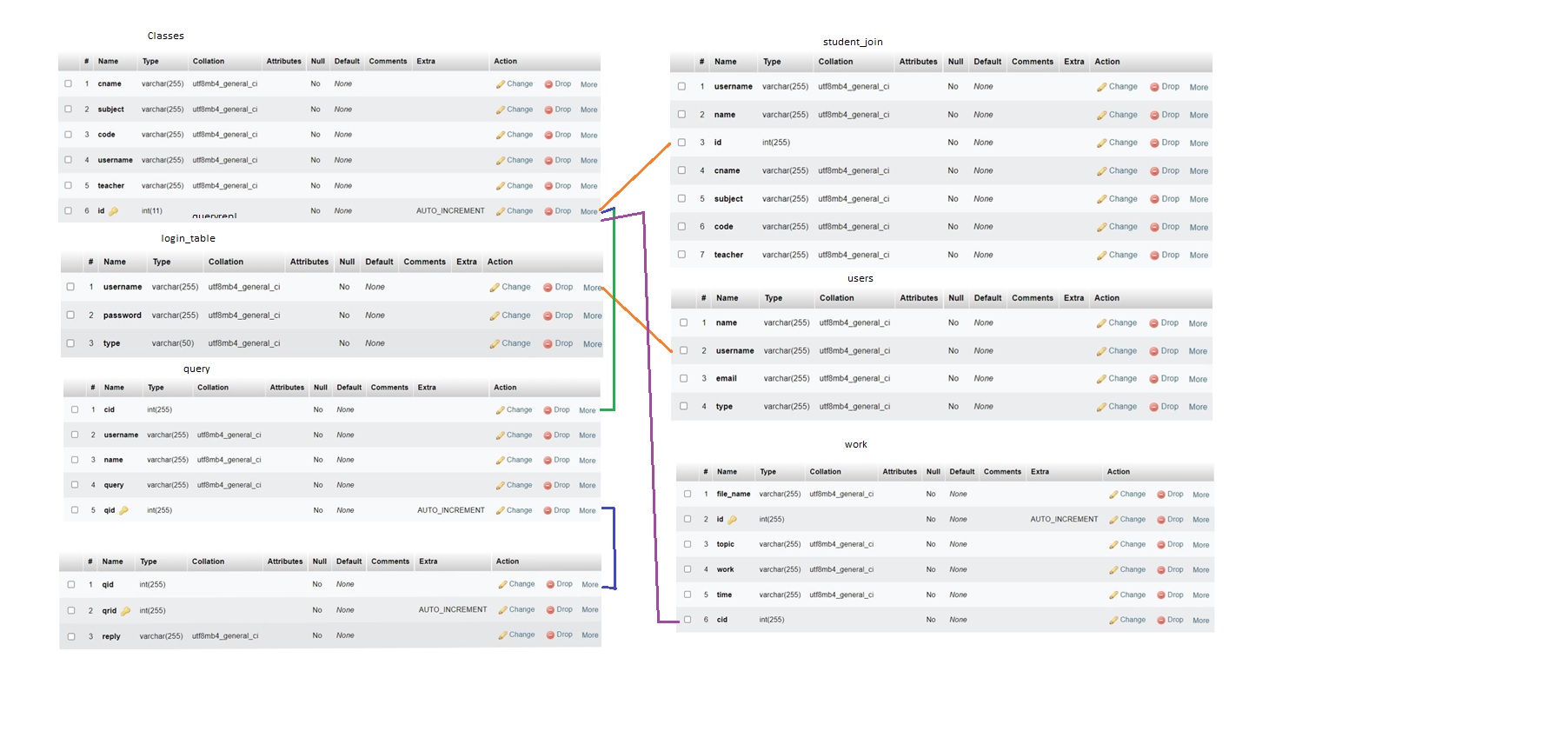
1. A ***Requirements*** phase, in which the requirements for the software are gathered and analysed. Iteration should eventually result in a requirements phase that produces a complete and final specification of requirements.
2. A ***Design*** phase, in which a software solution to meet the requirements is designed. This may be a new design, or an extension of an earlier design.
3. An ***Implementation and Test*** phase, when the software is coded, integrated and tested.
4. A ***Review*** phase, in which the software is evaluated, the current requirements are reviewed, and changes and additions to requirements proposed.
5. For each cycle of the model, a decision has to be made as to whether the software produced by the cycle will be discarded, or kept as a starting point for the next cycle (sometimes referred to as incremental prototyping). Eventually a point will be reached where the requirements are complete and the software can be delivered, or it becomes impossible to enhance the software as required, and a fresh start has to be made.
6. The iterative lifecycle model can be likened to producing software by successive approximation. Drawing an analogy with mathematical methods that use successive approximation to arrive at a final solution, the benefit of such methods depends on how rapidly they converge on a solution.
7. The key to successful use of an iterative software development lifecycle is rigorous validation of requirements, and verification (including testing) of each version of the software against those requirements within each cycle of the model. The first three phases of the example iterative model is in fact an abbreviated form of a sequential V or waterfall lifecycle model. Each cycle of the model produces software that requires testing at the unit level, for software integration, for system integration and for acceptance. As the software evolves through successive cycles, tests have to be repeated and extended to verify each version of the software.

**DATA FLOW DIAGRAM**



**Fig-3: Data Flow Diagram**

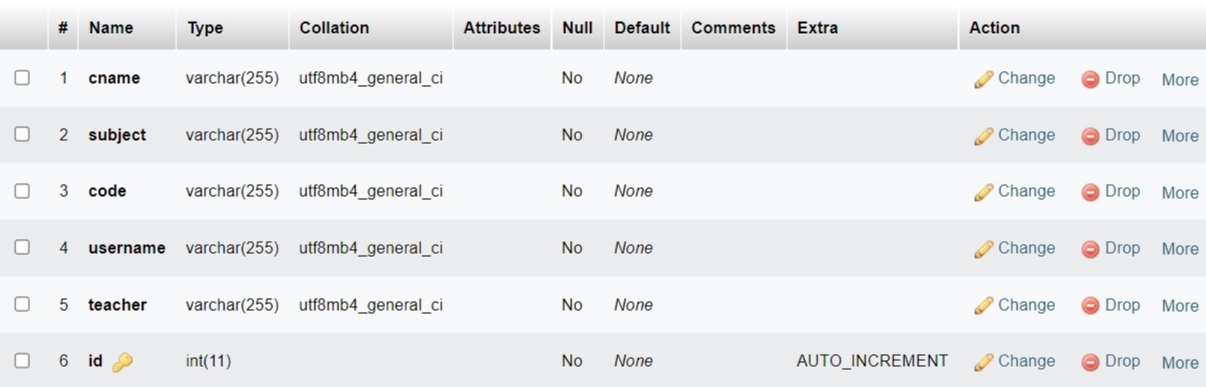
**ER-DIAGRAM**



**Fig-4: ER-Diagram**

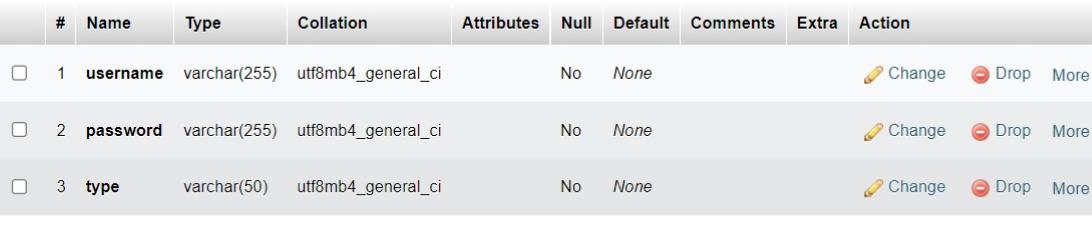
**DATA-SCHEMA**

**Classes:** Classes table is used to store the data of class which is created by the Teacher for the Student of each subject. It also contain unique id which prefer the classes uniquely.



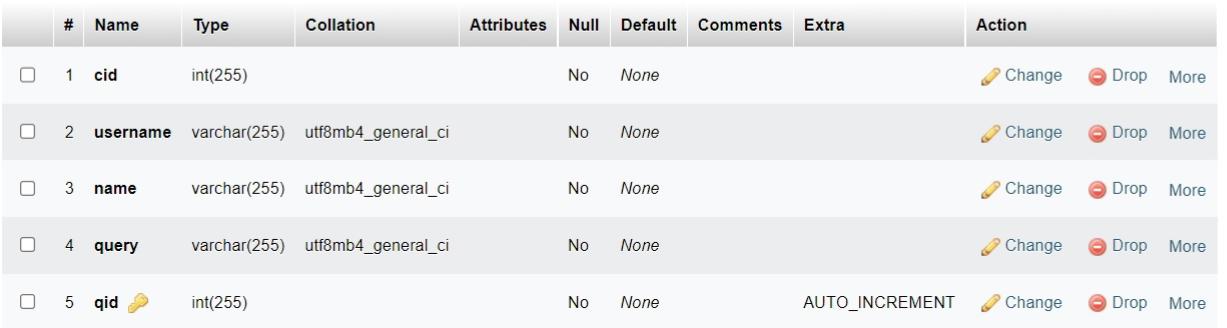
**Fig-5: Classes Table**

**Login\_table:** Login table store the data of users which helps them to log in the student login page or Teacher login page according to their type.



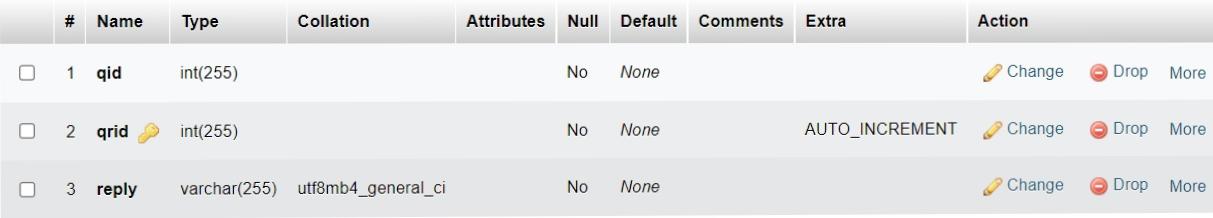
**Fig-6: Login Table**

**Query:** Query table store the data of student means which student ask the query to the teacher and it also store query in it. This Table also contain unique id which prefer the students uniquely.



**Fig-7: Query Table**

**Queryreply:** Queryreply Table store the reply of teacher to the query which is asked by the student to their teacher.



**Fig-8: Queryreply Table**

**Student\_join:** Student\_join Table used to store the data of classes and students means which student join the different classes according to their subjects.



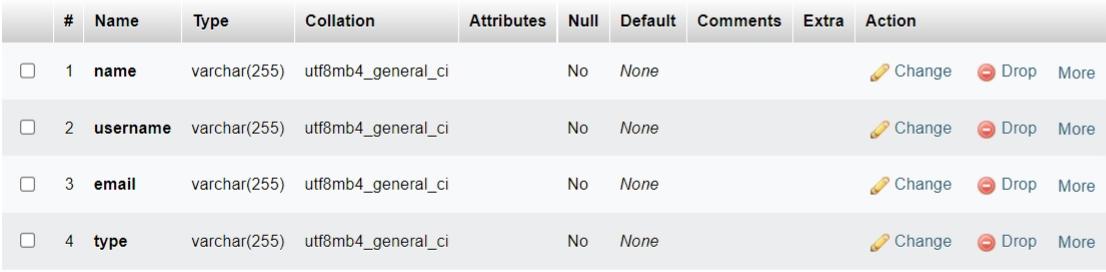
**Fig-9: Student join Table**

**Tasks:** Tasks Table store the data of To-do List which is written by the Teachers and students in their To-do list which is uniquely prefer by the id.

