IOSR Journal Of Humanities And Social Science (IOSR-JHSS)
Volume 30, Issue 3, Series 4 (March, 2025) 50-74
e-ISSN: 2279-0837, p-ISSN: 2279-0845.
www.iosrjournals.org

Optimizing Teaching and Learning: A Web-Based Grammar Correction Tool for Multifaceted Classrooms

Md. Tariqur Rahman[1], Adnan Shakur[2]
[1] (Undergraduate Researcher, Department of English, University of Global Village (UGV), Barishal)
[2] (Lecturer and Deputy Head, Department of English, University of Global Village (UGV), Barishal)

Abstract
This study presents a web-based grammar correction tool designed to support language education in large and diverse classrooms. Using advanced natural language processing (NLP) techniques, including LanguageTool and TextBlob, the tool provides real-time grammar analysis and correction. It is equipped with a robust API that enables automated feedback and seamless integration into classroom workflows.
Designed for multifaceted large classrooms, this tool stands out by allowing both teachers and students to access grammatical error reports, facilitating targeted teaching and personalized learning. Teachers can automate task management, analyze common student errors, and tailor their instruction based on real-time data. Meanwhile, students receive instant feedback, enabling them to identify their mistakes, track their progress, and enhance their grammar skills. Unlike traditional approaches, this tool fosters engagement by making grammar correction more interactive and accessible.
Preliminary results indicate a significant improvement in grammatical accuracy and student participation, with positive feedback from both educators and learners. This study underscores the potential of technology to transform language learning by addressing key challenges in grammar instruction and optimizing the teaching-learning experience.
Keywords: *Natural Language Processing (NLP), Multifaceted Classroom, Educational Technology (EdTech), Grammar Correction Tool, Automated Feedback, Digital Language Learning, Technology Integrated Classroom*

Date of Submission: 07-03-2025 Date of Acceptance: 17-03-2025

I. Introduction

Grammar instruction in language education encounters various challenges, particularly in large classrooms where students exhibit a wide range of proficiency levels and teacher-student interactions are limited. These obstacles are especially evident in Bangladesh, where numerous universities, including the University Global Village (UGV), face high student-to-teacher ratios alongside differing levels of language competence. Consequently, educators often struggle to deliver timely and personalized feedback, leaving many learners with persistent grammatical issues that continue throughout their academic experience. Traditional approaches to grammar instruction are further complicated by the difficulties in addressing individual learning styles and proficiency levels within diverse classroom settings, as well as by time constraints, curriculum requirements, and a lack of capacity for individualized attention.

In this context, teachers frequently encounter challenges in assessing student submissions effectively, due to the size of their classes, which leads to delays in providing targeted feedback. As a result, students often remain unaware of and unable to rectify grammatical errors in their assignments—an ongoing issue that contributes to the continuation of such mistakes over time. The growing adoption of digital tools in contemporary education offers a significant opportunity to surmount these difficulties through innovative solutions that can enhance both teaching and learning experiences.

This research endeavors to create and implement a web-based grammar correction tool specifically designed to meet the needs of diverse classrooms. By taking advantage of digital platforms and the widespread availability of the internet, this tool has the potential to transform grammar education. It aims to deliver immediate feedback, create personalized learning trajectories, and boost student engagement, all of which are essential to overcoming the shortcomings of conventional grammar instruction. Moreover, the tool is intended to accommodate the diverse linguistic backgrounds, learning styles, and technological skills characteristic of today's classrooms, ensuring that it is adaptable and inclusive.

Within the Bangladeshi educational context, where classrooms often reflect a wide range of student requirements and substantial class sizes, the adoption of such technology can help ease the burden on teachers while providing effective solutions to the challenges encountered in grammar instruction. Under the guidance of my undergraduate research supervisor, Adnan Shakur, I have led this project to develop a web-based tool that addresses the shortcomings of traditional teaching methods, promotes self-directed learning, and enhances students' grammatical proficiency. My supervisor's insights have been instrumental in shaping the project's direction, particularly in identifying the pedagogical gaps in existing grammar instruction and integrating technology-driven solutions to enhance classroom learning. His expertise has helped refine the tool's functionality to ensure it provides meaningful feedback and aligns with the needs of students and educators within the Bangladeshi educational context. The tool aims to improve the overall teaching and learning experience by automating the error correction process and supplying constructive feedback, thereby offering a sustainable solution for language education within the Bangladeshi framework. While it is still in the development phase, initial progress has demonstrated its potential to address key challenges in grammar instruction, and further refinements are being made to enhance its accuracy and usability.

II. Literature Review

In recent years, the integration of technology into educational settings has gained significant attention as educators seek innovative ways to enhance teaching and learning experiences. One area of particular interest is the utilization of web-based grammar correction tools to support language acquisition and proficiency development in multifaceted classrooms. This literature review explores existing research related to the implementation and impact of such tools on teaching and learning outcomes.

Theoretical Framework: Technology in Education

The integration of technology in education has been a subject of extensive research. Scholars like Mishra and Koehler (2006) introduced the Technological Pedagogical Content Knowledge (TPACK) framework, which emphasizes the importance of integrating technology seamlessly into teaching practice while considering both pedagogical and content knowledge. This framework provides a theoretical lens for examining how web-based grammar correction tools can be effectively incorporated into language instruction**.**

Challenges in Grammar Instruction

Research highlights the persistent challenges students face in grammar and writing proficiency. Alfaki (2015) conducted a study at Nile Valley University, North Sudan, examining students' compositions describing their home villages and towns. The findings revealed that learners struggle with tenses, spelling, grammar, and cognitive organization**,** making writing a particularly difficult skill to master.

Biswas (2021) investigated the writing challenges faced by tertiary-level students at North South University, Bangladesh, particularly those transitioning from Bengali-medium to English-medium instruction. The study identified difficulties in both mechanical (e.g., punctuation, spelling, grammar) and rhetorical (e.g., idea generation, argumentation, audience awareness) aspects of writing. the study underscores the need for structured teacher feedback and collaborative teaching approaches to address students' difficulties and facilitate their adaptation to English-medium education**.**

Both studies emphasize the need for structured grammar instruction and targeted feedback to help students develop greater accuracy and confidence in academic writing.

Additionally, Teaching grammar in multifaceted classrooms presents unique challenges due to the diverse linguistic and cultural backgrounds of students. Educators often encounter difficulties in providing differentiated instruction that caters to varying proficiency levels, as well as in engaging students who may find grammar abstract or uninteresting.

Mamadiyorova and Rahmonqulova (2024) highlight the complexities educators face in teaching grammar in linguistically diverse classrooms. Their study identifies challenges such as the abstract nature of grammar rules, lack of student engagement, and varying proficiency levels. They emphasize the need for context-based teaching and the integration of interactive tools to enhance grammar instruction effectiveness.

Similarly, Nurmatova (2025) explores the difficulties educators face when teaching grammar to non-native speakers, particularly in managing linguistic diversity and differing levels of prior grammatical knowledge among students. The study suggests practical strategies, such as the integration of technology, communicative approaches, and contextualized instruction, to improve grammar teaching in diverse classrooms.

In addressing the persistent challenges of grammar instruction, technology emerges as a vital tool for enhancing student engagement and improving learning outcomes. Interactive tools, AI-driven feedback systems, and context-based digital platforms can help bridge proficiency gaps, making grammar instruction more accessible and effective. By integrating technology with structured teaching approaches, educators can create dynamic, student-centered learning environments that foster accuracy and confidence in academic writing.

Digital Tools for Grammar Improvement

Existing research underscores the impact of digital grammar correction tools on students' writing abilities. Studies on Grammarly indicate that its use leads to substantial improvements, with mean scores increasing from 76.10 to 80.68 over two cycles of action research **(**Maulidina & Wibowo, 2022**).** Similarly, the integration of AI-driven tools such as ChatGPT and DeepL Write has shown notable enhancements in grammar and sentence structure, with 68.8% of participants reporting improvements **(**Obari, 2024**).** These findings suggest that digital tools can act as effective writing assistants, helping learners refine their grammatical accuracy**.**

A study by Cavaleri and Dianati (2016) titled *"You Want Me to Check Your Grammar Again? The Usefulness of an Online Grammar Checker as Perceived by Students"* found that the majority of learners in the study required proofreading services and lacked confidence in their writing. The research suggests that grammar checker tools help students become more self-reliant and confident in their writing.

Teacher Feedback Mechanisms

Despite the advancements in AI-assisted correction, teacher feedback remains a critical component of effective grammar instruction**.** Research shows that students' grammatical accuracy improves significantly when feedback is implemented systematically. For instance, in a study by Solfiyatuzzahro, Santihastuti, and Erfan (2019) students' accuracy increased from 72% to 78% after two cycles of targeted feedback application. This highlights the importance of personalized, instructor-led guidance, which AI tools alone may not fully replace.

Engaging Digital Contexts

Beyond conventional grammar correction platforms, researchers have explored the role of social media and digital engagement in language learning. Utilizing platforms like TikTok and Twitter for grammar instruction has been found to increase student engagement and linguistic awareness (Crovitz et al., 2022). These interactive environments encourage students to apply grammatical concepts in real-world contexts, fostering meta-awareness of language use beyond the classroom.

Research Gap

While digital grammar correction tools, AI-based writing assistants, and teacher-led feedback have demonstrated their effectiveness in improving students' writing skills, a significant gap remains in their integration within interactive classroom settings**.** Most existing grammar correction applications provide individualized feedback but fail to inform instructors about students' common weaknesses or areas requiring additional support. As a result, teachers lack comprehensive error reports that could help them tailor their instruction more effectively.

Previous studies have not extensively explored how real-time feedback and collective grammar analysis can be incorporated into classroom instruction. The ability to display frequent grammatical errors on digital projectors and facilitate live discussions on writing challenges could significantly enhance students' understanding of grammar in context.

Proposed Solution

Our tool seeks to address this gap by enabling teachers to obtain detailed error reports, identify patterns in students' mistakes, and provide real-time feedback during lessons. By bridging the gap between automated error detection and interactive classroom instruction, this tool has the potential to enhance both individual student outcomes and overall classroom dynamics**.**

III. Methodology

System Architecture:

The grammar correction tool was developed using a combination of technologies to ensure a user-friendly interface, efficient processing, and accurate feedback. The system follows a client-server architecture, where teachers create tasks, students submit responses, and the tool provides real-time grammar analysis and corrective feedback.

1. **Frontend Development:** The interface was designed using HTML, CSS, JavaScript, and Bootstrap to ensure responsiveness and ease of use.
2. **Backend Processing:** PHP and Python were employed for server-side scripting, with Python's Flask **framework** playing a crucial role in API development.
3. **Database Management:** MySQL was used to store and manage data efficiently, handling tasks, user roles, and responses.
4. **Natural Language Processing (NLP) Integration:** The grammar correction module incorporated the following tools:

o **LanguageTool** for rule-based grammar checking.
o **TextBlob** for contextual analysis.
o **SpellChecker** for real-time spelling correction.

By integrating these technologies, the tool was designed to provide real-time, automated grammar feedback to students while reducing the administrative burden on educators.

Implementation and Testing

The tool was piloted in classrooms at the University of Global Village, Barishal, Bangladesh, where it was tested on undergraduate students in the Department of English enrolled in English Composition (0231-1102) course. Given the challenges of large classrooms and varying English proficiency levels, this institution provided a suitable environment for evaluating the tool's effectiveness.

The implementation process involved the following steps:

1. **Integration into Coursework:** The tool was introduced as a supplementary grammar correction system for students' writing tasks.
2. **Training and Orientation:** A brief training session was conducted to familiarize students with the tool's functionality.
3. **Feedback Collection:** Students used the tool for multiple assignments, after which feedback was gathered through a structured survey.

User Evaluation & Data Collection

To assess the effectiveness of the grammar correction tool, a survey questionnaire was administered to 50 undergraduate students. The survey included:

- **Likert-scale questions** to measure aspects such as effectiveness, engagement, ease of use, and time efficiency.
- **Open-ended questions** to capture qualitative insights on usability, areas for improvement, and the tool's impact on confidence in grammar usage.

Data Analysis

Survey responses were analyzed using descriptive statistics (mean ratings and response distributions) to identify trends in student satisfaction. Additionally, qualitative responses were categorized thematically to highlight key insights regarding usability and areas for improvement.

The mean rating for each Likert-scale question was calculated using the formula:

$$\bar{X} = \frac{\sum_{i=1}^{n} f_i x_i}{N}$$

where:

- x_i = rating score (1 to 5),
- f_i = frequency of responses for each rating,
- N = total number of responses.

Ethical Considerations

All participants were informed about the research objectives, and participation was voluntary. The collected data was anonymized and used solely for research and tool refinement purposes.

IV. Detailed Structure of the Tool

Teacher Interface:

Home(index.html):

The home page includes two buttons:

- **Teacher**
- **Student**

Users can select the respective role by clicking on the relevant button.

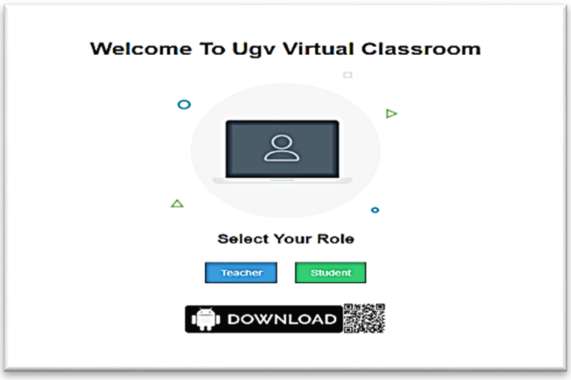


Figure 1: Home page

Teacher Login/Signup (teacher.html):
Once the user selects the **Teacher** button, they are directed to the teacher.html page where two options are available:

- **Login**
- **Signup**

Teachers must create an account by selecting **Signup** or **Login** with their existing credentials by choosing Login.