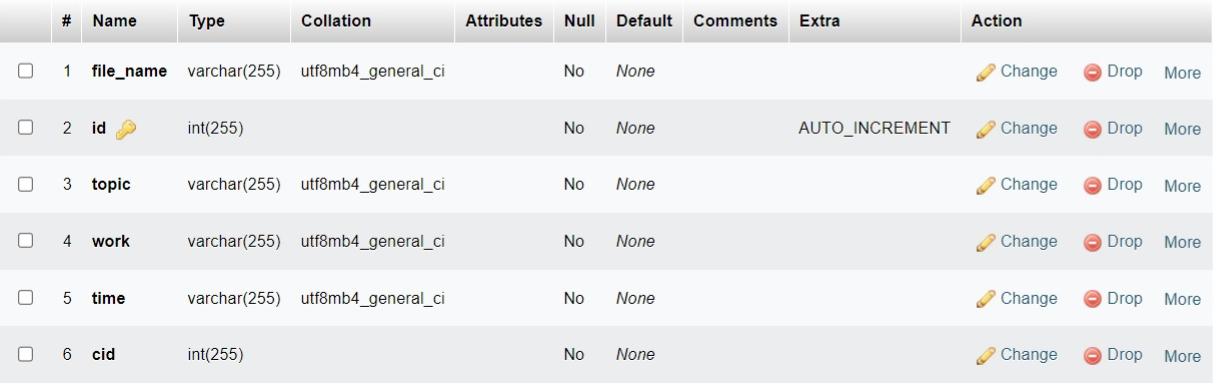


**Fig-10: Tasks Table**

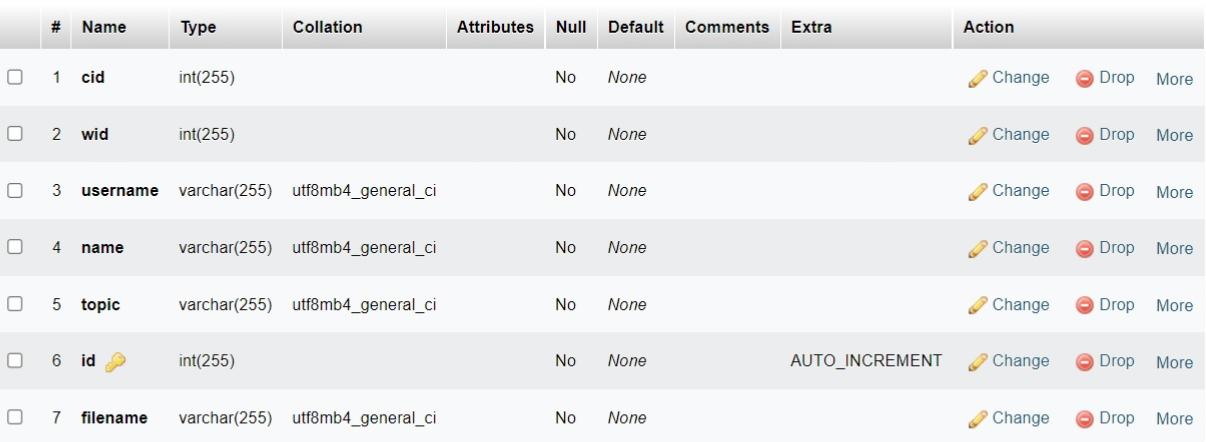
**Users:** Users Table store the all information of users which

**Fig-11: Users Table**

**Work:** Work Table store the work which is given by the teachers to their students in their subjects.

**Fig-12: Work Table**

**Work\_upload:** Work\_upload store the submitted work by the student in according to their subject. It also store the all information means which student upload which work.



**Fig-13: Work upload Table**

**FEASIBILITY STUDY:**

After doing the project Student Help Website, study and analysing all the existing or required functionalities of the system, the next task is to do feasibility study for the project.

The goal of feasibility study is to evaluate alternative systems to propose the most feasible and desirable systems for development.

**Below written are the Feasibility Study of our Project:**

1. **Operational Feasibility:**

The operational feasibility is responsible for the operations of management, Teachers, and students involved in a project.

Our Project (Student Help Website) is operationally feasible because the features we have considered such as Downloading, And Uploading of pdfs provides a combined platform by using which the related tasks can be handled in an easy manner.

1. **Technical Feasibility:**

Technical feasible can be demonstrated if reliable hardware and software capable of meeting needs of proposed system can be developed or acquired by the business in required time.

Our Project (Student Help Website) is Technically feasible because the required hardware and software which are needed to make the Project are available.

1. **Economic Feasibility:**

This is a very important aspect to be considered while developing a Project.

We decided the technology based on minimum possible cost factor on our Project. All Hardware and Software cost is efficient and Our Project (Student Help Website) is free for all the Users.

**PROJECT DETAILS**

**INTRODUCTION:**

Student Help Website project based upon full stack development. In the modern age of online education, this website helps students in their education. Student Help Website is online tool that allows Teachers to set Assignments, have work submitted by students, to mark, and to return graded papers. It was initially planned for use with laptops in schools, such as Chromebooks, in order to allow the teachers and students to more efficiently share assignments and information.

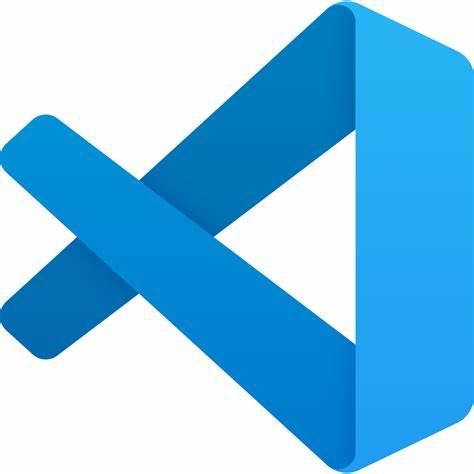
Student Help Website is very beneficial for Students and Teachers for managing their School Work online. In this Website Teacher can create their different classes according to their respective subjects. Teacher can also set the different subject code in their classes according to their subjects.

Teacher can upload tests and assignments with due dates with their students. Teachers can also see their students queries about the subjects and answer them.

Students can submit their work online to their teachers. Students can also write the queries to their Teachers according to the problem in their work.

**SOFTWARE USED:**

**VS Code:**



Visual Studio (VS) Code is an open-source code editor primarily used to correct and repair cloud and [**web applications**](https://www.webopedia.com/definitions/web-application/) coding errors. VS Code is developed by [**Microsoft**](https://www.microsoft.com/en-ng) and supports the [**macOS**](https://www.webopedia.com/definitions/mac-os/), [**Linux**](https://www.webopedia.com/definitions/linux/), and [**Windows**](https://www.webopedia.com/definitions/microsoft-windows/) operating systems. VS Code’s tools can be used to enhance the functionality of any written code. Based on the [Electron framework](https://www.electronjs.org/), VS Code employs the same editor component used in Azure DevOps.

**XAMPP Server:**



XAMPP is a cross-platform web server that is free and open-source. XAMPP is a short form for Cross-Platform, Apache, MySQL, PHP, and Perl. XAMPP is a popular cross-platform web server that allows programmers to write and test their code on a local webserver. It was created by Apache Friends, and the public can revise or modify its native source code. It includes MariaDB, Apache HTTP Server, and interpreters for PHP and Perl, among other computer languages.

**HARDWARE & SOFTWARE REQUIREMENTS :**

**Hardware Requirements Server:**

1. Intel i3/i5/i7 processor
2. 4GB/8GB/12GB RAM
3. Optical Drive
4. Internet Connection

**Software Requirements Server:**

1. Server Based Operating System
2. Apache Server
3. Node JS 18.7.0
4. AJAX Support

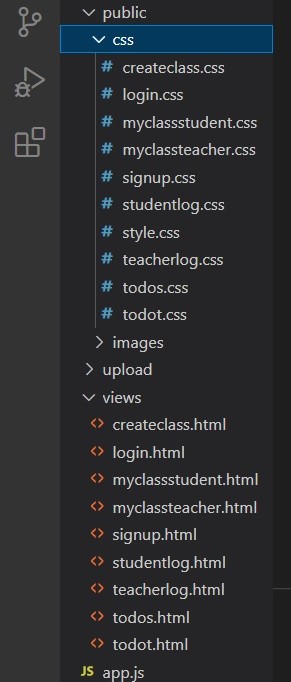
**Hardware Requirements User:**

1. Intel i3/i5/i7 processor
2. 4GB/8GB/12GB RAM
3. Internet Connection

**Software Requirements User:**

1. Any Operating System
2. Web Browser

**STRUCTURE OF THE PROJECT**

****

**Fig-14: Project Folder**

Inside of the folder of the main project locks like this. There is a app,js file which is the server file and is responsible for majority of work performed in the project. Then there are three folders- public, upload and views

Views folder contains the all the html files which are routed using the app.js file. And all the files which are attachments in these html files are stored in public folder. Public folder has two subfolders for different category of files - CSS and images. All the CSS and images used are in public folder. The last folder which remains is upload folder. In which uploaded files from students and teachers will appear and can be downloaded when needed in the program from upload folder.

**FEATURES:**

In Student Help Website there will be two types of users, who can login from the home page and can get access to different features according to user:

* Teacher
* Students

**Features provided to Teachers:**

1. Teacher login
2. Edit their details
3. CRUD operations on Classes
4. Can upload notes for students
5. Can upload assignments and tests with due dates
6. CRUD operations on assignments notes and tests
7. Can view students queries and reply them
8. Use the **to-do list** feature

**Features provided to Students:**

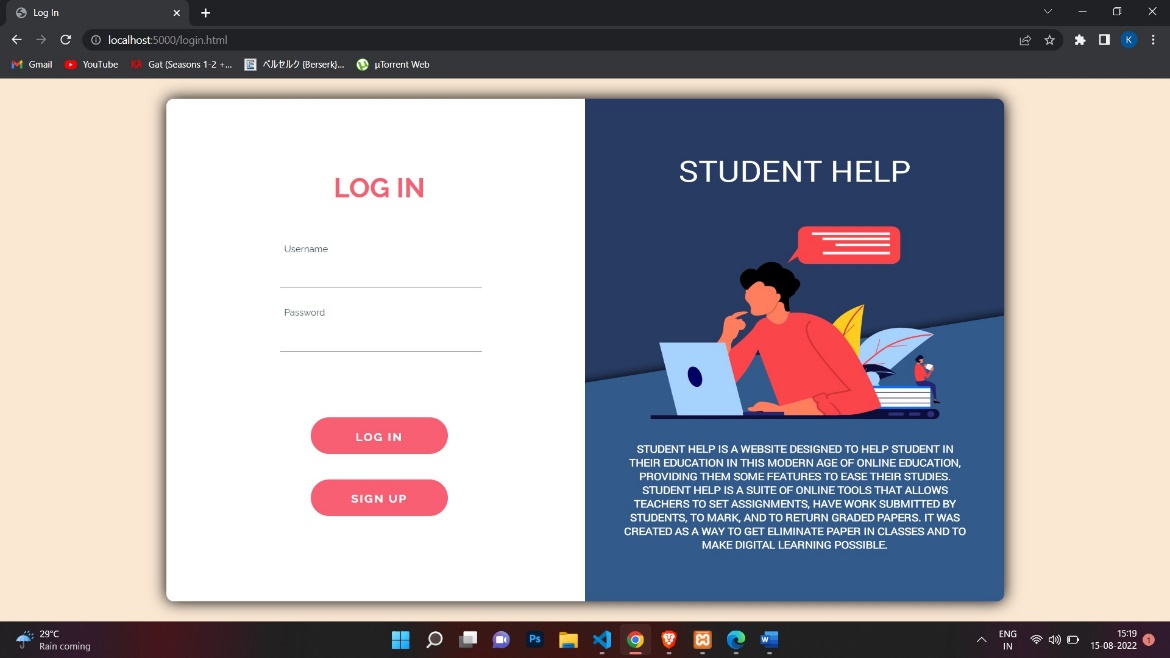
1. Student login
2. Edit their details
3. Join different classes
4. Can see different notes uploaded by teacher
5. Can submit their assignments and tests
6. Can ask queries about subject
7. Use the **to-do-list** feature

**BENEFITS:**

1. **Easy to use:** Student Help Website is very easy to use. It is not complex everything is readable and structured in clean format.
2. **Cost Efficient:** It is free for every user like Teacher and Students.
3. **Online Tool:** It is an online tool which is very helpful for every user. They can use them in their home without any difficulties. Teachers and Students can upload and submit their work online on it.
4. **Queries:** In this Website we have Query Section which help Students to write their every Query in it. After receiving Query from Student Teacher can answer these queries also.
5. **To-do List:** To-do List is provided to the both Teachers and Students to write their tasksand delete them after the completion of the tasks. This feature helps both the users to remember their tasks which were pending.
6. **Privacy:** Privacy means Assignment which were submit by the Students to their Teachers according to their work were only seen by the Teachers. Other Students are not able to sees the submitted assignments of other Students.
7. **CRUD:** In this website we provide CRUD operation only to the Teacher. They can Create, Read, Update and Delete the data of themselves and Students too.
8. **Different Codes:** In this website we provide different codes on every subject which is create by the Teachers.

**PROJECT SCREENISHOTS WITH EXPLANATION**

**LOGIN PAGE**

****

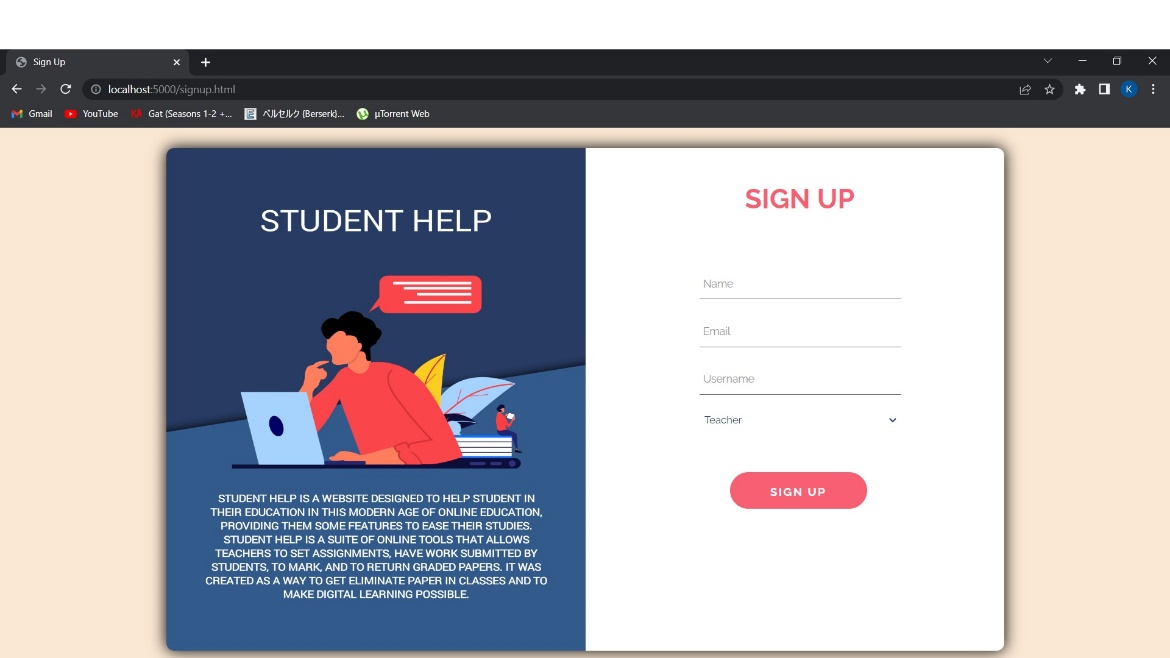
**Fig-15: Login Page**

A login page from where teacher and student can login. A student will automatically be logged to student log page and a teacher to teacher log page.

There is a signup button on pressing which a new user can create its teacher or student account for login. Errors will be displayed if a user enters wrong password or username.

We used a localhost 5000, which means our server will communicate which webpage when its open in any browser with this port. The server will run within the system.

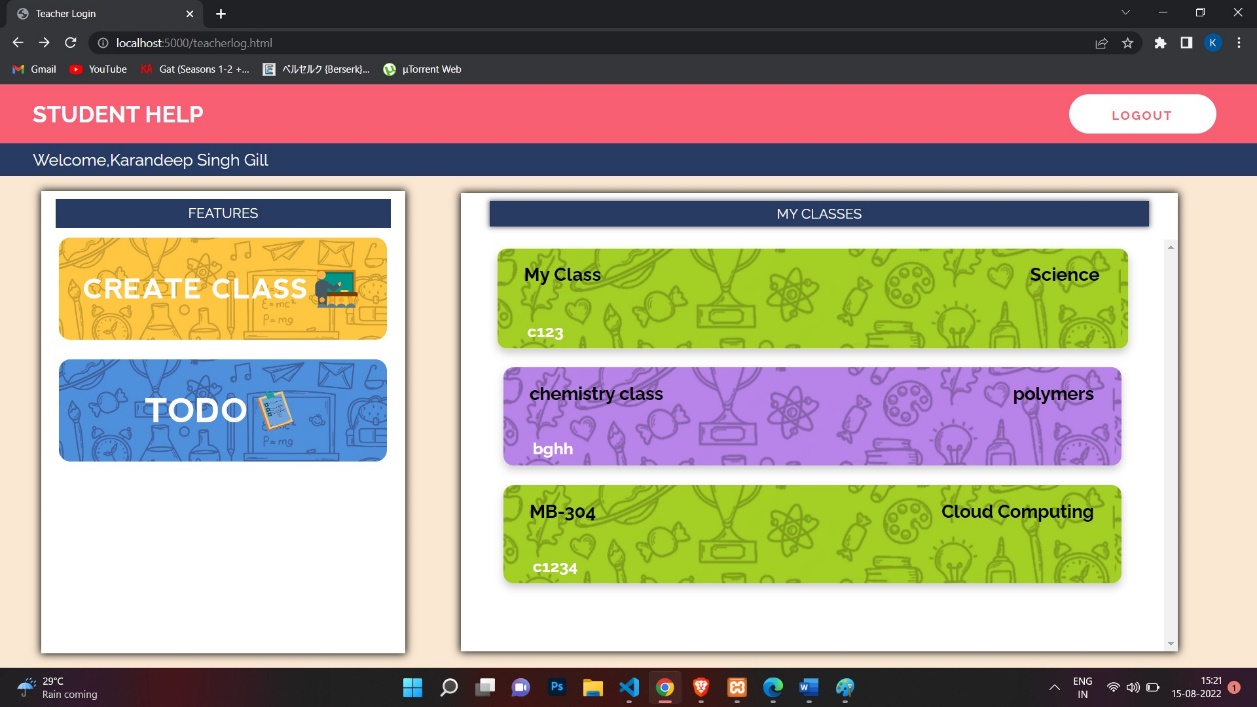
**SIGNUP PAGE**



**Fig-16: SignUp Page**

In signup page a user can create account of a student or a teacher with the requirements of name username and email. On pressing the signup button the server will check if the username entered already exists or not. If yes an error will be shown. Otherwise the user account will be created and the password for the user account will be randomly generated for the time and will be sent to the email provided by the user. After this the user will be automatically redirected to the login page.

**TEACHER LOGIN**

****

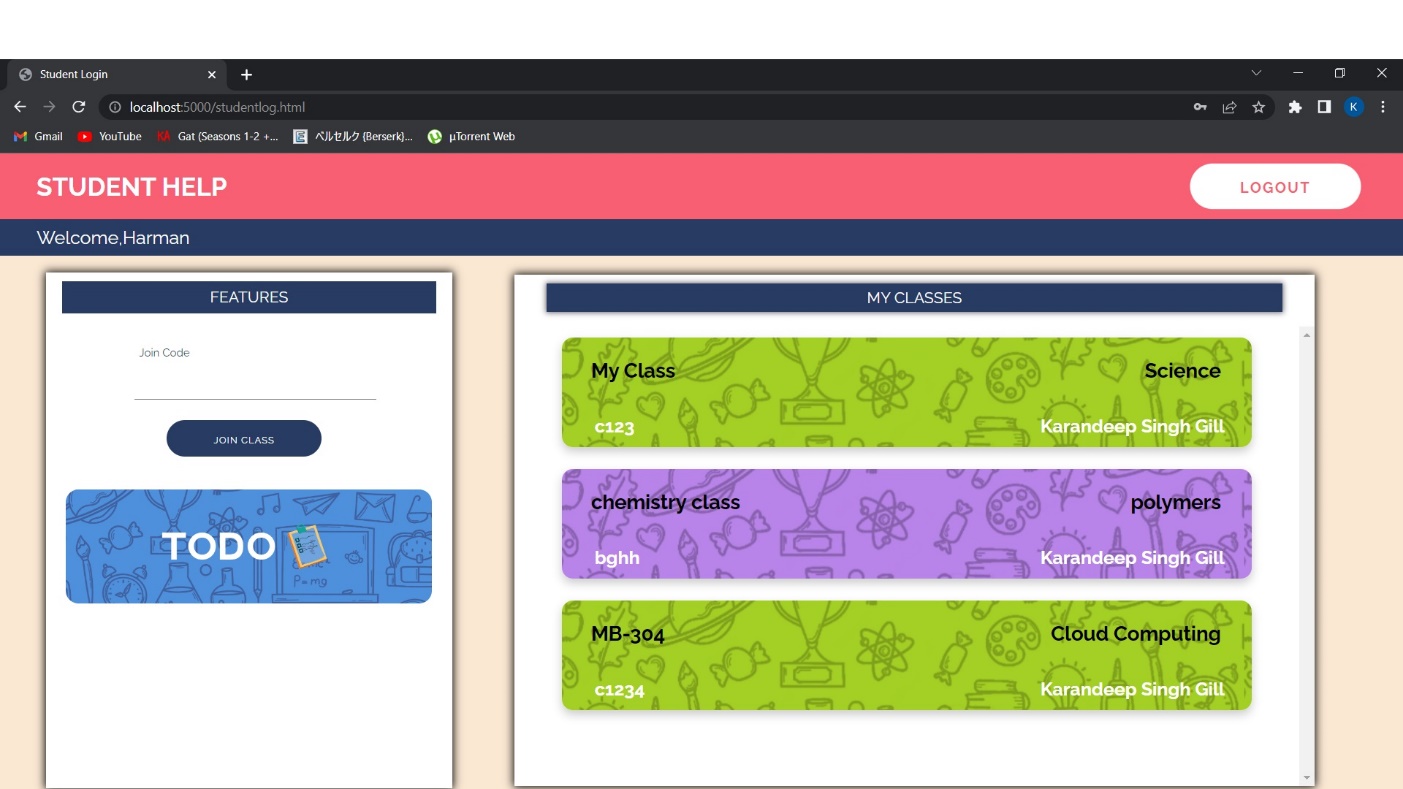
**Fig-17: Teacher Login Page**

When a teacher logs this window will appear. In this window first there is a button to logout the current account and below that there is a tab where the name of the teacher is displayed. There is a features section and my classes section. Under features section there are currently two features – create class and to do. Using create class, the teacher can create a class for students to interact with them. Using to do, the teacher can create a list of tasks and delete them as per requirement.

Under my classes section there will be classes created by the teacher. On clicking on a certain class the webpage respective to it will open. Details of the class are displayed on the class button.