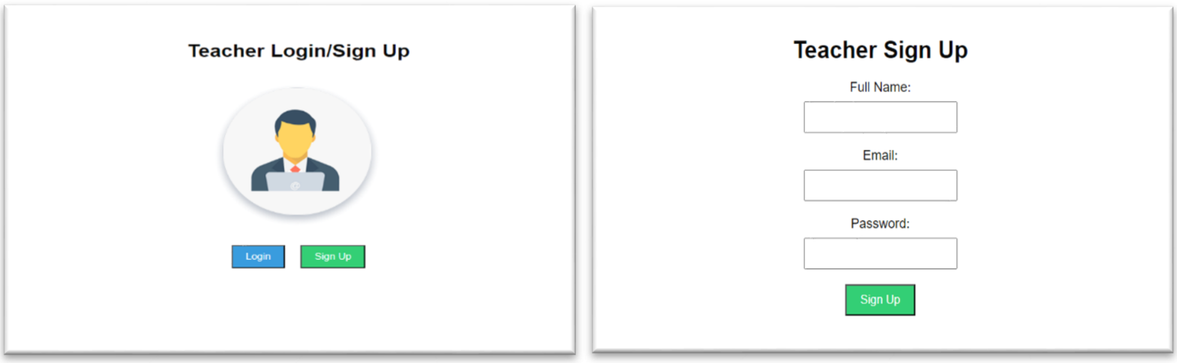
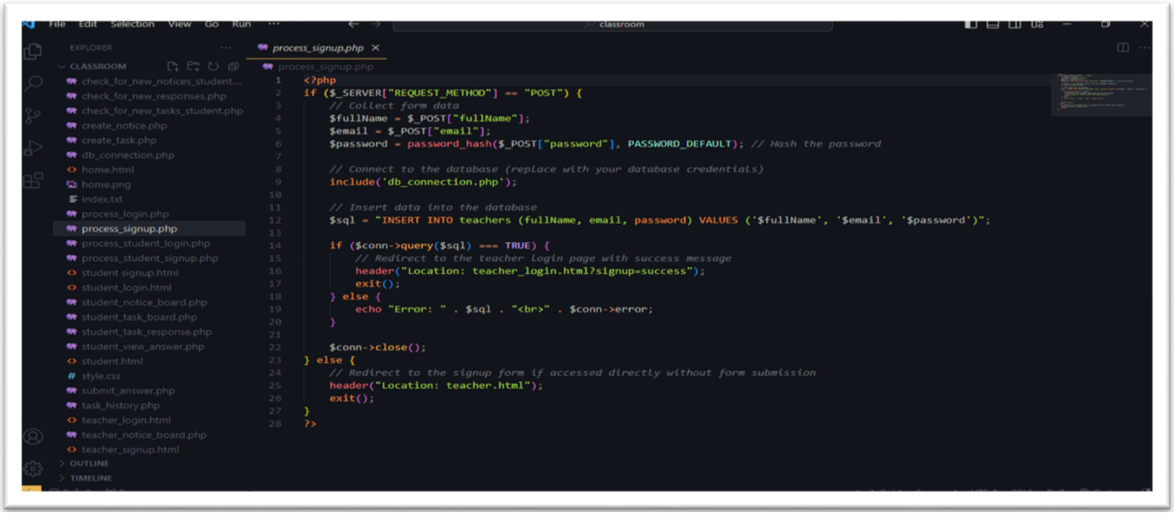


Figure 2: Frontend:teacher.html

Teacher Sign Up (teacher_signup.html & process_signup.php):
Upon successful registration, the system redirects the user to the login page (teacher_login.html). The registration details are stored in the **teachers** table in the database through the process_signup.php script.



```php
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // Collect form data
    $fullName = $_POST["fullName"];
    $email = $_POST["email"];
    $password = password_hash($_POST["password"], PASSWORD_DEFAULT); // Hash the password

    // Connect to the database (replace with your database credentials)
    include('db_connection.php');

    // Insert data into the database
    $sql = "INSERT INTO teachers (fullName, email, password) VALUES ('$fullName', '$email', '$password')";

    if ($conn->query($sql) === TRUE) {
        // Redirect to the teacher login page with success message
        header("Location: teacher_login.html?signup=success");
        exit();
    } else {
        echo "Error: " . $sql . "<br>" . $conn->error;
    }

    $conn->close();
} else {
    // Redirect to the signup form if accessed directly without form submission
    header("Location: teacher.html");
    exit();
}
?>
```

Figure 3: Backend: process_signup.php

Teacher Login (teacher_login.html & process_login.php): Teachers can log in with their email and password. The process_login.php script verifies the login credentials against the teachers table. If the credentials match, the user is redirected to their portal (welcome_teacher.php), otherwise, an error message is shown.

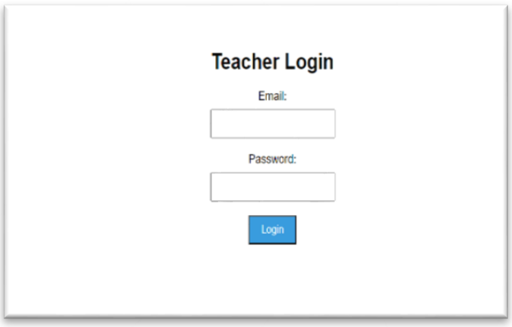
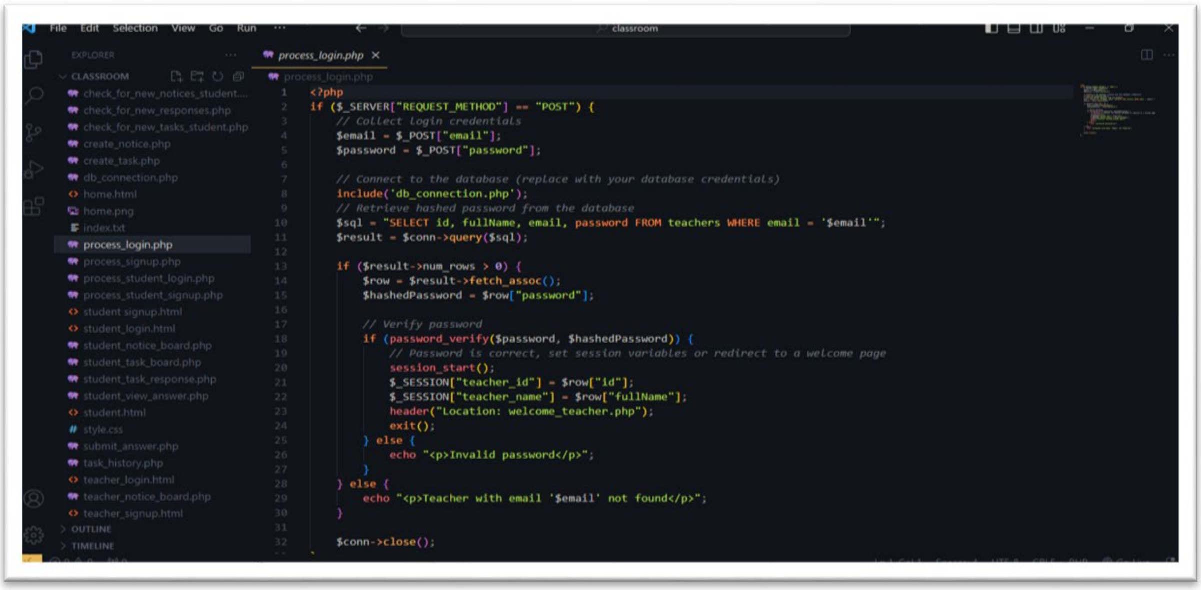


Figure 4: Frontend: teacher_login.html



```php
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // Collect login credentials
    $email = $_POST["email"];
    $password = $_POST["password"];

    // Connect to the database (replace with your database credentials)
    include('db_connection.php');
    // Retrieve hashed password from the database
    $sql = "SELECT id, fullName, email, password FROM teachers WHERE email = '$email'";
    $result = $conn->query($sql);

    if ($result->num_rows > 0) {
        $row = $result->fetch_assoc();
        $hashedPassword = $row["password"];

        // Verify password
        if (password_verify($password, $hashedPassword)) {
            // Password is correct, set session variables or redirect to a welcome page
            session_start();
            $_SESSION["teacher_id"] = $row["id"];
            $_SESSION["teacher_name"] = $row["fullName"];
            header("Location: welcome_teacher.php");
            exit();
        } else {
            echo "<p>Invalid password</p>";
        }
    } else {
        echo "<p>Teacher with email '$email' not found</p>";
    }

    $conn->close();
```

Figure 5: Backend: process_login.php

Welcome Teacher (welcome_teacher.php & check_for_new_responses.php):
Once logged in, teachers have access to the following options:

- **Create Task**
- **Task Response**
- **Notice**

Notifications regarding new responses are displayed using the check_for_new_responses.php script, which queries the **teacher_notifications** table.

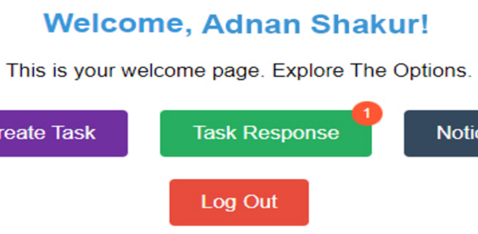
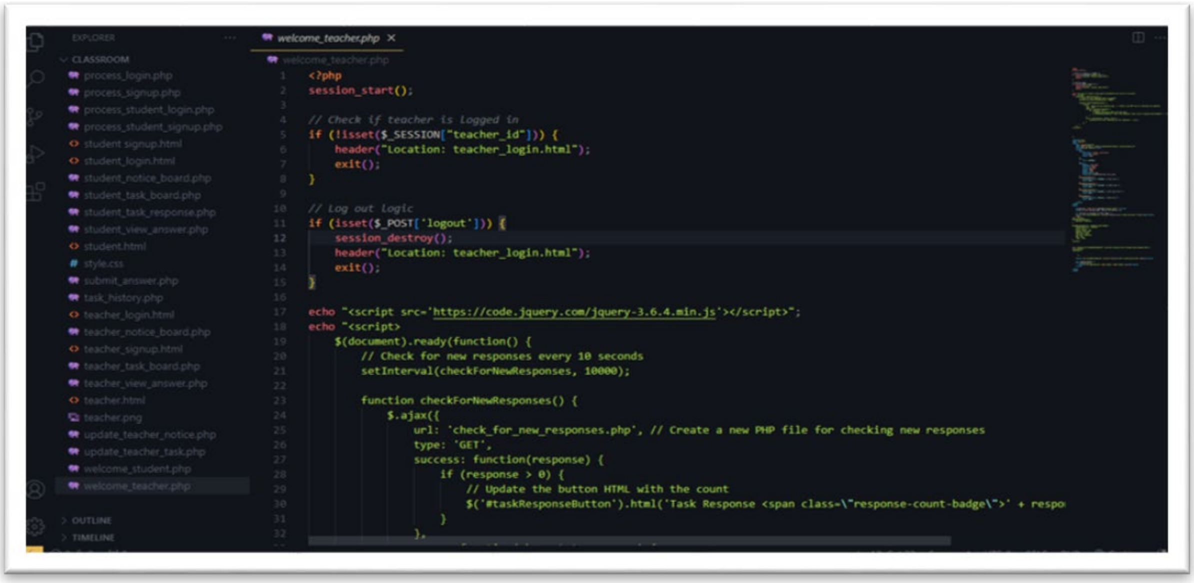
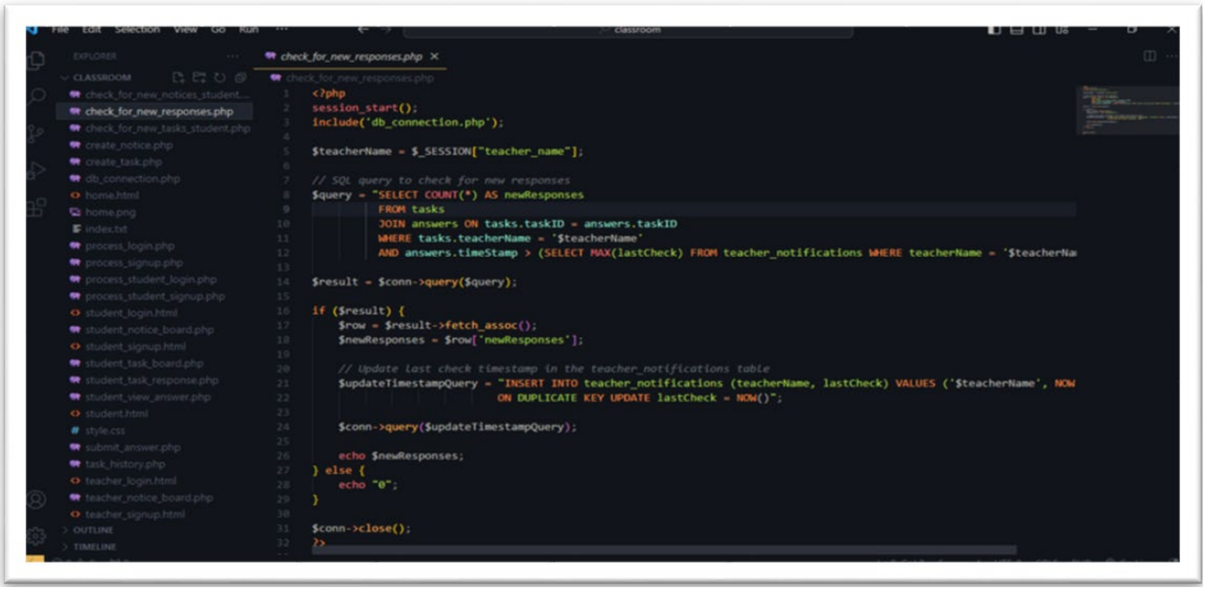


Figure 6: frontend: welcome_teacher.php



```
<?php
session_start();

// Check if teacher is logged in
if (!isset($_SESSION["teacher_id"])) {
    header("Location: teacher_login.html");
    exit();
}

// Log out logic
if (isset($_POST['logout'])) {
    session_destroy();
    header("Location: teacher_login.html");
    exit();
}

echo "<script src='https://code.jquery.com/jquery-3.6.4.min.js'></script>";
echo "<script>
    $(document).ready(function() {
        // Check for new responses every 10 seconds
        setInterval(checkForNewResponses, 10000);

        function checkForNewResponses() {
            $.ajax({
                url: 'check_for_new_responses.php', // Create a new PHP file for checking new responses
                type: 'GET',
                success: function(response) {
                    if (response > 0) {
                        // Update the button HTML with the count
                        $('#taskResponseButton').html('Task Response <span class=\"response-count-badge\">' + respo
                    }
                },
```

Figure 7: Backend: welcome_teacher.php



```
<?php
session_start();
include('db_connection.php');

$teacherName = $_SESSION["teacher_name"];

// SQL query to check for new responses
$query = "SELECT COUNT(*) AS newResponses
          FROM tasks
          JOIN answers ON tasks.taskID = answers.taskID
          WHERE tasks.teacherName = '$teacherName'
          AND answers.timeStamp > (SELECT MAX(lastCheck) FROM teacher_notifications WHERE teacherName = '$teacherNa

$result = $conn->query($query);

if ($result) {
    $row = $result->fetch_assoc();
    $newResponses = $row['newResponses'];

    // Update last check timestamp in the teacher_notifications table
    $updateTimestampQuery = "INSERT INTO teacher_notifications (teacherName, lastCheck) VALUES ('$teacherName', NOW
                            ON DUPLICATE KEY UPDATE lastCheck = NOW()";

    $conn->query($updateTimestampQuery);

    echo $newResponses;
} else {
    echo "0";
}

$conn->close();
?>
```

Figure 8: Backend: check_for_new_responses.php

Create Task (create_task.php):

Teachers can create tasks by selecting a semester from a dropdown menu and entering the task details. Upon submission, the task is stored in the **tasks** table, and a success message is displayed. Additionally, the interface provides three buttons:

- **Task Board** – To view assigned tasks
- **Go Portal** – To navigate back to the main portal
- **Logout** – To exit the system

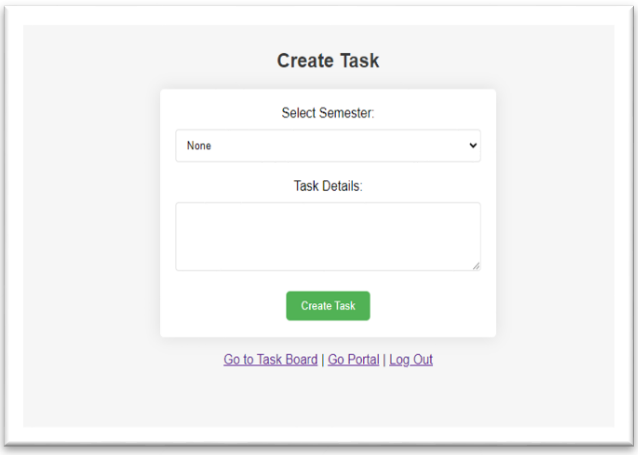
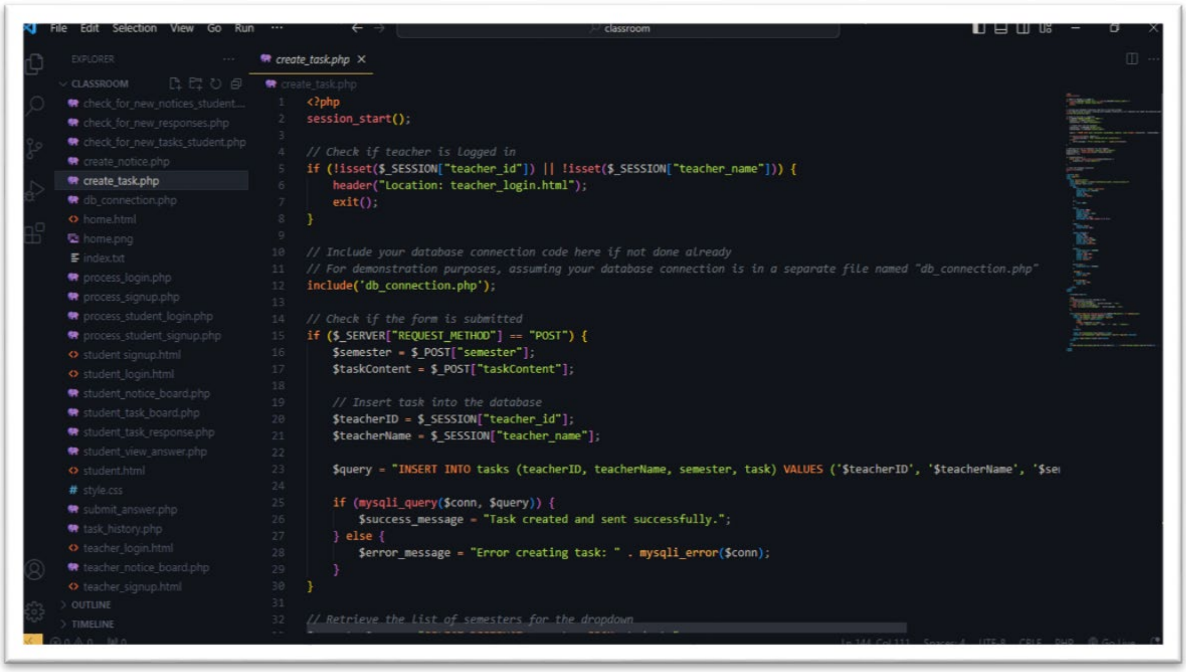


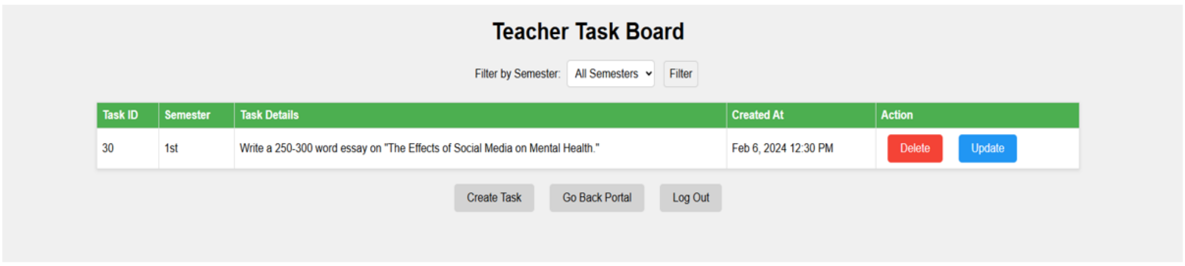
Figure 9: frontend: create_task.php



```php
<?php
session_start();

// Check if teacher is logged in
if (!isset($_SESSION["teacher_id"]) || !isset($_SESSION["teacher_name"])) {
    header("Location: teacher_login.html");
    exit();
}

// Include your database connection code here if not done already
// For demonstration purposes, assuming your database connection is in a separate file named "db_connection.php"
include('db_connection.php');

// Check if the form is submitted
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $semester = $_POST["semester"];
    $taskContent = $_POST["taskContent"];

    // Insert task into the database
    $teacherID = $_SESSION["teacher_id"];
    $teacherName = $_SESSION["teacher_name"];

    $query = "INSERT INTO tasks (teacherID, teacherName, semester, task) VALUES ('$teacherID', '$teacherName', '$se

    if (mysqli_query($conn, $query)) {
        $success_message = "Task created and sent successfully.";
    } else {
        $error_message = "Error creating task: " . mysqli_error($conn);
    }
}

// Retrieve the list of semesters for the dropdown
```

Figure 10: Backend: create_task.php

Teacher Task Board (teacher_task_board.php):
Teachers can view their created tasks on the **Teacher Task Board**. This page fetches task details from the **tasks** table and allows for task updates or deletions.

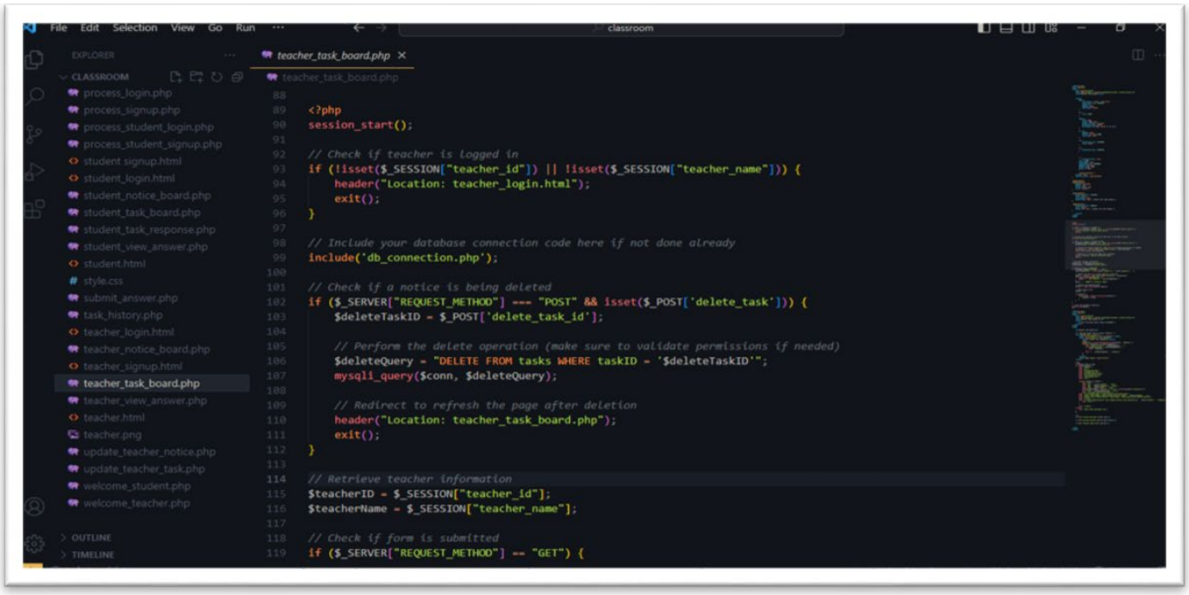


Teacher Task Board

Filter by Semester: All Semesters Filter

Task ID	Semester	Task Details	Created At	Action
30	1st	Write a 250-300 word essay on "The Effects of Social Media on Mental Health."	Feb 6, 2024 12:30 PM	Delete Update

Figure 11: frontend: teacher_task_board.php

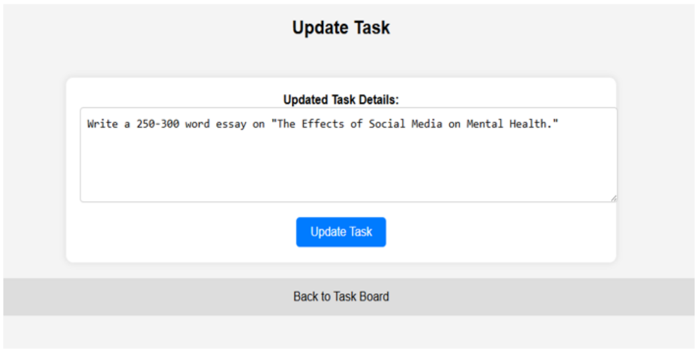


```php

<?php
session_start();

// Check if teacher is logged in
if (!isset($_SESSION["teacher_id"]) || !isset($_SESSION["teacher_name"])) {
    header("Location: teacher_login.html");
    exit();
}

// Include your database connection code here if not done already
include('db_connection.php');

// Check if a notice is being deleted
if ($_SERVER["REQUEST_METHOD"] === "POST" && isset($_POST['delete_task'])) {
    $deleteTaskID = $_POST['delete_task_id'];

    // Perform the delete operation (make sure to validate permissions if needed)
    $deleteQuery = "DELETE FROM tasks WHERE taskID = '$deleteTaskID'";
    mysqli_query($conn, $deleteQuery);

    // Redirect to refresh the page after deletion
    header("Location: teacher_task_board.php");
    exit();
}

// Retrieve teacher information
$teacherID = $_SESSION["teacher_id"];
$teacherName = $_SESSION["teacher_name"];

// Check if form is submitted
if ($_SERVER["REQUEST_METHOD"] == "GET") {
```

Figure 12: Backend: teacher_task_board.php

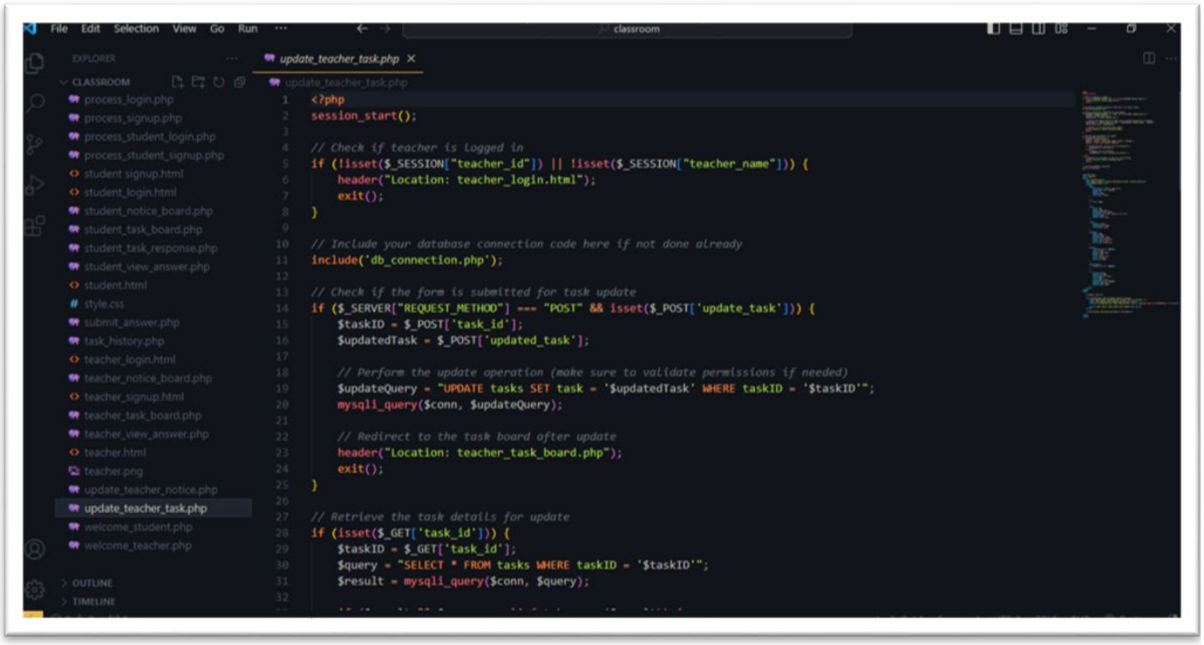
Update Task (update_teacher_task.php):

If a user notices an error in their task details, they can update the information by clicking the **Update** button. Upon clicking, they will be redirected to *update_teacher_task.php*, where they can modify the task details. After updating, the system will automatically redirect them back to the previous page *teacher_task_board.php*.