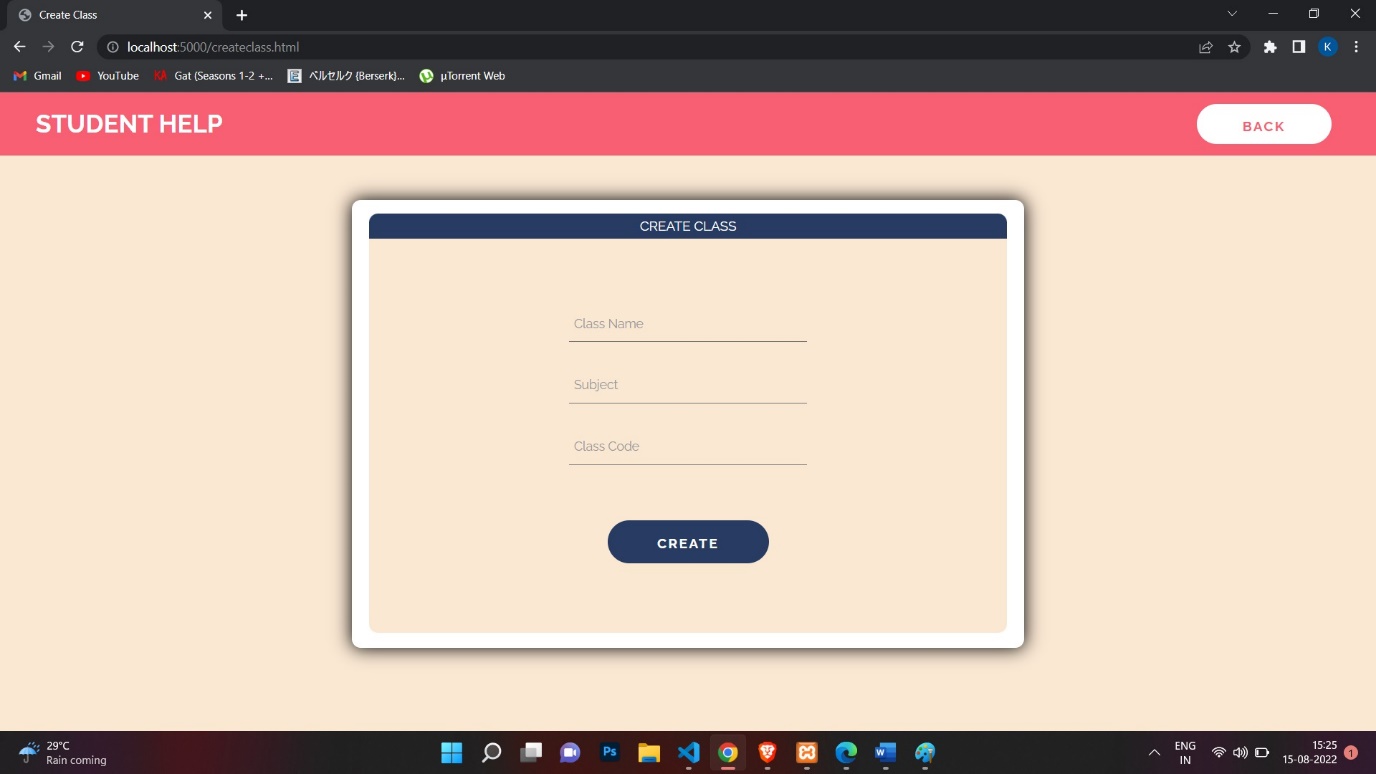
**STUDENT LOGIN**

This is the window which will be displayed when a student logs in. The interface work of this window is quit similar to the teacher login except the join class part. In student login there is join class input field instead of create class button. In this input field the student can enter the unique code of a class to join that class. An error will be displayed if the class code is wrong. Also the name of the teacher is shown in the details of the class button.



**Fig-18: Student Login Page**

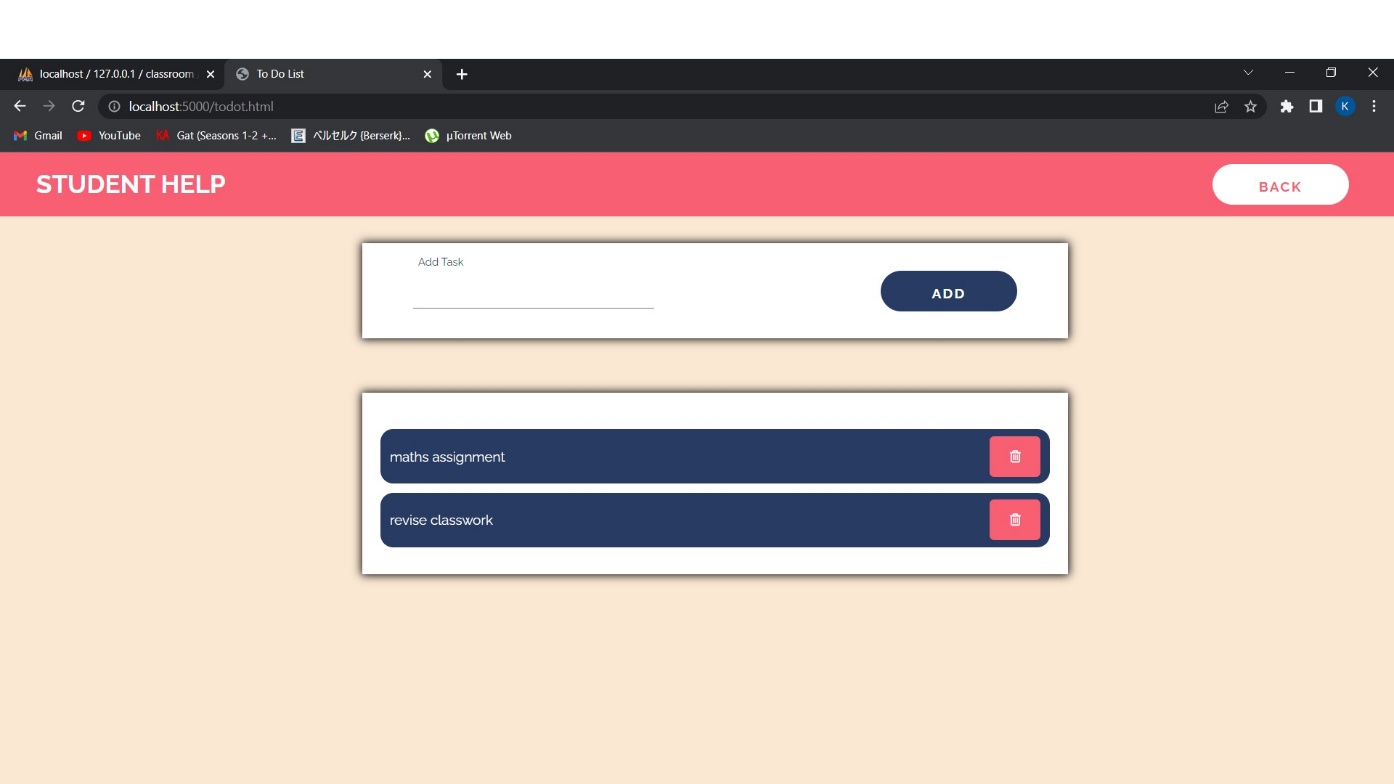
**CREATE CLASS**

****

**Fig-19: Create Class Page**

When a teacher clicks on create class button on teacher login page this window will appear. In this window teacher can create class by filling the details like subject, name and class code. A class code is unique , if code entered already exists there will be an error message.

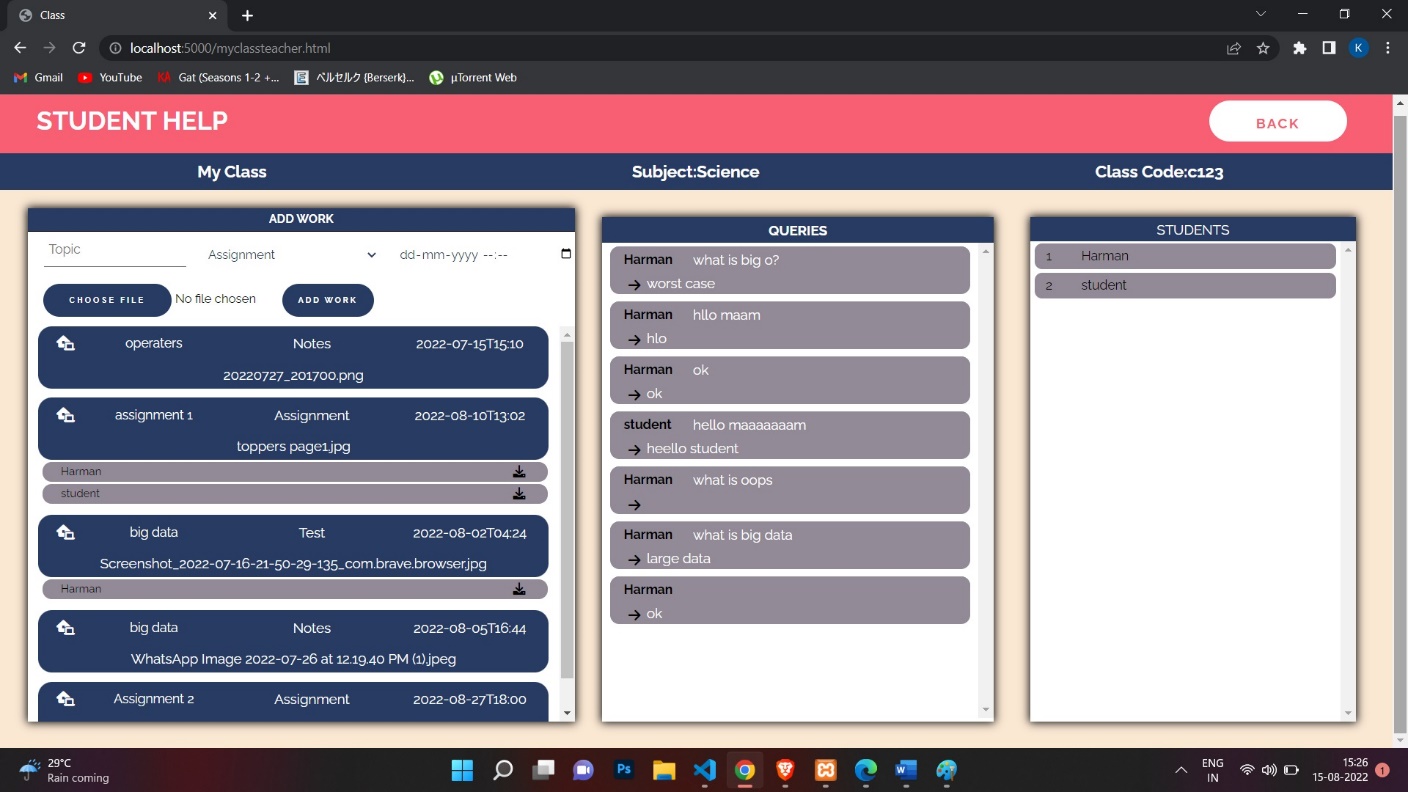
**TO-DO PAGE**



**Fig-20: To-Do Page**

On clicking on the to do features by any student or teacher this window will appear along with their tasks which they have created. User can create tasks and the task will be added to the list along with the delete button. When a task is completed the user can simply delete the respective task by clicking on trash bin icon. There is also a back button on top to go back to respective login page.