Write a 250-300 word essay on "The Effects of Social Media on Mental Health."

Figure 13: frontend: update_teacher_task.php



```php
<?php
session_start();

// Check if teacher is logged in
if (!isset($_SESSION["teacher_id"]) || !isset($_SESSION["teacher_name"])) {
    header("Location: teacher_login.html");
    exit();
}

// Include your database connection code here if not done already
include('db_connection.php');

// Check if the form is submitted for task update
if ($_SERVER["REQUEST_METHOD"] === "POST" && isset($_POST['update_task'])) {
    $taskID = $_POST['task_id'];
    $updatedTask = $_POST['updated_task'];

    // Perform the update operation (make sure to validate permissions if needed)
    $updateQuery = "UPDATE tasks SET task = '$updatedTask' WHERE taskID = '$taskID'";
    mysqli_query($conn, $updateQuery);

    // Redirect to the task board after update
    header("Location: teacher_task_board.php");
    exit();
}

// Retrieve the task details for update
if (isset($_GET['task_id'])) {
    $taskID = $_GET['task_id'];
    $query = "SELECT * FROM tasks WHERE taskID = '$taskID'";
    $result = mysqli_query($conn, $query);
```

Figure 14: Backend: update_teacher_task.php

Student Task Response (student_task_response.php):

When students respond to a task, the teacher is notified of the response count. The system retrieves response details from the **answers** and **tasks** tables.

After clicking the **Task Response** button, the user can view a table displaying student response details. Initially, responses can be evaluated based on the **mistake count** field. Additionally, the **timestamp** helps identify the fastest responder. This information is retrieved from the **answers** and **tasks** tables, where the **answers** table stores student submissions.

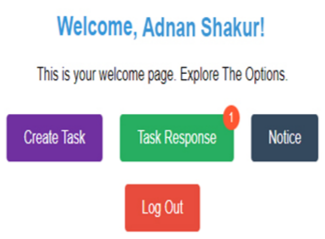
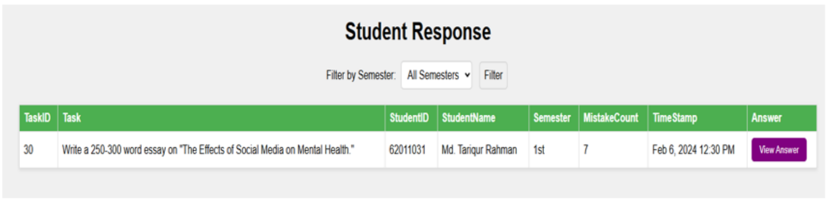
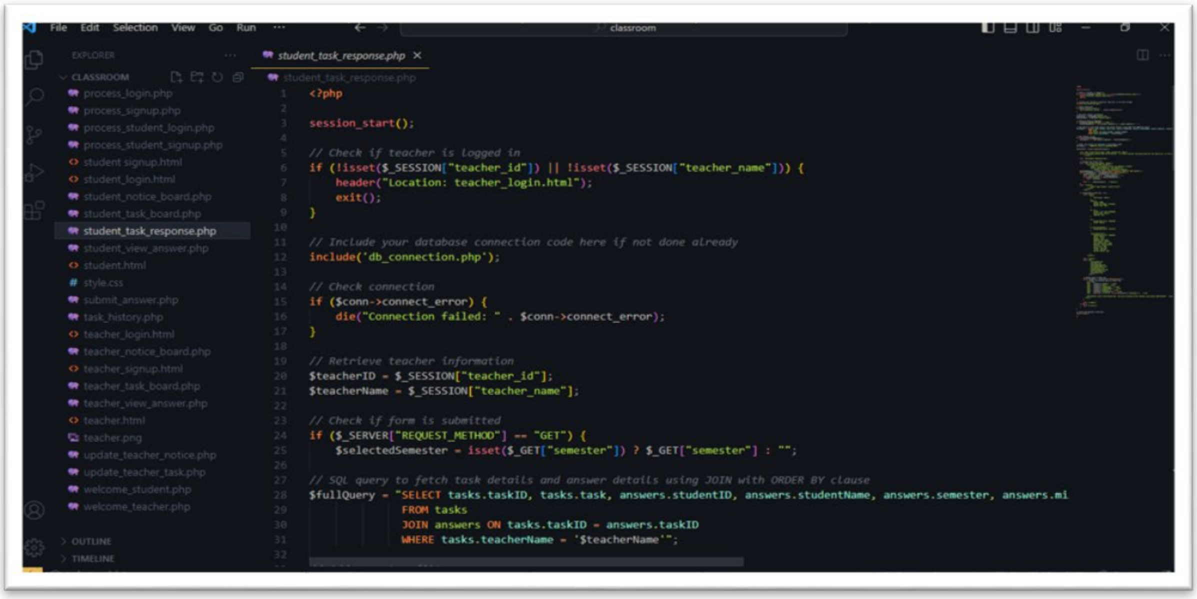


Figure 15: check student_task_response.php



TaskID	Task	StudentID	StudentName	Semester	MistakeCount	TimeStamp	Answer
30	Write a 250-300 word essay on "The Effects of Social Media on Mental Health."	62011031	Md. Tariqur Rahman	1st	7	Feb 6, 2024 12:30 PM	View Answer

Figure 16: frontend: student_task_response.php

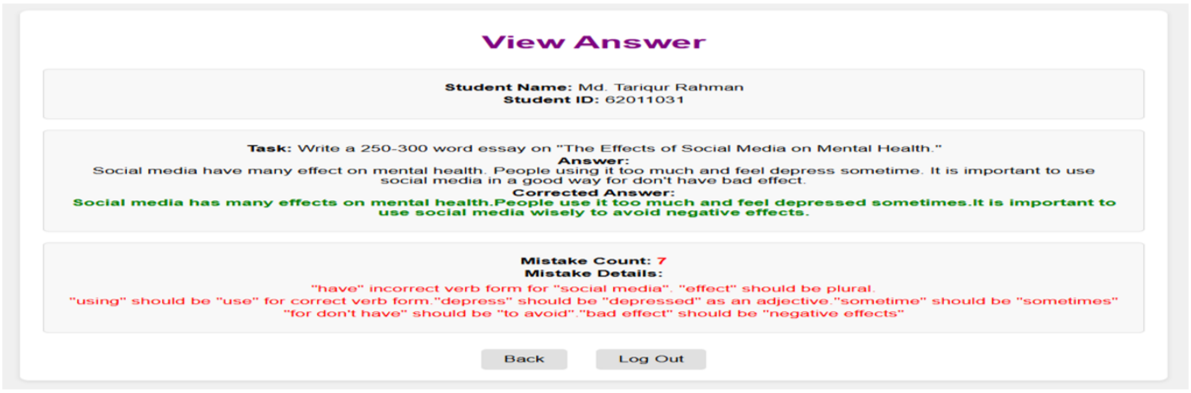


```
<?php

session_start();

// Check if teacher is logged in
if (!isset($_SESSION["teacher_id"]) || !isset($_SESSION["teacher_name"])) {
    header("Location: teacher_login.html");
    exit();
}

// Include your database connection code here if not done already
include('db_connection.php');

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// Retrieve teacher information
$teacherID = $_SESSION["teacher_id"];
$teacherName = $_SESSION["teacher_name"];

// Check if form is submitted
if ($_SERVER["REQUEST_METHOD"] == "GET") {
    $selectedSemester = isset($_GET["semester"]) ? $_GET["semester"] : "";

// SQL query to fetch task details and answer details using JOIN with ORDER BY clause
$fullQuery = "SELECT tasks.taskID, tasks.task, answers.studentID, answers.studentName, answers.semester, answers.mi
              FROM tasks
              JOIN answers ON tasks.taskID = answers.taskID
              WHERE tasks.teacherName = '$teacherName'";
```

Figure 17: Backend; student_task_response.php

View Answer (teacher_view_answer.php):

Teachers can view full student responses to tasks by clicking on the **View Answer** button. This information is fetched from the **tasks** and **answers** tables.



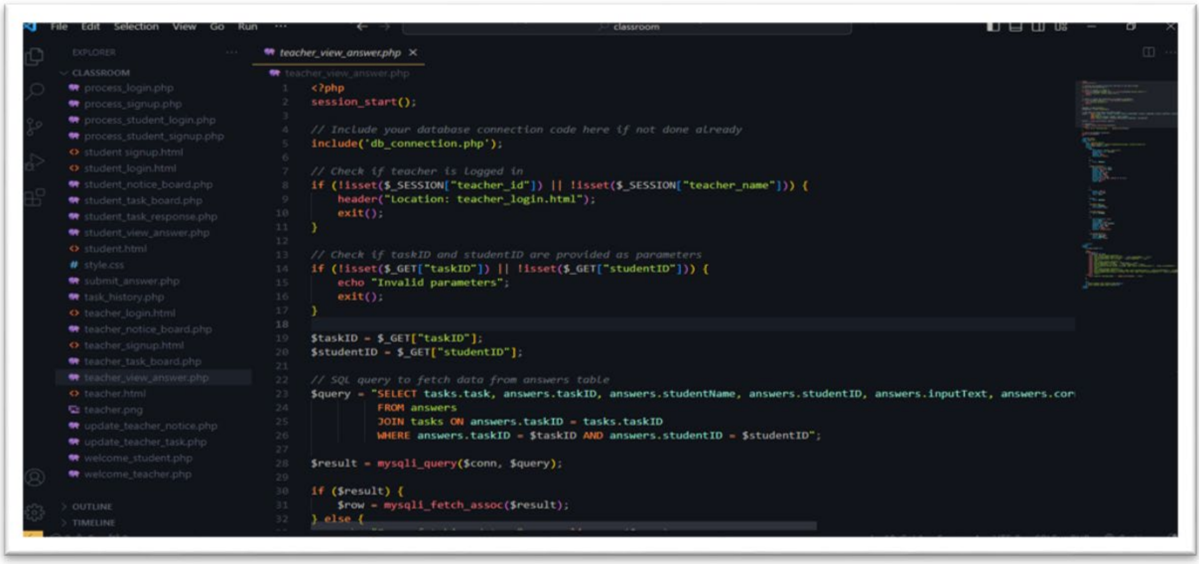
View Answer

Student Name: Md. Tariqur Rahman
Student ID: 62011031

Task: Write a 250-300 word essay on "The Effects of Social Media on Mental Health."
Answer:
Social media have many effect on mental health. People using it too much and feel depress sometime. It is important to use social media in a good way for don't have bad effect.
Corrected Answer:
Social media has many effects on mental health.People use it too much and feel depressed sometimes.It is important to use social media wisely to avoid negative effects.

Mistake Count: 7
Mistake Details:
"have" incorrect verb form for "social media". "effect" should be plural.
"using" should be "use" for correct verb form."depress" should be "depressed" as an adjective."sometime" should be "sometimes"
"for don't have" should be "to avoid"."bad effect" should be "negative effects"

Figure 18: frontend: teacher_view_answer.php



```
<?php
session_start();

// Include your database connection code here if not done already
include('db_connection.php');

// Check if teacher is logged in
if (!isset($_SESSION["teacher_id"]) || !isset($_SESSION["teacher_name"])) {
    header("Location: teacher_login.html");
    exit();
}

// Check if taskID and studentID are provided as parameters
if (!isset($_GET["taskID"]) || !isset($_GET["studentID"])) {
    echo "Invalid parameters";
    exit();
}

$taskID = $_GET["taskID"];
$studentID = $_GET["studentID"];

// SQL query to fetch data from answers table
$query = "SELECT tasks.task, answers.taskID, answers.studentName, answers.studentID, answers.inputText, answers.cor
          FROM answers
          JOIN tasks ON answers.taskID = tasks.taskID
          WHERE answers.taskID = $taskID AND answers.studentID = $studentID";

$result = mysqli_query($conn, $query);

if ($result) {
    $row = mysqli_fetch_assoc($result);
} else {
```

Figure 19: Backend: teacher_view_answer.php

Notice (create_notice.php):
Teachers can create notices for students, specifying the target semester or selecting all semesters. The information is stored in the **notices** table.

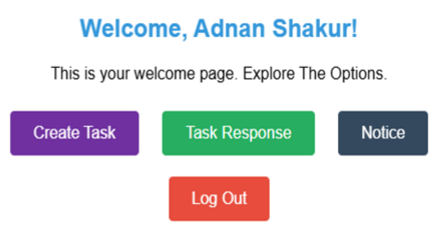


Figure 20: welcome teacher

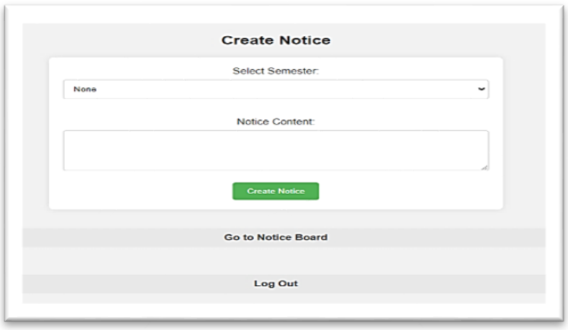
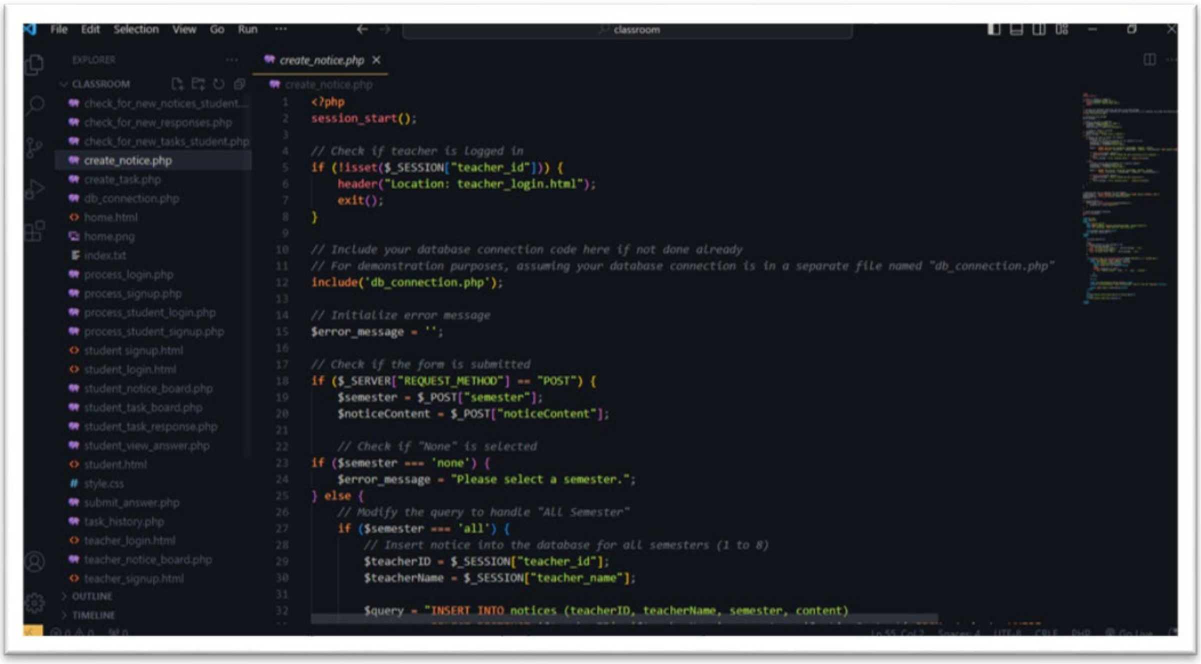


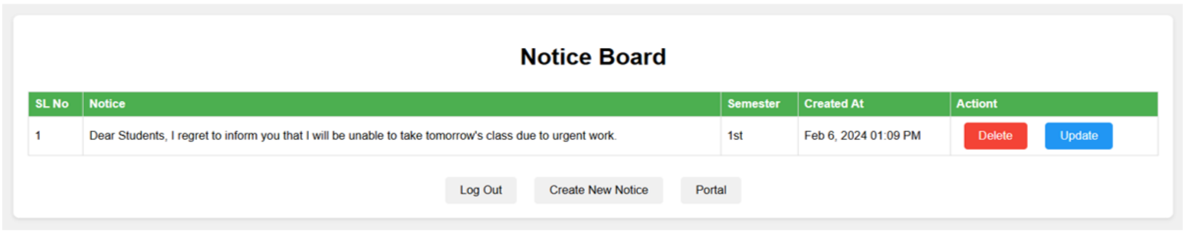
Figure 21: frontend: create_notice.php



```
<?php
session_start();

// Check if teacher is logged in
if (!isset($_SESSION["teacher_id"])) {
    header("Location: teacher_login.html");
    exit();
}

// Include your database connection code here if not done already
// For demonstration purposes, assuming your database connection is in a separate file named "db_connection.php"
include('db_connection.php');

// Initialize error message
$error_message = '';

// Check if the form is submitted
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $semester = $_POST["semester"];
    $noticeContent = $_POST["noticeContent"];

    // Check if "None" is selected
if ($semester === 'none') {
    $error_message = "Please select a semester.";
} else {
    // Modify the query to handle "All Semester"
    if ($semester === 'all') {
        // Insert notice into the database for all semesters (1 to 8)
        $teacherID = $_SESSION["teacher_id"];
        $teacherName = $_SESSION["teacher_name"];

        $query = "INSERT INTO notices (teacherID, teacherName, semester, content)
```

Figure 22: Backend: create_notice.php

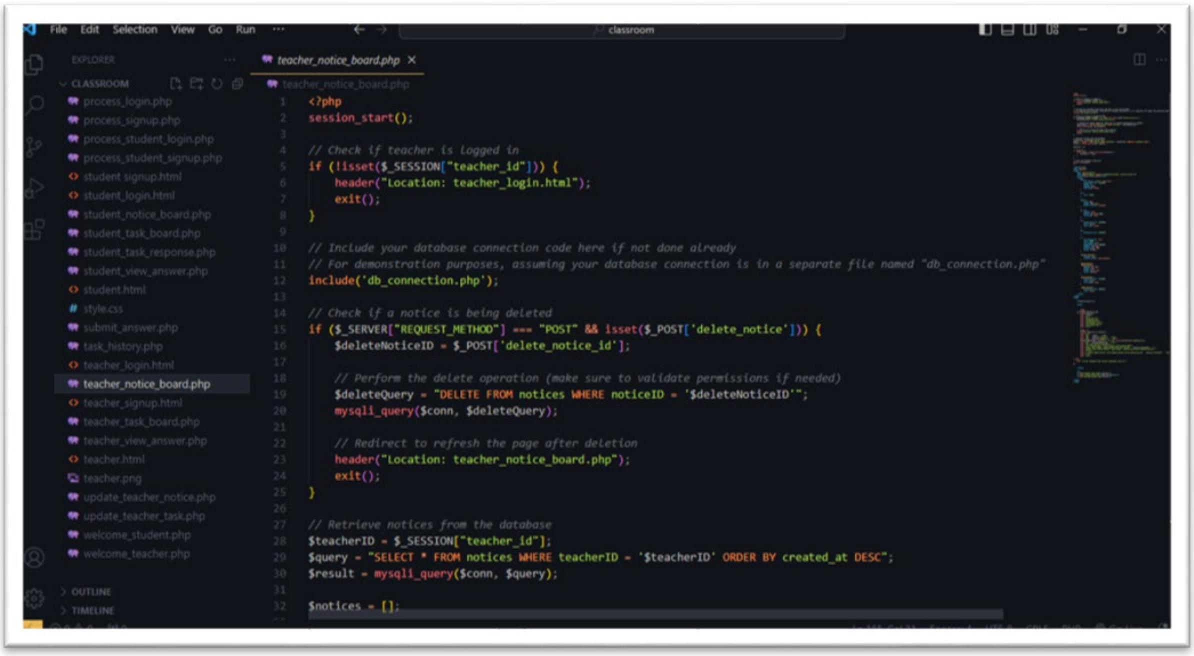
Go To Notice Board (teacher_notice_board.php):
Teachers can view and manage their notices, with options to update or delete them. There will be two buttons **Delete** and **Update**. So here if the user wants to delete his/her notice she/he can be able to do that.



Notice Board

SL No	Notice	Semester	Created At	Actiont
1	Dear Students, I regret to inform you that I will be unable to take tomorrow's class due to urgent work.	1st	Feb 6, 2024 01:09 PM	Delete Update

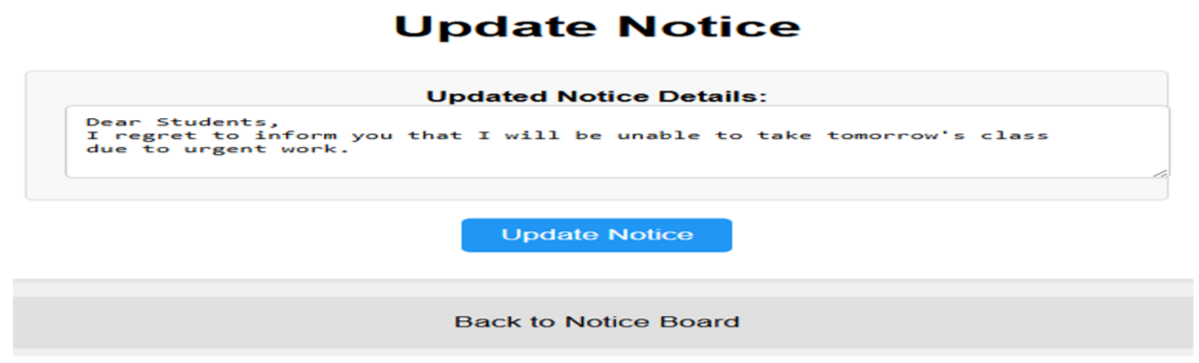
Figure 23: frontend: teacher_notice_board.php



```php
<?php
session_start();

// Check if teacher is logged in
if (!isset($_SESSION["teacher_id"])) {
    header("Location: teacher_login.html");
    exit();
}

// Include your database connection code here if not done already
// For demonstration purposes, assuming your database connection is in a separate file named "db_connection.php"
include('db_connection.php');

// Check if a notice is being deleted
if ($_SERVER["REQUEST_METHOD"] === "POST" && isset($_POST['delete_notice'])) {
    $deleteNoticeID = $_POST['delete_notice_id'];

    // Perform the delete operation (make sure to validate permissions if needed)
    $deleteQuery = "DELETE FROM notices WHERE noticeID = '$deleteNoticeID'";
    mysqli_query($conn, $deleteQuery);

    // Redirect to refresh the page after deletion
    header("Location: teacher_notice_board.php");
    exit();
}

// Retrieve notices from the database
$teacherID = $_SESSION["teacher_id"];
$query = "SELECT * FROM notices WHERE teacherID = '$teacherID' ORDER BY created_at DESC";
$result = mysqli_query($conn, $query);

$notices = [];
```

Figure 24: Backend: teacher_notice_board.php

Update (update_teacher_notice.php):
Teachers can update their notices through this page, which is triggered when the Update button is clicked.

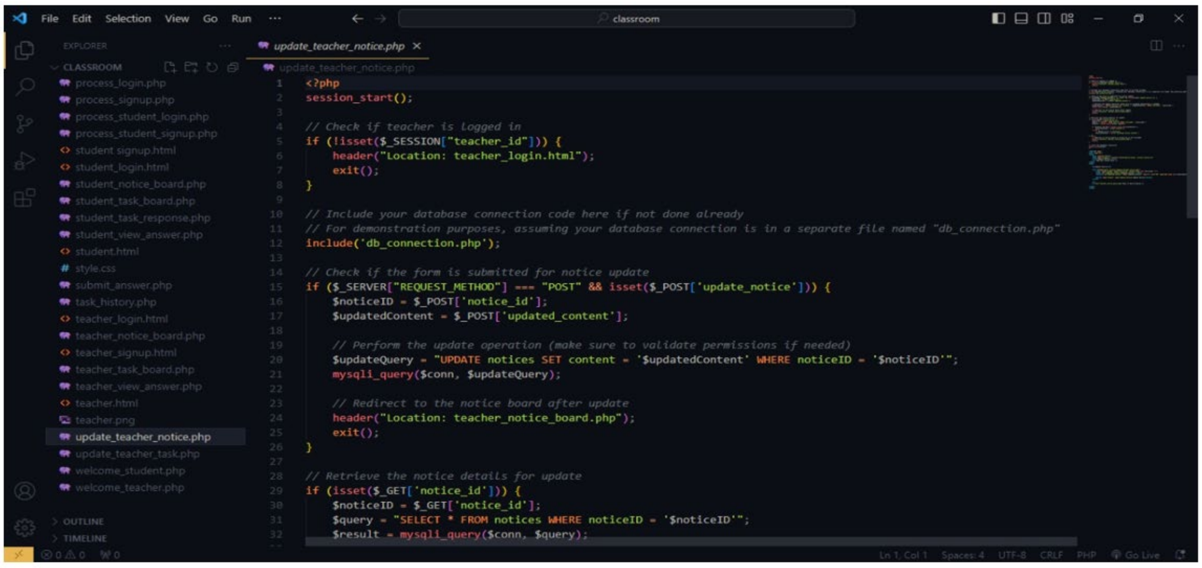


Update Notice

Updated Notice Details:

```
Dear Students,
I regret to inform you that I will be unable to take tomorrow's class
due to urgent work.
```

Figure 25: frontend: update_teacher_notice.php



```php
<?php
session_start();

// Check if teacher is logged in
if (!isset($_SESSION["teacher_id"])) {
    header("Location: teacher_login.html");
    exit();
}

// Include your database connection code here if not done already
// For demonstration purposes, assuming your database connection is in a separate file named "db_connection.php"
include('db_connection.php');

// Check if the form is submitted for notice update
if ($_SERVER["REQUEST_METHOD"] === "POST" && isset($_POST['update_notice'])) {
    $noticeID = $_POST['notice_id'];
    $updatedContent = $_POST['updated_content'];

    // Perform the update operation (make sure to validate permissions if needed)
    $updateQuery = "UPDATE notices SET content = '$updatedContent' WHERE noticeID = '$noticeID'";
    mysqli_query($conn, $updateQuery);

    // Redirect to the notice board after update
    header("Location: teacher_notice_board.php");
    exit();
}

// Retrieve the notice details for update
if (isset($_GET['notice_id'])) {
    $noticeID = $_GET['notice_id'];
    $query = "SELECT * FROM notices WHERE noticeID = '$noticeID'";
    $result = mysqli_query($conn, $query);
```

Figure 26: Backend: update_teacher_notice.php

Student Interface:

Home (Index.html):
User as a student have to click on '**Student**' button to enter into the student section.