**TEACHER CLASS**

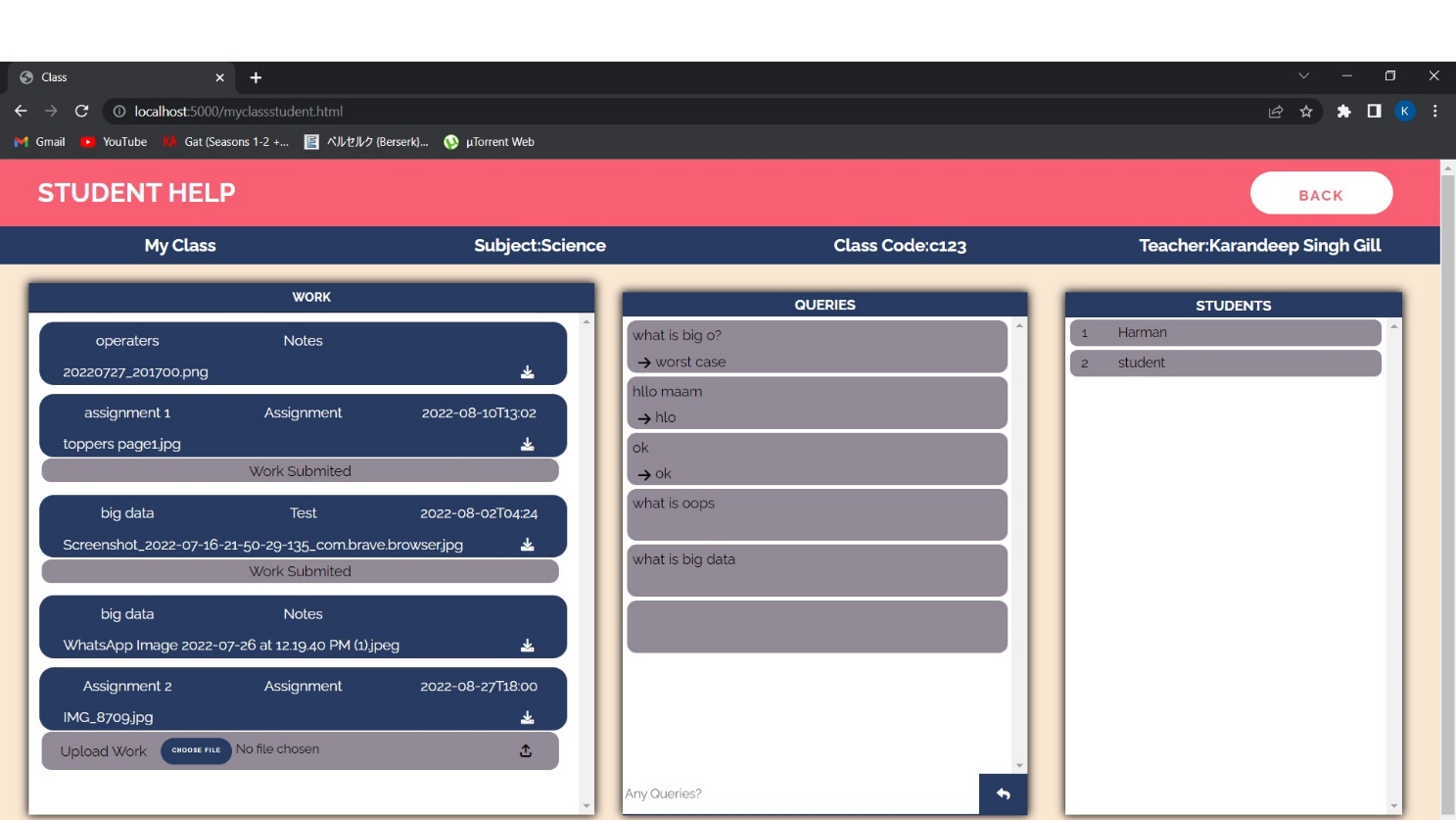


**Fig-21: Teacher Class Page**

On clicking on any class by a teacher such webpage will appear. On top there is back button as usual.

Underneath that there is tab which displays the details of the class teacher has opened. Then the main part of the page is divided into three sections add work, query, students. In add work section teacher can upload any file regarding to a assignment, test or note along which topic name and a deadline, if everything is ok then the work will be uploaded to class and can be viewed by the students who have joined the class. If any student uploads the assignment or test respective to a work then it will be shown underneath that work and teacher can download that. In query section the teacher can reply to queries asked by students who have joined the class all the queries will be displayed in this section. Under the student section there will be the list of students who has joined the class with their names.

**STUDENT CLASS**



**Fig-22: Student Class Page**

This the page which will appear when a student opens a class. Back button and detail part is same as

Teacher class page but the name of teacher is displayed along with it. One section is work in which work uploaded by the teacher is displayed along with the download button to download the file upload by the teacher and details of the work. If the work type is notes, no sub section will be shown under a work. But if the work is assignment or test there will be a sub section to select the file upload the work asked by the teacher. If student has submitted the work the phrase “work submitted” will be displayed.

Then there is query section under which student can ask queries to teacher regarding the subject and

the replies from previous queries from teacher will be shown under this section. Then there is last section of student same as the teacher class page.

**CODE OF THE PROJECT**

**SERVER CODE OF NODE JS FILE**

var express = require("express");

var app = express();

var fileUpload = require("express-fileupload");

const fs = require("fs");

var nodemailer = require('nodemailer');

var mysql = require("mysql");

var db = mysql.createConnection({

    host: "localhost",

    user: "root",

    password: '',

    database: "classroom"

});

db.connect(function (error) {

    if (error) throw error;

    console.log("Database Connection Created");

});

app.use(fileUpload());

app.use(express.static("views"));

app.use(express.static("public"));

app.use(express.static("upload"));

app.use(express.urlencoded({extended:true}));

app.post("/load-sname",(req,res)=>{

  var a = req.body.username;

  var query = `SELECT name,email FROM users WHERE username= '${a}'`;

  db.query(query,function(error,data){

    if(error){

      throw error;

    }else{

      res.send(data);

    }

  });

  })

app.post("/load-tname",(req,res)=>{

var a = req.body.username;

var query = `SELECT cname,teacher FROM classes WHERE username= '${a}'`;

db.query(query,function(error,data){

  if(error){

    throw error;

  }else{

    res.send(data);

  }

});

})

app.post("/replied-query",(req,res)=>{

var query = `SELECT qid,qrid,reply FROM queryreply `;

db.query(query,function(error,data){

  if(error){

    throw error;

  }else{

    res.send(data);

  }

});

})

app.post("/reply-query",(req,res)=>{

var a = req.body.qid;

var b = req.body.reply;

var query  = `INSERT INTO queryreply (qid,reply) VALUES ('${a}', '${b}')`;

db.query(query,function(error){

  if(error){

    throw error;

  }else{

    res.send("done");

  }

});

})

app.post( "/show-teacher-queries",(req,res)=>{

var a = req.body.id;

var query = `SELECT query,qid,name FROM query WHERE cid = '${a}'`;

db.query(query,function(error,data){

  if(error){

    throw error;

  }else{

    res.send(data);

  }

});

});

app.post("/show-query",(req,res)=>{

  var a = req.body.cid;

  var b  = req.body.username;

  var query = `SELECT query,qid FROM query WHERE username ='${b}' AND cid = '${a}'`;

  db.query(query,function(error,data){

    if(error){

      throw error;

    }else{

      res.send(data);

    }

  });

})

app.post("/ask-query",(req,res)=>{

var a = req.body.cid;

var b  = req.body.username;

var c  = req.body.query;

var query = `SELECT name,email FROM users WHERE username ='${b}'`;

db.query(query,function(error,data){

  if(error){

    throw error;

  }else{

    var query2 =`INSERT INTO query (cid,username,name,query) VALUES ('${a}', '${b}','${data[0].name}','${c}')`;

    db.query(query2,function(error){

      if(error){

        throw error;

      }else{

        res.send("done");

      }

    });

  }

});

})

app.get("/stu-download-work",(req,res)=>{

  var a = req.query.nothing;

  res.download(\_\_dirname + "/upload/" + a, function (err) {

    if (err) {

      throw error;

    }

  });

  });

app.post( "/download-stu-work",(req,res)=>{

var a  = req.body.id;

var query =  `SELECT wid,name,filename FROM work\_upload WHERE cid ='${a}' ORDER BY wid`;

db.query(query,function(error,data){

  if(error){

    throw error;

  }else{

    res.send(data);

  }

});

})

app.post("/work-submit",(req,res)=>{

var a  = req.body.cid;

var b  = req.body.username;

var query =  `SELECT wid,name,id FROM work\_upload WHERE username ='${b}' AND cid = '${a}'`;

db.query(query,function(error,data){

  if(error){

    throw error;

  }else{

    res.send(data);

  }

});

});

app.get("/stu-download",(req,res)=>{

var a = req.query.nothing;

res.download(\_\_dirname + "/upload/" + a, function (err) {

  if (err) {

    throw error;

  }

});

});

app.post("/stu-upload",(req,res)=>{

if (req.files && Object.keys(req.files).length &&req.body.topic&&req.body.username&&req.body.cid&&req.body.wid !== 0) {

    var uploadedFile = req.files.myfile;

    var uploadPath = \_\_dirname

        + "/upload/" + uploadedFile.name;

    uploadedFile.mv(uploadPath, function (err) {

      if (err) {

        console.log(err);

      } else {

        var uploadedFile = req.files.myfile;

        var filename = uploadedFile.name;

        var topic = req.body.topic;

        var wid = req.body.wid;

        var user = req.body.username;

        var cid = req.body.cid;

        var query = `SELECT name,email FROM users WHERE username ='${user}'`;

        db.query(query,function(error,data){

          if(error){

            throw error;

          }else{

            var query1 =`INSERT INTO work\_upload (cid,wid,username,name,topic,filename) VALUES ('${cid}', '${wid}','${user}','${data[0].name}','${topic}','${filename}')`;

        db.query(query1,function(error){

          if(error){

            throw error;

          }else{

            res.send("done");

          }

        });

      };

    });

  }});} else res.send("Enter the all Information!");

});

app.post("/load-content",(req,res)=>{

var a = req.body.id;

var query = `SELECT file\_name,topic,work,time,id FROM work WHERE cid ='${a}'`;

db.query(query,function(error,data){

  if(error){

    throw error;

  }else{

    res.send(data);

  }

});

});

app.post("/show-content",(req,res)=>{

var a = req.body.id;

var query = `SELECT file\_name,topic,work,id,time FROM work WHERE cid ='${a}'`;

db.query(query,function(error,data){

  if(error){

    throw error;

  }else{

    res.send(data);

  }

});

});

app.post("/work-upload",(req,res)=>{

  if (req.files && Object.keys(req.files).length &&req.body.topic&&req.body.work&&req.body.time !== 0) {

    var uploadedFile = req.files.myfile;

    var uploadPath = \_\_dirname

        + "/upload/" + uploadedFile.name;

    uploadedFile.mv(uploadPath, function (err) {

      if (err) {

        console.log(err);

      } else {

        var uploadedFile = req.files.myfile;

        var filename = uploadedFile.name;

        var topic = req.body.topic;

        var work = req.body.work;

        var time = req.body.time;

        var cid = req.body.cid;

        var query =`INSERT INTO work (file\_name,topic,work,time,cid) VALUES ('${filename}', '${topic}','${work}','${time}','${cid}')`;

        db.query(query,function(error){

          if(error){

            throw error;

          }else{

            res.send("done");

          }

        });

      };

    });

  } else res.send("Enter the all Information!");

});

app.post("/load-stu" ,(req,res)=>{

var cid=req.body.id;

var query = `SELECT name,subject FROM student\_join WHERE id ='${cid}'`;

db.query(query,function(error,data){

  if(error) throw error;

  if(data.length>0){

    res.send(data);

  }

});

});

app.post("/load-classes-stu",(req,res)=>{

  var a = req.body.username;

  var classquery = `SELECT id,cname,subject,code,teacher FROM student\_join WHERE username ='${a}'`;

  db.query(classquery,function(error,data){

    if(error) throw error;

    if(data.length>0){

          res.send(data);

    }

  });

})

app.post("/join-class",(req,res)=>{

var a = req.body.code;

var b = req.body.username;

var query = `SELECT id,cname,subject,code,teacher FROM classes WHERE code ='${a}'`;

db.query(query,function(error,data){

  if(error) throw error;

  if(data.length>0){

  var query2 = `SELECT name FROM student\_join WHERE username ='${b}' AND id = '${data[0].id}'`;

  db.query(query2,function(error,data2){

    if(error) throw error;

    if(data2.length>0){

      res.send("Class Already Joined");

    }else{

      var query3 = `SELECT email,name FROM users WHERE username ='${b}'`;

      db.query(query3,function(error,data3){

        if(error) throw error;

        if(data3.length>0){

          var query4 = `INSERT INTO student\_join (username,name,id,cname,subject,code,teacher) VALUES ('${b}', '${data3[0].name}','${data[0].id}','${data[0].cname}','${data[0].subject}','${data[0].code}','${data[0].teacher}')`;

          db.query(query4,function(error){

            if(error){

              throw error;

            }else{

              res.send("Class Joined");

            }

          });

        }

      });

    }

  });

   }else{

    res.send("Wrong Class Code");

   }

});

});

app.post("/load-data",(req,res)=>{

var a = req.body.id;

var query = `SELECT cname,subject,code,teacher FROM classes WHERE id ='${a}'`;

db.query(query,function(error,data){

  if(error) throw error;

  if(data.length>0){

    res.send(data);

   }

})

});

app.post("/load-classes",(req,res)=>{

  var a = req.body.username;

  var classquery = `SELECT cname,subject,code,id,teacher FROM classes WHERE username ='${a}'`;

  db.query(classquery,function(error,data){

    if(error) throw error;

    if(data.length>0){

      res.send(data);

     }

  })

})

app.post("/todo-action",(req,res)=>{

var a = req.body.username;

var taskquery = `SELECT task,id FROM tasks WHERE username ='${a}'`;

db.query(taskquery,function(error,data){

 if(error) throw error;

 if(data.length>0){

  res.send(data);

 }

});