Student Login/Sign Up (student.html):
After clicking '**Student**' button, Students are presented with options to either log in or sign up. New users must register before logging in.

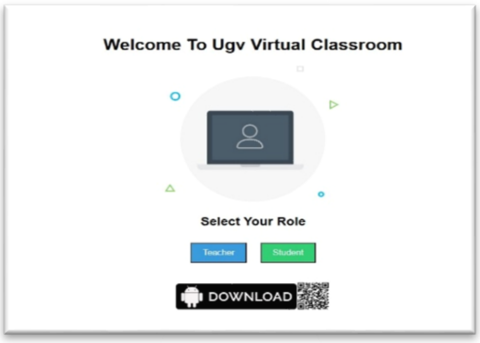


Figure 27: Home page

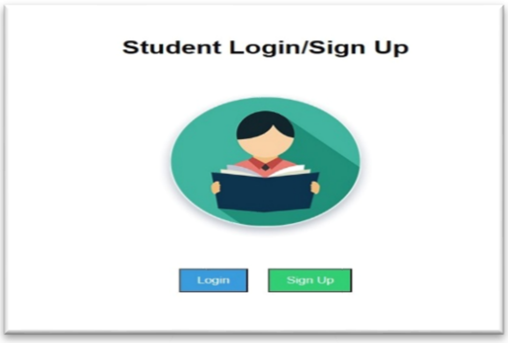


Figure 28: Student Login/Sign Up

Student Sign Up (student_signup.html & process_student_signup.php):
When a first-time user clicks the **Sign Up** button, they are presented with a registration form requiring essential details. Upon successful completion, the provided information is stored in the **students** table, a confirmation message is displayed, and the user is redirected to the **Student Login** page.

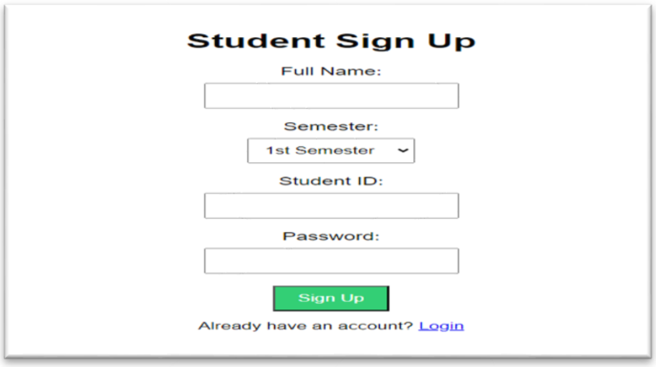
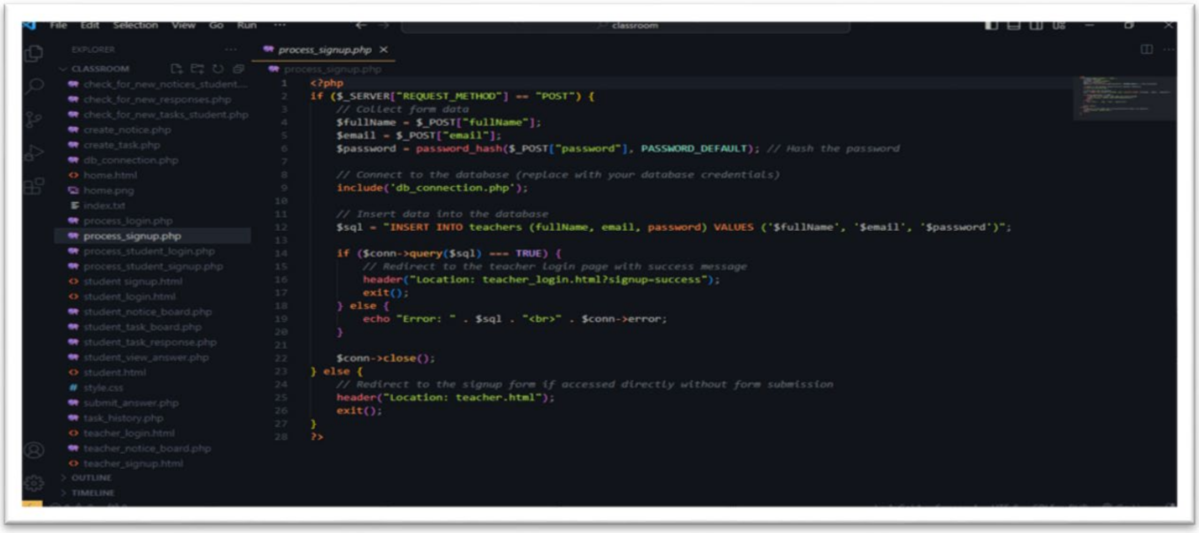


Figure 29: Frontend: student_signup.html



```php
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // Collect form data
    $fullName = $_POST["fullName"];
    $email = $_POST["email"];
    $password = password_hash($_POST["password"], PASSWORD_DEFAULT); // Hash the password

    // Connect to the database (replace with your database credentials)
    include('db_connection.php');

    // Insert data into the database
    $sql = "INSERT INTO teachers (fullName, email, password) VALUES ('$fullName', '$email', '$password')";

    if ($conn->query($sql) === TRUE) {
        // Redirect to the teacher login page with success message
        header("Location: teacher_login.html?signup=success");
        exit();
    } else {
        echo "Error: " . $sql . "<br>" . $conn->error;
    }

    $conn->close();
} else {
    // Redirect to the signup form if accessed directly without form submission
    header("Location: teacher.html");
    exit();
}
?>
```

Figure 30: Backend: process_student_signup.php

Student Login (student_login.html & process_student_login.php):
After successfully signing up, a student can log in using their signup credentials. If the login details do not match the registered information, a failure message stating “Invalid Student ID or Password” will be displayed. If the credentials are correct, the system will redirect the student to the "Student Welcome" page.

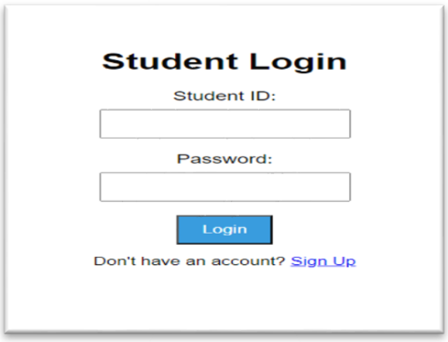
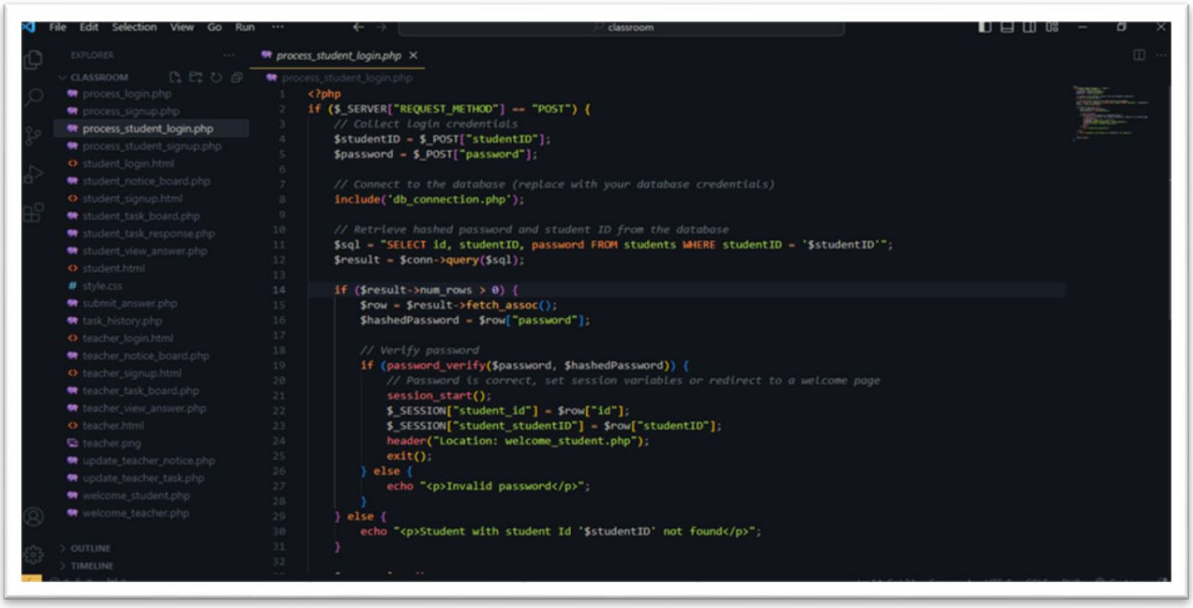


Figure 31: Frontend: student_login.html



```php
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // Collect login credentials
    $studentID = $_POST["studentID"];
    $password = $_POST["password"];

    // Connect to the database (replace with your database credentials)
    include('db_connection.php');

    // Retrieve hashed password and student ID from the database
    $sql = "SELECT id, studentID, password FROM students WHERE studentID = '$studentID'";
    $result = $conn->query($sql);

    if ($result->num_rows > 0) {
        $row = $result->fetch_assoc();
        $hashedPassword = $row["password"];

        // Verify password
        if (password_verify($password, $hashedPassword)) {
            // Password is correct, set session variables or redirect to a welcome page
            session_start();
            $_SESSION["student_id"] = $row["id"];
            $_SESSION["student_studentID"] = $row["studentID"];
            header("Location: welcome_student.php");
            exit();
        } else {
            echo "<p>Invalid password</p>";
        }
    } else {
        echo "<p>Student with student Id '$studentID' not found</p>";
    }
```

Figure 32: Backend: process_student_login.php

Student Welcome Page (welcome_student.php, check_for-new_tasks_student.php, and check_for_new_notices_student.php):
Upon successful login, students can access:

- Task Board
- Task History
- Notice Board

Notifications for new tasks and notices are provided using the check_for_new_tasks_student.php and check_for_new_notices_student.php scripts. Here, the student's image is displayed from the image directory of the University of Global Village (UGV) official website.

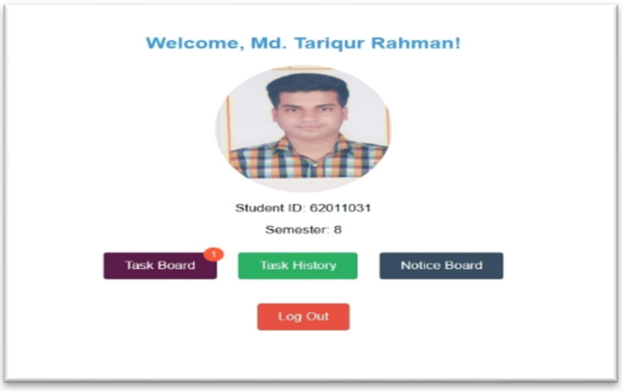
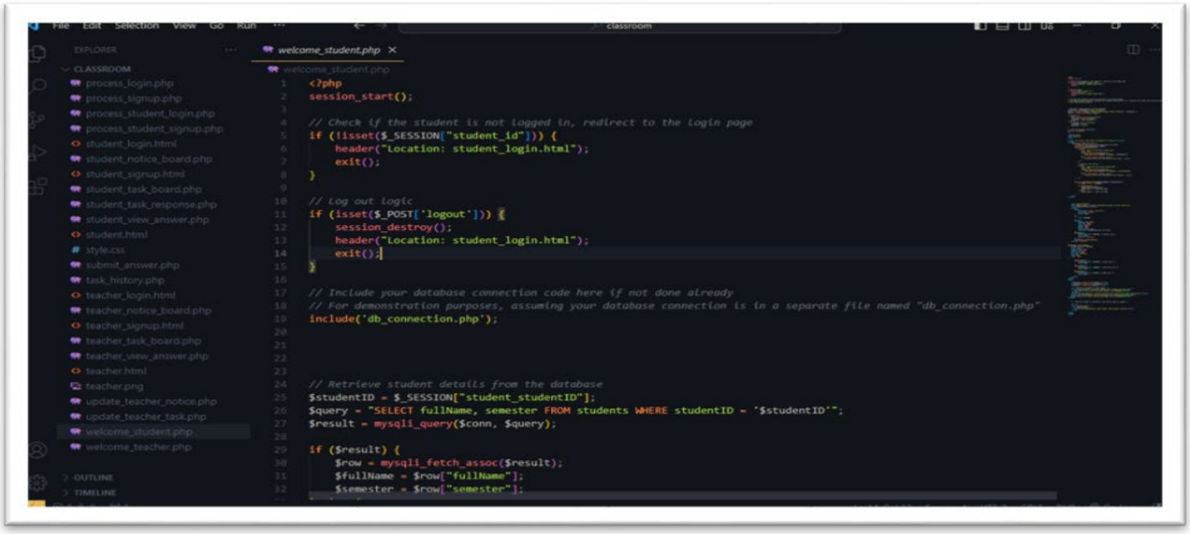
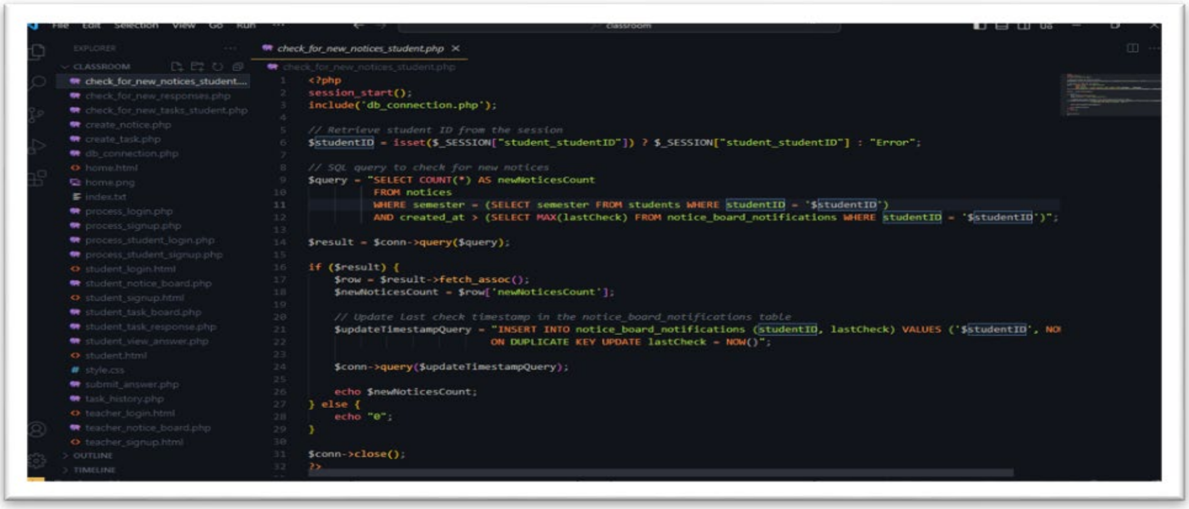


Figure 33: frontend: welcome_student.php



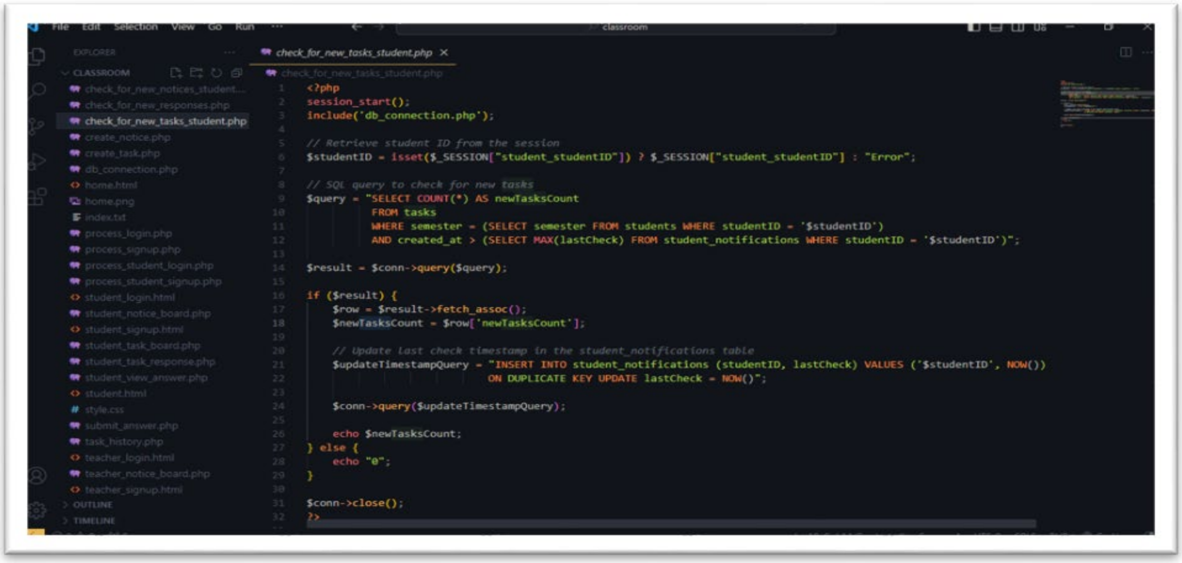
```php
<?php
session_start();

// Check if the student is not logged in, redirect to the login page
if (!isset($_SESSION["student_id"])) {
    header("Location: student_login.html");
    exit();
}

// Log out logic
if (isset($_POST['logout'])) {
    session_destroy();
    header("Location: student_login.html");
    exit();
}

// Include your database connection code here if not done already
// For demonstration purposes, assuming your database connection is in a separate file named "db_connection.php"
include('db_connection.php');




// Retrieve student details from the database
$studentID = $_SESSION["student_studentID"];
$query = "SELECT fullName, semester FROM students WHERE studentID = '$studentID'";
$result = mysqli_query($conn, $query);

if ($result) {
    $row = mysqli_fetch_assoc($result);
    $fullName = $row["fullName"];
    $semester = $row["semester"];
```

Figure 34: Backend: welcome_student.php



```php
<?php
session_start();
include('db_connection.php');

// Retrieve student ID from the session
$studentID = isset($_SESSION["student_studentID"]) ? $_SESSION["student_studentID"] : "Error";

// SQL query to check for new notices
$query = "SELECT COUNT(*) AS newNoticesCount
          FROM notices
          WHERE semester = (SELECT semester FROM students WHERE studentID = '$studentID')
          AND created_at > (SELECT MAX(lastCheck) FROM notice_board_notifications WHERE studentID = '$studentID')";

$result = $conn->query($query);

if ($result) {
    $row = $result->fetch_assoc();
    $newNoticesCount = $row['newNoticesCount'];

    // Update last check timestamp in the notice_board_notifications table
    $updateTimestampQuery = "INSERT INTO notice_board_notifications (studentID, lastCheck) VALUES ('$studentID', NO
                             ON DUPLICATE KEY UPDATE lastCheck = NOW()";

    $conn->query($updateTimestampQuery);

    echo $newNoticesCount;
} else {
    echo "0";
}

$conn->close();
?>
```

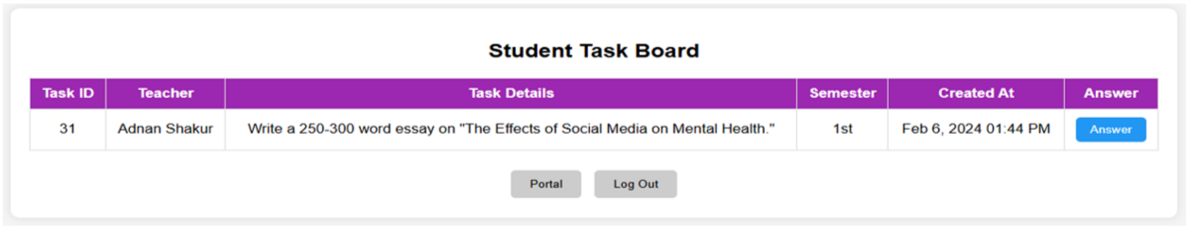
Figure 35: Backend: check_for_new_notices_student.php



```php
<?php
session_start();
include('db_connection.php');

// Retrieve student ID from the session
$studentID = isset($_SESSION["student_studentID"]) ? $_SESSION["student_studentID"] : "Error";

// SQL query to check for new tasks
$query = "SELECT COUNT(*) AS newTasksCount
          FROM tasks
          WHERE semester = (SELECT semester FROM students WHERE studentID = '$studentID')
          AND created_at > (SELECT MAX(lastCheck) FROM student_notifications WHERE studentID = '$studentID')";

$result = $conn->query($query);

if ($result) {
    $row = $result->fetch_assoc();
    $newTasksCount = $row['newTasksCount'];

    // Update last check timestamp in the student_notifications table
    $updateTimestampQuery = "INSERT INTO student_notifications (studentID, lastCheck) VALUES ('$studentID', NOW())
                             ON DUPLICATE KEY UPDATE lastCheck = NOW()";

    $conn->query($updateTimestampQuery);

    echo $newTasksCount;
} else {
    echo "0";
}

$conn->close();
?>
```

Figure 36: Backend: check_for_new_tasks_student.php

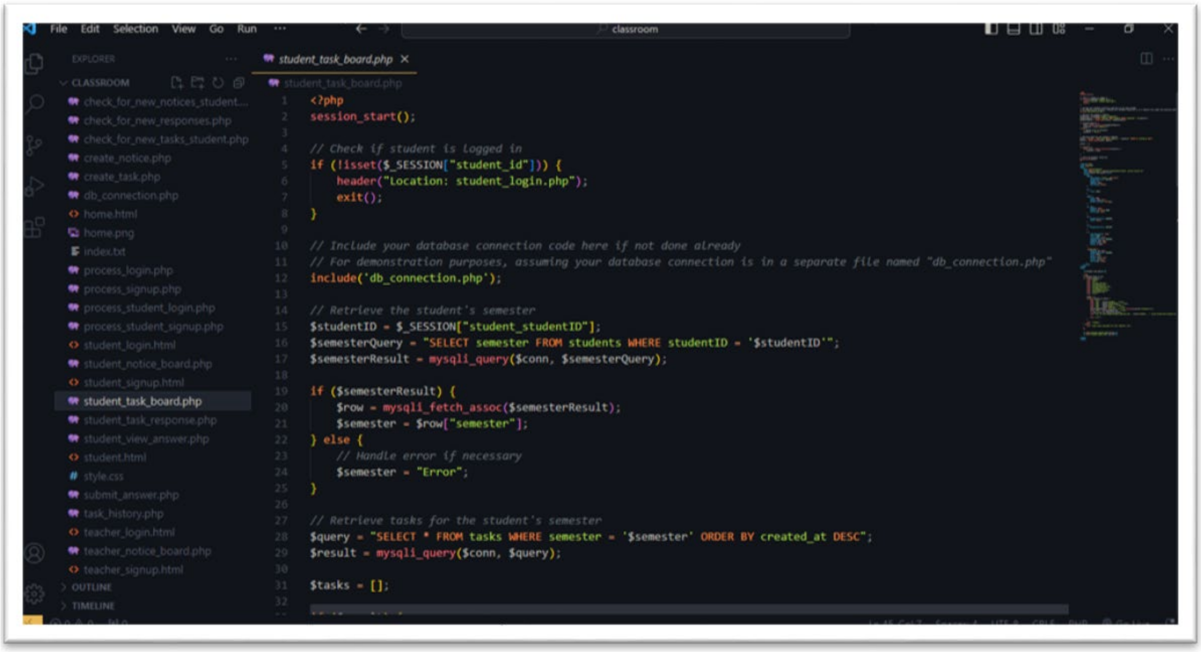
Task Board (student_task_board.php):
By clicking the **Task Board** button, the student will view the task table containing task details created by the teacher. There will also be another button named **Answer**.



Student Task Board

Task ID	Teacher	Task Details	Semester	Created At	Answer
31	Adnan Shakur	Write a 250-300 word essay on "The Effects of Social Media on Mental Health."	1st	Feb 6, 2024 01:44 PM	Answer

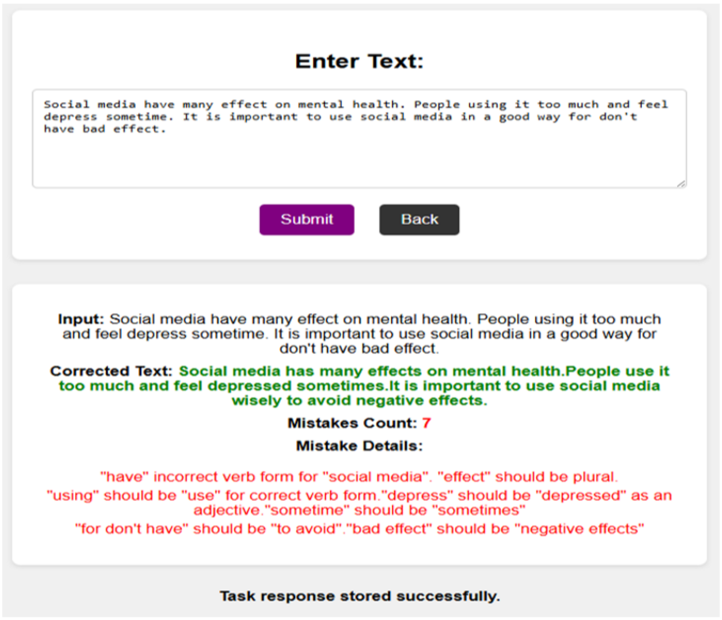
Figure 37: frontend: student_task_board.php



```php
<?php
session_start();

// Check if student is logged in
if (!isset($_SESSION["student_id"])) {
    header("Location: student_login.php");
    exit();
}

// Include your database connection code here if not done already
// For demonstration purposes, assuming your database connection is in a separate file named "db_connection.php"
include('db_connection.php');

// Retrieve the student's semester
$studentID = $_SESSION["student_studentID"];
$semesterQuery = "SELECT semester FROM students WHERE studentID = '$studentID'";
$semesterResult = mysqli_query($conn, $semesterQuery);

if ($semesterResult) {
    $row = mysqli_fetch_assoc($semesterResult);
    $semester = $row["semester"];
} else {
    // Handle error if necessary
    $semester = "Error";
}

// Retrieve tasks for the student's semester
$query = "SELECT * FROM tasks WHERE semester = '$semester' ORDER BY created_at DESC";
$result = mysqli_query($conn, $query);

$tasks = [];
```

Figure 38: Backend: student_task_board.php

Answer (submit_answer.php):
Students can submit responses for tasks. The **Grammar-Checker** API is used to check for grammar and spelling mistakes, displaying the corrected text and mistakes. To prevent duplicate submissions, the process of recording the submission in the **Submissions** database table and marking it in the session occurs only after the API call and grammar check have been completed. The submission will be inserted into the database and marked in the session only if no prior submission exists for the specified task ID and student ID combination in the **Submissions** table.