});

app.post("/addtask",(req,res)=>{

var a = req.body.username;

var b = req.body.task;

console.log(a);

console.log(b);

var atask = `INSERT INTO tasks (task, username) VALUES ('${b}', '${a}')`;

db.query(atask,function(error){

  if(error) {

    throw error;

  }else{

    res.send("ok");

  }

})

})

app.post("/deletetask",(req,res)=>{

  var a = req.body.id;

  var dtask = `DELETE FROM tasks WHERE id = '${a}'`;

  db.query(dtask,function(error){

    if(error){

      throw error;

    }else{

      res.send("ok");

    }

  })

})

app.post("/class-signup",(req,res)=>{

var a = req.body.cname;

var b = req.body.subject;

var c = req.body.username;

var d = req.body.classcode;

var selectquery=`SELECT code FROM classes WHERE code ='${d}'`;

db.query(selectquery,function(error,rows,fields){

  if(error) throw error;

  if(rows.length<=0){

    var selectquery2=`SELECT name FROM users WHERE username ='${c}'`;

    db.query(selectquery2,function(error,rows,fields){

      if(error) {

        throw error;

      }else{

        var insertSQL = `INSERT INTO classes(cname, subject, code,username,teacher) VALUES('${a}','${b}','${d}','${c}','${rows[0].name}')`;

        db.query(insertSQL,function(error){

          if(error) {

            throw error;

          }else{

           res.send("ok");

          }

        })

      }

    })

  }else{

    res.send("Code Already Exists");

  }

})

});

app.post("/signup-action", (req, res) => {

console.log("active");

    var a = req.body.name;

    var b = req.body.email;

    var c = req.body.username;

    var f = req.body.user;

    var selectquery=`SELECT username FROM login\_table WHERE username ='${c}'`;

    db.query(selectquery,function(error,rows,fields){

        if(error) throw error;

        if(rows.length<=0){

          function generateP() {

            var pass = '';

            var str = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ' +

                    'abcdefghijklmnopqrstuvwxyz0123456789@#$';

            for (let i = 1; i <= 8; i++) {

                var char = Math.floor(Math.random()

                            \* str.length + 1);

                pass += str.charAt(char)

            }

            return pass;

        }

        var password = generateP();

          var transporter = nodemailer.createTransport({

            service: 'gmail',

            auth: {

              user: 'kchhabra416@gmail.com',

              pass: 'rexldpmgpxwgynqr'

            }

          });

          var mailOptions = {

            from: 'kchhabra416@gmail.com',

            to: b,

            subject: 'Login Password',

            html: `Hello ${a}! <br><br>Your Password is ${password}<br><br>Thanks,<br>Student Help`

          };

          transporter.sendMail(mailOptions, function(error, info){

            if (error) {

              console.log(error);

            } else {

              console.log('Email sent: ' + info.response);

            }

          });

          var insertSQL = `INSERT INTO login\_table(username, password, type) VALUES('${c}','${password}','${f}')`;

          var insertSQL2 = `INSERT INTO users(name, username,email, type) VALUES('${a}','${c}','${b}','${f}')`;

            db.query(insertSQL2,function(error){

              if (error) throw error;

            });

            db.query(insertSQL, function (error) {

         if (error) throw error;});

         res.send("ok");

}else{

    res.send("Username Already Exists");

}

})

});

app.post("/login-action",(req,res)=>{

var a = req.body.username;

var b = req.body.password;

var selectquery = `SELECT password,type FROM login\_table WHERE username='${a}'`;

db.query(selectquery,function(error,rows,fields){

if(error)throw error;

if(rows.length<=0){

   res.send("Incorrect Username or Password");

}else{

  if(b==rows[0].password){

    if(rows[0].type=='teacher'){

      res.send("teacher");

   }

    else{

      res.send("student");

    }

  }else{

    res.send("Incorrect Username or Password");

  }

}

});

});

app.get("/", function (req, res) {

    res.redirect("login.html");

});

app.listen(5000);

**CODE OF HTML FILES**

**SIGNUP**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <link rel="stylesheet" href="css/signup.css">

    <title>Sign Up</title>

</head>

<body>

            <div class="box">

                <div class="box1"><img src="images/main.jpg" alt="IMAGE" class="signupimg"></div>

                <div class="box2">

              <form>

                <h1 >SIGN UP</h1>

                <div class="inputfields">

                <div class="group">

                    <input type="text" name="name"  id="name" autocomplete="off" required>

                    <span class="highlight"></span>

                    <span class="bar"></span>

                    <label>Name</label>

                  </div>

                  <div class="group">

                    <input type="text" name="email" id ="email"autocomplete="off" required>

                    <span class="highlight"></span>

                    <span class="bar"></span>

                    <label>Email</label>

                  </div><div class="group">

                    <input type="text" name="username" id="username"autocomplete="off" required>

                    <span class="highlight"></span>

                    <span class="bar"></span>

                    <label>Username</label>

                  </div>

                <select class="drpdwn" name="user" id="user" autocomplete="off" required>

                    <option value="teacher">Teacher</option>

                    <option value="student">Student</option>

                </select>

            </div>

                <br><span id="error" class="errormsg"></span> <br>

                <button class="btndiv" type="button" onclick="UserSignup()">Sign Up</button>

              </form>

                </div>

            </div>

        <script  >

        var UserSignup = () => {

        var name = document.getElementById("name").value;

        var username = document.getElementById("username").value;

        var email = document.getElementById("email").value;

        var user = document.getElementById("user").value;

        var formData = new FormData();

        formData.append("name", name);

        formData.append("email", email);

        formData.append("username", username);

        formData.append("user", user);

        var http = new XMLHttpRequest();

        http.onreadystatechange = function () {

            if (this.readyState == 4 && this.status == 200){

            if(this.response=='ok'){

                document.getElementById("error").innerHTML="Password Sent to Your Email";

                setTimeout(function(){

                    window.location.replace("login.html");

                },4000);

            }else{

                document.getElementById("error").innerHTML=this.response;

             } };

       }

        http.open("POST", "/signup-action", true);

        http.send(formData);

    }

        </script>

</body>

</html>

**LOGIN**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <link rel="stylesheet" href="css/login.css">

    <title>Log In</title>

</head>

<body>

    <div class="box">

        <div class="box1">

            <form>

            <h1>LOG IN</h1>

            <div class="inputfields">

            <div class="group">

                <input type="text"  name="username" id="username" autocomplete="off">

                <span class="highlight"></span>

                <span class="bar"></span>

                <label>Username</label>

              </div>

              <div class="group">

                <input type="password"  name="password" id="password" >

                <span class="highlight"></span>

                <span class="bar"></span>

                <label>Password</label>

              </div>

            </div>

            <span id="msg" class="errormsg"></span><br>

            <div class="btndiv1">

            <button  type="button" onclick="Userlogin()">Log In</button>

        </div>

            <br>

            <div class="btndiv2">

            <button type="submit" ><a href="signup.html">Sign Up</a></button> </div>

        </form>

        </div>

        <div class="box2"><img src="images/main.jpg" alt="IMAGE" class="loginimg"></div>

    </div>

    <script >

      var Userlogin = () => {

        var username = document.getElementById("username").value;

        var password = document.getElementById("password").value;

        var formData = new FormData();

        formData.append("username", username);

        formData.append("password", password);

        var http = new XMLHttpRequest();

        http.onreadystatechange = function () {

            if (this.readyState == 4 && this.status == 200){

            if(this.response=='teacher'){

                window.location.replace("teacherlog.html");

                sessionStorage.setItem("USERN",username);

            }

            else if(this.response=='student'){

                window.location.replace("studentlog.html");

                sessionStorage.setItem("USERN",username);

            }else{

                document.getElementById("msg").innerHTML=this.response;

             } };

       }

        http.open("POST", "/login-action", true);

        http.send(formData);

    }

    </script>

</body>

</html>

**TODO**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>To Do List</title>

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.1.1/css/all.min.css" integrity="sha512-KfkfwYDsLkIlwQp6LFnl8zNdLGxu9YAA1QvwINks4PhcElQSvqcyVLLD9aMhXd13uQjoXtEKNosOWaZqXgel0g==" crossorigin="anonymous" referrerpolicy="no-referrer" />

    <link rel="stylesheet" href="css/todos.css">

</head>

<body onload="getdata()">

    <div class="heading">

        <div class="wtitle">STUDENT HELP</div>

        <div class="backbtn"><button><a href="studentlog.html">BACK</a></button></div>

       </div>

    <div class="container">

        <div id="newtask" class="newtask">

            <div class="group">

                <input type="text" >

                <span class="highlight"></span>

                <span class="bar"></span>

                <label>Add Task</label>

              </div>

            <button id="push" onclick="AddTask()">ADD</button>

        </div>

        <div id="tasks">

        </div>

    </div>

    <script >

        function getdata ()  {

        var http = new XMLHttpRequest();

        var formData = new FormData();

        const username = sessionStorage.getItem('USERN');

        formData.append("username", username);

        http.onreadystatechange = function () {

        if (this.readyState == 4 && this.status == 200){

        var response = JSON.parse(this.response);

        response.forEach(function(value){

            document.querySelector('#tasks').innerHTML

        +=`

        <div class = "task">

        <span  id="taskname">

            ${value.task}

        </span>

        <button class = "delete" onclick="deletetask('${value.id}')">

            <i class="fa-regular fa-trash-can"></i>

        </button>

        </div>

        `;

    })}

    }

        http.open("POST", "/todo-action", true);

        http.send(formData);

    }

    function AddTask(){

        var http = new XMLHttpRequest();

        var formData = new FormData();

        const username = sessionStorage.getItem('USERN');

        formData.append("username", username);

        if(document.querySelector('#newtask input').value.length == 0){

        alert("Please Enter a Task")

        }else{

        var task =    document.querySelector('#newtask input').value;

        formData.append("task", task);

        document.querySelector("#newtask input").value='';

        http.onreadystatechange = function () {

        if (this.readyState == 4 && this.status == 200){

           if(this.response=="ok"){

            document.querySelector("#tasks").innerHTML='';

            getdata();

           }

        }

        }

    }

        http.open("POST", "/addtask", true);

        http.send(formData);

    }

    function deletetask(id){

        var http = new XMLHttpRequest();

        var formData = new FormData();

        formData.append("id", id);

        http.onreadystatechange = function () {

        if (this.readyState == 4 && this.status == 200){

           if(this.response=="ok"){

            document.querySelector("#tasks").innerHTML='';

            getdata();

           }

        }

        }

        http.open("POST", "/deletetask", true);

        http.send(formData);

    }

    </script>

</body>

</html>

**CREATE CLASS**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Create Class</title>

    <link rel="stylesheet" href="css/createclass.css">

</head>

<body>

    <div class="heading">

        <div class="wtitle">STUDENT HELP</div>

        <div class="backbtn"><button><a href="teacherlog.html">BACK</a></button></div>

       </div>

<div class="mainbox">

        <div class="cheading">CREATE CLASS</div>

        <div class="inputbox">

            <div class="inputfields">

                <div class="group">

                    <input type="text" name="cname"   id="cname" autocomplete="off" required>

                    <span class="highlight"></span>

                    <span class="bar"></span>

                    <label>Class Name</label>

                  </div>

                  <div class="group">

                    <input  type="text" name="classsub"  id="classsub" autocomplete="off" required>

                    <span class="highlight"></span>

                    <span class="bar"></span>

                    <label>Subject</label>

                  </div>

                  <div class="group">

                    <input  type="text" name="classcode"   id="classcode" autocomplete="off" required>

                    <span class="highlight"></span>

                    <span class="bar"></span>

                    <label>Class Code</label>

                  </div>

    </div>

    <span id="error" class="error"></span>

        <button class="cbtn" type="button" onclick="classsignup()">Create</button>

    </div>

</div>

    <script>

        function classsignup(){

            var http = new XMLHttpRequest();

            var formData = new FormData();

            const username = sessionStorage.getItem('USERN');

            var cname = document.getElementById("cname").value;

            var subject = document.getElementById("classsub").value;

            var classcode = document.getElementById("classcode").value;

            formData.append("cname", cname);

            formData.append("subject", subject);

            formData.append("username", username);

            formData.append("classcode", classcode);

            var http = new XMLHttpRequest();

        http.onreadystatechange = function () {

            if (this.readyState == 4 && this.status == 200){

                if(this.response=='ok'){

                    alert("Class Created");

                    window.location.replace("teacherlog.html");

                }else{

                    document.getElementById("error").innerHTML=this.response;

                }

            }

        }

        http.open("POST", "/class-signup", true);

        http.send(formData);

        }

    </script>

</body>

</html>

**TEACHER LOGIN**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Teacher Login</title>

    <link rel="stylesheet" href="css/teacherlog.css">

</head>

<body onload="loadclass()">

    <div class="heading">

        <div class="wtitle">STUDENT HELP</div>

        <div class="logbtn"><button><a href="login.html">Logout</a></button></div>

       </div>

            <div class="title"><span class="teachername" id="teachername"></span></div>

        <div class="mainbox">

<div class="featurebox">

    <div class="featureheading">FEATURES</div>

    <div class="cfeatures"><a href="createclass.html"><img src="images/createclass.jpg" alt="create class" class="imagg"></a></div>

    <div class="cfeatures"><a href="todot.html"><img src="images/todo.jpg" alt="todo" class="imagg"></a></div>

</div>

<div class="classbox">

<div class="classheadingg">MY CLASSES</div>

<div id="spawnclasses" class="spawnclass"></div>

</div>

        </div>

    <script>

    function loadclass ()  {

        var http = new XMLHttpRequest();

        var formData = new FormData();

        const username = sessionStorage.getItem('USERN');

        formData.append("username", username);

        var count = 3;

        var class1 = "theclass1";

        var class2 = "theclass2";

        var tname = "Welcome,"

        http.onreadystatechange = function () {

        if (this.readyState == 4 && this.status == 200){

        var response = JSON.parse(this.response);

        response.forEach(function(value){

            if((count%2)!=0){

            document.querySelector('#spawnclasses').innerHTML

        +=`<div class="${class1}" onclick="myclass('${value.id}')" >

         <div class="classtitle"  >

            <div class="classname">${value.cname}</div>

            <div class="classsubject">${value.subject}</div>

         </div>

         <div class="classcode">${value.code}</div>

        </div>`;

        count=count+1;

    }else{

        document.querySelector('#spawnclasses').innerHTML

        +=`<div class="${class2}" onclick="myclass('${value.id}')" >

         <div class="classtitle"  >

            <div class="classname">${value.cname}</div>

            <div class="classsubject">${value.subject}</div>

         </div>

         <div class="classcode">${value.code}</div>

        </div>`;

        count=count+1;

    }

        })

    }

        }

        http.open("POST", "/load-classes", true);

        http.send(formData);

        loadname();

    }

    function loadname(){

        var http = new XMLHttpRequest();

        var formData = new FormData();

        const username = sessionStorage.getItem('USERN');

        formData.append("username", username);

        http.onreadystatechange = function () {

        if (this.readyState == 4 && this.status == 200){

        var response = JSON.parse(this.response);

        response.forEach(function(value){

            document.getElementById("teachername").innerHTML="Welcome,"+value.teacher;

        });

        }

    }

        http.open("POST", "/load-tname", true);

        http.send(formData);

    }

    function myclass(id){

        sessionStorage.setItem("OPENCLASS",id);

        window.location.replace("myclassteacher.html");

    }

    </script>

</body>

</html>

**STUDENT LOGIN**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Student Login</title>

    <link rel="stylesheet" href="css/studentlog.css">

</head>

<body onload="loadclass()">

    <div class="heading">

        <div class="wtitle">STUDENT HELP</div>

        <div class="logbtn"><button><a href="login.html">Logout</a></button></div>

       </div>

            <div class="title"><span class="teachername" id="teachername"></span></div>

        <div class="mainbox">

<div class="featurebox">

    <div class="featureheading">FEATURES</div>

    <div class="cfeatures">

         <div class="joinclass">

            <div class="group">

                <input type="text" id="joincode">

                <span class="highlight"></span>

                <span class="bar"></span>

                <label>Join Code</label>

              </div>

        <button onclick="joincls()">Join class</button>

       <br> <span id="error" class="error"></span>

    </div></div>

    <div class="cfeatures"><a href="todos.html"><img src="images/todo.jpg" alt="todo" class="imagg"></a></div>

</div>

<div class="classbox">

<div class="classheadingg">MY CLASSES</div>

<div id="spawnclasses" class="spawnclass">

</div>

        </div>

        <script>

            function joincls(){

            var http = new XMLHttpRequest();

            var formData = new FormData();

            var code = document.getElementById("joincode").value;

            const username = sessionStorage.getItem('USERN');

            formData.append("code", code);

            formData.append("username", username);

            http.onreadystatechange = function () {

            if (this.readyState == 4 && this.status == 200){

            if(this.response=="Class Joined"){

                alert("Class Joined");

                document.querySelector("#spawnclasses").innerHTML='';

                loadclass()

            }else{

                document.getElementById("error").innerHTML=this.response;

                document.getElementById("joincode").value='';

            }

            }

        }

            http.open("POST", "/join-class", true);

            http.send(formData);

            }

        function loadclass ()  {

            var http = new XMLHttpRequest();

            var formData = new FormData();

            const username = sessionStorage.getItem('USERN');

            formData.append("username", username);

            var count = 3;

            var class1 = "theclass1";

            var class2 = "theclass2";

            http.onreadystatechange = function () {

            if (this.readyState == 4 && this.status == 200){

            var response = JSON.parse(this.response);

            response.forEach(function(value){

                if((count%2)!=0){

                document.querySelector('#spawnclasses').innerHTML

            +=`<div class="${class1}" onclick="myclass('${value.id}')" >

             <div class="classtitle"  >

                <div class="classname">${value.cname}</div>

                <div class="classsubject">${value.subject}</div>

             </div>

             <div class="lowerdiv">

             <div class="classcode">${value.code}</div><div class="teacher">${value.teacher}</div>

            </div>

            </div>`;

            count=count+1;

        }else{

            document.querySelector('#spawnclasses').innerHTML

            +=`<div class="${class2}" onclick="myclass('${value.id}')" >

             <div class="classtitle"  >

                <div class="classname">${value.cname}</div>

                <div class="classsubject">${value.subject}</div>

             </div>

             <div class="lowerdiv">

             <div class="classcode">${value.code}</div><div class="teacher">${value.teacher}</div>

            </div>

            </div>`;

            count=count+1;

        }

            })

        }

            }

            http.open("POST", "/load-classes-stu", true);

            http.send(formData);

            loadname();

        }

        function myclass(id){

            sessionStorage.setItem("OPENCLASS",id);

            window.location.replace("myclassstudent.html");

        }

        function loadname(){

        var http = new XMLHttpRequest();

        var formData = new FormData();

        const username = sessionStorage.getItem('USERN');

        formData.append("username", username);

        http.onreadystatechange = function () {

        if (this.readyState == 4 && this.status == 200){

        var response = JSON.parse(this.response);

        response.forEach(function(value){

            document.getElementById("teachername").innerHTML="Welcome,"+value.name;

        });

        }

    }

        http.open("POST", "/load-sname", true);

        http.send(formData);

    }

        </script>

</body>

</html>

**TEACHER CLASS**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Class</title>

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.1.2/css/all.min.css" integrity="sha512-1sCRPdkRXhBV2PBLUdRb4tMg1w2YPf37qatUFeS7zlBy7jJI8Lf4VHwWfZZfpXtYSLy85pkm9GaYVYMfw5BC1A==" crossorigin="anonymous" referrerpolicy="no-referrer" />

    <link rel="stylesheet" href="css/myclassteacher.css">

</head>

<body onload="loaddata()">

   <div class="heading">

    <div class="wtitle">STUDENT HELP</div>

    <div class="backbtn"><button><a href="teacherlog.html">BACK</a></button></div>

   </div>

   <div class="classhead" id="titleid">

   </div>

   <div class="classwork">

    <div class="classcontent">

        <div class="contenthead">ADD WORK</div>

        <div class="addwork">

        <div class="worktitle">

        <div class="group">

            <input type="text"  id="topic" placeholder="Topic" class="wtopicc">

            <span class="highlight"></span>

            <span class="bar"></span>

            <label>TOPIC</label>

          </div>

            <select class="drpdwn" name="work" id="work" autocomplete="off" required>

                <option value="Assignment">Assignment</option>

                <option value="Notes">Notes</option>

                <option value="Test">Test</option>

            </select>

           <input type="datetime-local" id="time" class="drpdwn"> </div>

           <div class="secondiv">

            <input type="file" id="myfile1" >

            <button type="button" onclick="uploadwork()">ADD WORK</button><span id="error1" class="error1"></span></div>

        </div>

            <div class="spawncontent" id="spawncontent">

            </div>

    </div>

    <div class="queries"><div class="queryhead">QUERIES</div>

    <div class ="querybox">

        <div class="spawnqueries" id="spawnqueriesteacher">

    </div>

    </div>

</div>

    <div class="classstudent">

        <div class="stuhead">STUDENTS</div>

        <div id="spawnstudents" class="spawnstudent">

        </div>

    </div>

   </div>

   <script>

    function loaddata(){

        var http = new XMLHttpRequest();

        var formData = new FormData();

        const classid = sessionStorage.getItem('OPENCLASS');

        formData.append("id", classid);

        http.onreadystatechange = function () {

        if (this.readyState == 4 && this.status == 200){

            var response = JSON.parse(this.response);

            response.forEach(function(value){

            document.querySelector('#titleid').innerHTML +=

            `<div class="classtitle">${value.cname}</div>

             <div class="classtitle">Subject:${value.subject}</div>

             <div class="classtitle">Class Code:${value.code}</div>`;}

             )

        }

    }

        http.open("POST", "/load-data", true);

        http.send(formData);

        loadstudent();

        loadqueries();

    }

    function loadstudent(){

        var http = new XMLHttpRequest();

        var formData = new FormData();

        const classid = sessionStorage.getItem('OPENCLASS');

        formData.append("id", classid);

        count = 1;

        http.onreadystatechange = function () {

        if (this.readyState == 4 && this.status == 200){

            var response = JSON.parse(this.response);

            response.forEach(function(value){

            document.querySelector('#spawnstudents').innerHTML +=

            ` <div class="thestudent">

            <div class="studentno">${count}</div>

            <div class="studentname">${value.name}</div>

         </div>  `;

            count = count+1;

            }

             )

        }

    }

        http.open("POST", "/load-stu", true);

        http.send(formData);

        showcontent();

    }

    function uploadwork(){

        var http = new XMLHttpRequest();

        var formData = new FormData();

        var myfile = document.getElementById("myfile1").files[0];

        var topic = document.getElementById("topic").value;

        var work = document.getElementById("work").value;

        var time = document.getElementById("time").value;

        var classid = sessionStorage.getItem('OPENCLASS');

        formData.append("myfile",myfile);

        formData.append("topic",topic);

        formData.append("work",work);

        formData.append("time",time);

        formData.append("cid",classid);

        http.onreadystatechange = function () {

        if (this.readyState == 4 && this.status == 200){

            if(this.response =="done"){

                alert("Work Uploaded");

                document.querySelector('#spawncontent').innerHTML = '';

                showcontent();

            }else{

                document.getElementById("error1").innerHTML=this.response;

            }

        }

    }

        http.open("POST", "/work-upload", true);

        http.send(formData);

    }

    function showcontent(){

        var http = new XMLHttpRequest();

        var formData = new FormData();

        const classid = sessionStorage.getItem('OPENCLASS');

        formData.append("id", classid);

        http.onreadystatechange = function () {

        if (this.readyState == 4 && this.status == 200){

            var response = JSON.parse(this.response);

            response.forEach(function(value){

            document.querySelector('#spawncontent').innerHTML +=

            `<div class="thecontent">

            <div class="w">

                <i class="fa-solid fa-house-laptop"></i>

              <div class="wtopic">${value.topic}</div>

              <div class="wwork">${value.work}</div>

              <div class="wtime">${value.time}</div>

            </div>

              <div class="wfile">${value.file\_name}</div>

              <div class = "stuuploadwork" id = "${value.id}">

             </div> `;

        })

        }

    }

        http.open("POST", "/show-content", true);

        http.send(formData);

        showstudentwork();

    }

    function showstudentwork(){

        var http = new XMLHttpRequest();

        var formData = new FormData();

        const classid = sessionStorage.getItem('OPENCLASS');

        formData.append("id", classid);

        http.onreadystatechange = function () {

        if (this.readyState == 4 && this.status == 200){

            var response = JSON.parse(this.response);

            response.forEach(function(value){

                document.getElementById(value.wid).innerHTML+=

`        <div class = "showstu">

    <div class="studdname">

               <span>${value.name}</span></div>

               <div class="workdwn">

               <form method = "get" action="/stu-download-work" class = "download" >

                    <input type = "hidden" name = "nothing" value = "${value.filename}" />

              <button type="submit"  class="dicon">

                <i class="fa-solid fa-download"></i>

              </button>

              </form>

            </div>

              </div>`;

            })

        }

    }

        http.open("POST", "/download-stu-work", true);

        http.send(formData);

    }

    function loadqueries(){

        var http = new XMLHttpRequest();

        var formData = new FormData();

        const classid = sessionStorage.getItem('OPENCLASS');

        formData.append("id", classid);

        http.onreadystatechange = function () {

        if (this.readyState == 4 && this.status == 200){

            var response = JSON.parse(this.response);

            response.forEach(function(value){

                iid = "q"+value.qid;

                rid = "r"+value.qid;

                document.getElementById("spawnqueriesteacher").innerHTML+=`

                <div class="thequery" id="thequery"><div class="queryheadstudent"><div class="studentnamequery">${value.name}</div>

            <div class="querybystudent">${value.query}</div></div>

                <div class="queryinput" id="${rid}">

    <input type="text" placeholder="Reply Query" id="${iid}">

    <button type="submit" class = "sendanswer" onclick="sendanswer('${value.qid}','${iid}')" >

        <i class="fa-solid fa-reply"></i>

              </button>

</div>

            </div>`});

        }

    }

        http.open("POST", "/show-teacher-queries", true);

        http.send(formData);

        loadrepliedquery();

    }

    function sendanswer(qid,reply){

        var http = new XMLHttpRequest();

        var formData = new FormData();

        var reply = document.getElementById(reply).value;

        formData.append("qid", qid);

        formData.append("reply", reply);

        http.onreadystatechange = function () {

        if (this.readyState == 4 && this.status == 200){

          if(this.response=="done"){

            loadrepliedquery()}

        }

    }

        http.open("POST", "/reply-query", true);

        http.send(formData);

    }

    function loadrepliedquery(){

        var http = new XMLHttpRequest();

        var formData = new FormData();

        http.onreadystatechange = function () {

        if (this.readyState == 4 && this.status == 200){

            var response = JSON.parse(this.response);

            response.forEach(function(value){

                var rid = "r"+value.qid;

                document.getElementById(rid).innerHTML=`

                <i class="fa-solid fa-arrow-right"></i>

                <div class="repq">${value.reply}</div>

                `

            });

        }

    }

       http.open("POST", "/replied-query", true);

       http.send(formData);

    }

   </script>

</body>

</html>

**STUDENT CLASS**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Class</title>

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.1.1/css/all.min.css" integrity="sha512-KfkfwYDsLkIlwQp6LFnl8zNdLGxu9YAA1QvwINks4PhcElQSvqcyVLLD9aMhXd13uQjoXtEKNosOWaZqXgel0g==" crossorigin="anonymous" referrerpolicy="no-referrer" />

    <link rel="stylesheet" href="css/myclassstudent.css">

</head>

<body onload="loaddata()">

   <div class="heading">

    <div class="wtitle">STUDENT HELP</div>

    <div class="backbtn"><button><a href="studentlog.html">Back</a></button></div>

   </div>

   <div class="classhead" id="titleid">

   </div>

   <div class="classwork">

    <div class="classcontent">

        <div class="contenthead">WORK</div>

            <div class="spawncontent" id="spawncontent">

            </div>

    </div>

    <div class="queries"><div class="queryhead">QUERIES</div>

    <div class ="querybox">

        <div class="spawnqueries" id="spawnqueries">

    </div>

    </div>

<div class="queryinput">

    <input type="text" placeholder="Any Queries?" id="askquery" class="askquery">

    <button type="submit" class = "sendquery" onclick="sendquery()" >

        <i class="fa-solid fa-reply"></i>

      </button>

</div>

</div>

    <div class="classstudent">

        <div class="stuhead">STUDENTS</div>

        <div id="spawnstudents" class="spawnstudent">

        </div>

    </div>

   </div>

   <script>

    function loaddata(){

        var http = new XMLHttpRequest();

        var formData = new FormData();

        const classid = sessionStorage.getItem('OPENCLASS');

        formData.append("id", classid);

        http.onreadystatechange = function () {

        if (this.readyState == 4 && this.status == 200){

            var response = JSON.parse(this.response);

            response.forEach(function(value){

            document.querySelector('#titleid').innerHTML =

            `<div class="classtitle">${value.cname}</div>

             <div class="classtitle">Subject:${value.subject}</div>

             <div class="classtitle">Class Code:${value.code}</div>

             <div class="classtitle">Teacher:${value.teacher}</div>`;}

             )

        }

    }

        http.open("POST", "/load-data", true);

        http.send(formData);

        loadstudent()

    }

    function loadstudent(){

        var http = new XMLHttpRequest();

        var formData = new FormData();

        const classid = sessionStorage.getItem('OPENCLASS');

        formData.append("id", classid);

        count = 1;

        http.onreadystatechange = function () {

        if (this.readyState == 4 && this.status == 200){

            var response = JSON.parse(this.response);

            response.forEach(function(value){

            document.querySelector('#spawnstudents').innerHTML +=

            ` <div class="thestudent">

            <div class="studentno">${count}</div>

            <div class="studentname">${value.name}</div>

         </div>  `;

            count = count+1;

            }

             )

        }

    }

        http.open("POST", "/load-stu", true);

        http.send(formData);

    loadcontent();

    showqueries();

    }

    function loadcontent(){

        var http = new XMLHttpRequest();

        var formData = new FormData();

        const classid = sessionStorage.getItem('OPENCLASS');

        formData.append("id", classid);

        http.onreadystatechange = function () {

        if (this.readyState == 4 && this.status == 200){

            var response = JSON.parse(this.response);

            response.forEach(function(value){

            if(value.work!="Notes"){

            document.querySelector('#spawncontent').innerHTML +=

            `<div class="thecontent">

            <div class="w">

              <div class="wtopic">${value.topic}</div>

              <div class="wwork">${value.work}</div>

              <div class="wtime">${value.time}</div>

            </div>

            <div class="wf">

              <div class="wfile">${value.file\_name}</div>

              <div class="dbtn">

                <form method = "get" action="/stu-download">

                    <input type = "hidden" name = "nothing" value = "${value.file\_name}" />

              <button type="submit" class = "download" >

                <i class="fa-solid fa-download"></i></i>

              </button>

              </form>

              </div>

              </div>

              <div id="${value.id}" class="wupload">

              <div class="wffupload">

                <span>Upload Work</span>

                <input type = "file" id="mystufile">

                <button type= "button" onclick="stuupload('${value.id}','${value.topic}')"><i class="fa-solid fa-arrow-up-from-bracket"></i></button>

                <span id='error2' class="error2"></span>

            </div>

            </div>

             </div> `;

            }else{

                document.querySelector('#spawncontent').innerHTML +=

            `<div class="thecontent">

            <div class="w">

              <div class="wtopic" id="wtopic">${value.topic}</div>

              <div class="wwork">${value.work}</div>

            </div>

            <div class="wff">

              <div class="wfile">${value.file\_name}</div>

              <div class="dbtn">

                <form method = "get" action="/stu-download">

                    <input type = "hidden" name = "nothing" value = "${value.file\_name}" />

              <button type="submit" class = "download" >

                <i class="fa-solid fa-download"></i>

              </button>

              </form>

              </div>

              </div>

             </div> `;

            }

        })

        }

    }

        http.open("POST", "/load-content", true);

        http.send(formData);

        loadworksubmit();

    }

    function download(fname){

        var http = new XMLHttpRequest();

        var formData = new FormData();

        formData.append("fname", fname);

        http.onreadystatechange = function () {

        if (this.readyState == 4 && this.status == 200){

            window.location = \_\_dirname + "/upload/" + fname;

        }

    }

        http.open("POST", "/stu-download", true);

        http.send(formData);

    }

    function stuupload(wid,topic){

        var http = new XMLHttpRequest();

        var formData = new FormData();

        var classid = sessionStorage.getItem('OPENCLASS');

        var username = sessionStorage.getItem('USERN');

        var myfile = document.getElementById("mystufile").files[0];

        formData.append("myfile",myfile);

        formData.append("wid",wid);

        formData.append("cid", classid);

        formData.append("username", username);

        formData.append("topic", topic);

        http.onreadystatechange = function () {

        if (this.readyState == 4 && this.status == 200){

            if(this.response =="done"){

                alert("Work Uploaded");

            }else{

                    document.getElementById("error2").innerHTML=this.response;

                }

        }

    }

        http.open("POST", "/stu-upload", true);

        http.send(formData);

        loadworksubmit()

    }

    function loadworksubmit(){

        var http = new XMLHttpRequest();

        var formData = new FormData();

        var classid = sessionStorage.getItem('OPENCLASS');

        var username = sessionStorage.getItem('USERN');

        formData.append("cid", classid);

        formData.append("username", username);

        http.onreadystatechange = function () {

        if (this.readyState == 4 && this.status == 200){

            var response = JSON.parse(this.response);

            response.forEach(function(value){

            document.getElementById(value.wid).innerHTML=

                `<div class="wfupload">

                <span>Work Submited</span>     </div>`

            });

        }

    }

        http.open("POST", "/work-submit", true);

        http.send(formData);

    }

    function sendquery(){

        var http = new XMLHttpRequest();

        var formData = new FormData();

        var classid = sessionStorage.getItem('OPENCLASS');

        var username = sessionStorage.getItem('USERN');

        var query = document.getElementById("askquery").value;

        formData.append("cid", classid);

        formData.append("username", username);

        formData.append("query", query);

        http.onreadystatechange = function () {

        if (this.readyState == 4 && this.status == 200){

            if(this.response =="done"){

                document.getElementById("askquery").innerHTML="";

                document.getElementById("spawnqueries").innerHTML= '' ;

                showqueries();

            }

        }

    }

        http.open("POST", "/ask-query", true);

        http.send(formData);

    }

    function showqueries(){

        var http = new XMLHttpRequest();

        var formData = new FormData();

        var classid = sessionStorage.getItem('OPENCLASS');

        var username = sessionStorage.getItem('USERN');

        formData.append("cid", classid);

        formData.append("username", username);

        http.onreadystatechange = function () {

        if (this.readyState == 4 && this.status == 200){

            var response = JSON.parse(this.response);

            response.forEach(function(value){

            var rid="r"+value.qid;

            document.getElementById("spawnqueries").innerHTML+=

           ` <div class="thequery" id="thequery"><div class="thequery1">

            ${value.query}</div>

            </div>

            <div class="queryreply" id="${rid}"></div>

            `

            });

        }

    }

        http.open("POST", "/show-query", true);

        http.send(formData);

        showrepliedqueries();

    }

    function showrepliedqueries(){

        var http = new XMLHttpRequest();

        var formData = new FormData();

        http.onreadystatechange = function () {

        if (this.readyState == 4 && this.status == 200){

            var response = JSON.parse(this.response);

            response.forEach(function(value){

                var rid = "r"+value.qid;

                document.getElementById(rid).innerHTML=`

                <i class="fa-solid fa-arrow-right"></i>

                <div class="qreply1">${value.reply}</div>

                `

            });

        }

    }

       http.open("POST", "/replied-query", true);

       http.send(formData);

    }

   </script>

</body>

</html>

**BIBLIOGRAPHY**

* Web Study Notes provided by VMM EDUCATION
* JavaScript by Stephen Blumenthal
* Head first HTML and CSS by Elisabeth Robson and Eric Freeman
* Mastering Node.js by Sandro Pasquali
* <https://www.w3schools.com>
* <https://tutorialspoint.com>
* <https://youtube.com>