Enter Text:

```
Social media have many effect on mental health. People using it too much and feel
depress sometime. It is important to use social media in a good way for don't
have bad effect.
```

Input: Social media have many effect on mental health. People using it too much and feel depress sometime. It is important to use social media in a good way for don't have bad effect.

Corrected Text: **Social media has many effects on mental health.People use it too much and feel depressed sometimes.It is important to use social media wisely to avoid negative effects.**

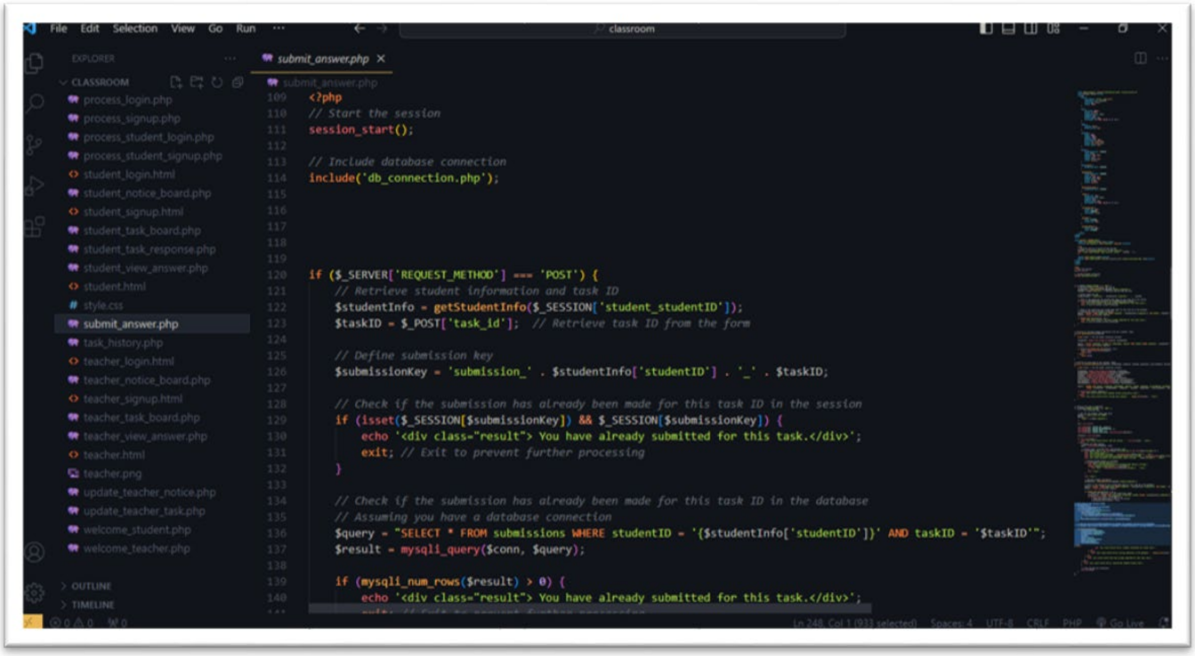
Mistakes Count: 7

Mistake Details:

"have" incorrect verb form for "social media". "effect" should be plural.
"using" should be "use" for correct verb form."depress" should be "depressed" as an adjective."sometime" should be "sometimes"
"for don't have" should be "to avoid"."bad effect" should be "negative effects"

Task response stored successfully.

Figure 39: frontend: submit_answer.php



```php
<?php
// Start the session
session_start();

// Include database connection
include('db_connection.php');




if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    // Retrieve student information and task ID
    $studentInfo = getStudentInfo($_SESSION['student_studentID']);
    $taskID = $_POST['task_id'];  // Retrieve task ID from the form

    // Define submission key
    $submissionKey = 'submission_' . $studentInfo['studentID'] . '_' . $taskID;

    // Check if the submission has already been made for this task ID in the session
    if (isset($_SESSION[$submissionKey]) && $_SESSION[$submissionKey]) {
        echo '<div class="result"> You have already submitted for this task.</div>';
        exit; // Exit to prevent further processing
    }

    // Check if the submission has already been made for this task ID in the database
    // Assuming you have a database connection
    $query = "SELECT * FROM submissions WHERE studentID = '{$studentInfo['studentID']}' AND taskID = '$taskID'";
    $result = mysqli_query($conn, $query);

    if (mysqli_num_rows($result) > 0) {
        echo '<div class="result"> You have already submitted for this task.</div>';
```

Figure 40: Backend: submit_answer.php

Task History (task_history.php):
Students can view a history of their submitted tasks by clicking the **Task History** button.

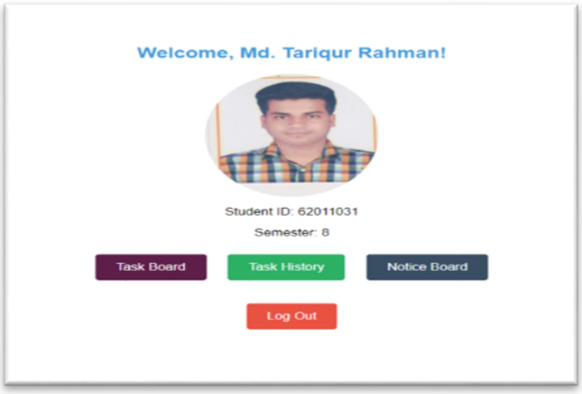
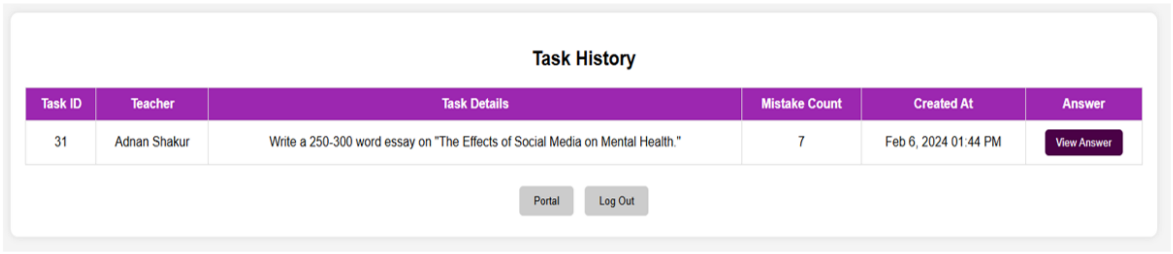


Figure 41: welcome student

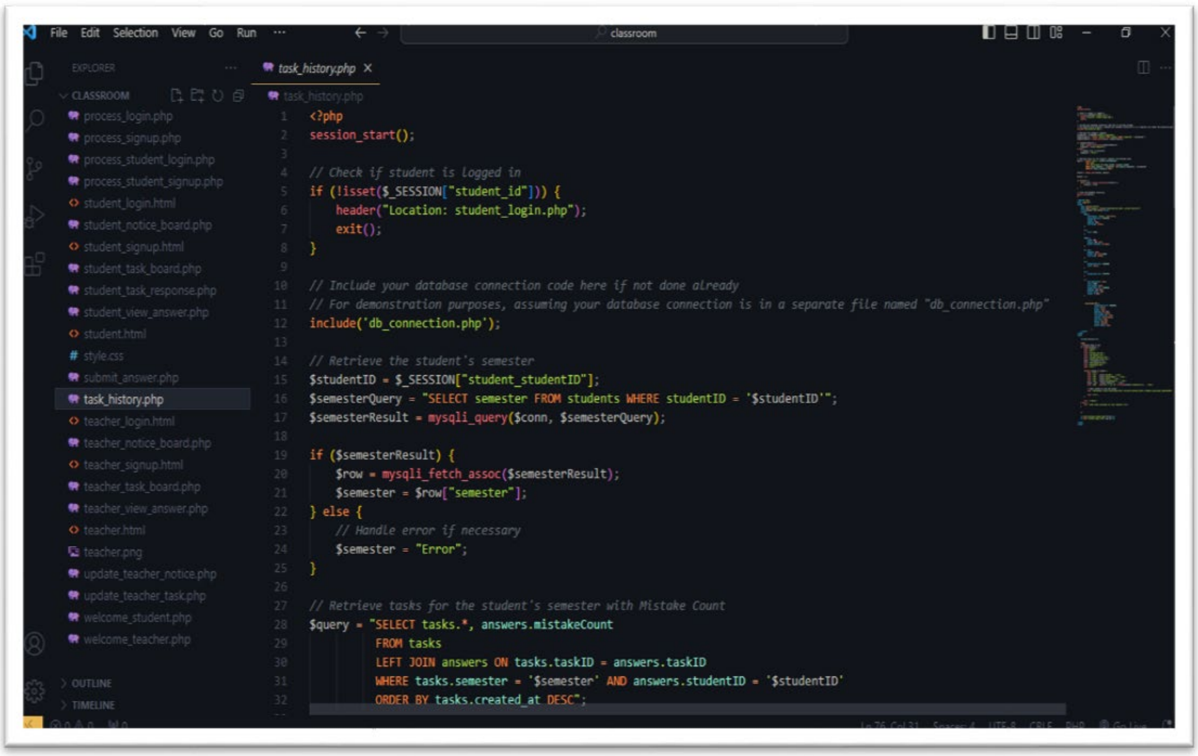
After clicking the **Task History** button, students will see all their recently submitted tasks along with a **View Answer** button.



Task History

Task ID	Teacher	Task Details	Mistake Count	Created At	Answer
31	Adnan Shakur	Write a 250-300 word essay on "The Effects of Social Media on Mental Health."	7	Feb 6, 2024 01:44 PM	View Answer

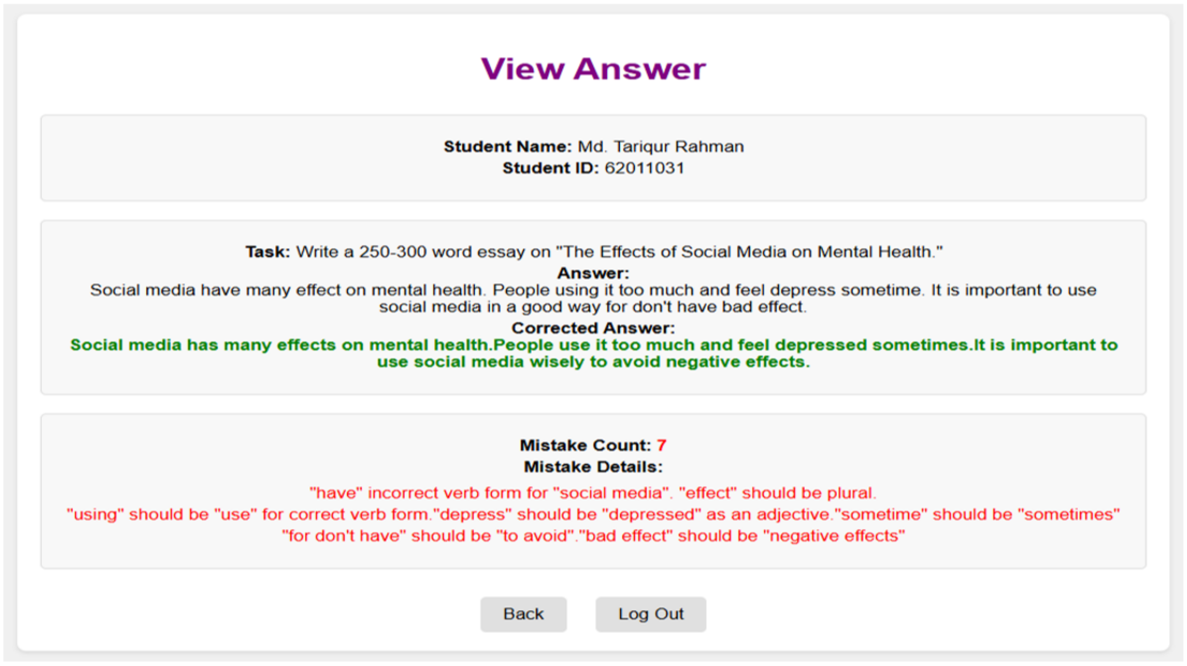
Figure 42: frontend: task_history.php



```php
<?php
session_start();

// Check if student is logged in
if (!isset($_SESSION["student_id"])) {
    header("Location: student_login.php");
    exit();
}

// Include your database connection code here if not done already
// For demonstration purposes, assuming your database connection is in a separate file named "db_connection.php"
include('db_connection.php');

// Retrieve the student's semester
$studentID = $_SESSION["student_studentID"];
$semesterQuery = "SELECT semester FROM students WHERE studentID = '$studentID'";
$semesterResult = mysqli_query($conn, $semesterQuery);

if ($semesterResult) {
    $row = mysqli_fetch_assoc($semesterResult);
    $semester = $row["semester"];
} else {
    // Handle error if necessary
    $semester = "Error";
}

// Retrieve tasks for the student's semester with Mistake Count
$query = "SELECT tasks.*, answers.mistakeCount
          FROM tasks
          LEFT JOIN answers ON tasks.taskID = answers.taskID
          WHERE tasks.semester = '$semester' AND answers.studentID = '$studentID'
          ORDER BY tasks.created_at DESC";
```

Figure 43: backend: task_history.php

View Answer (student_view_answer.php):
Students can view their submitted answers by clicking the **View Answer** button.



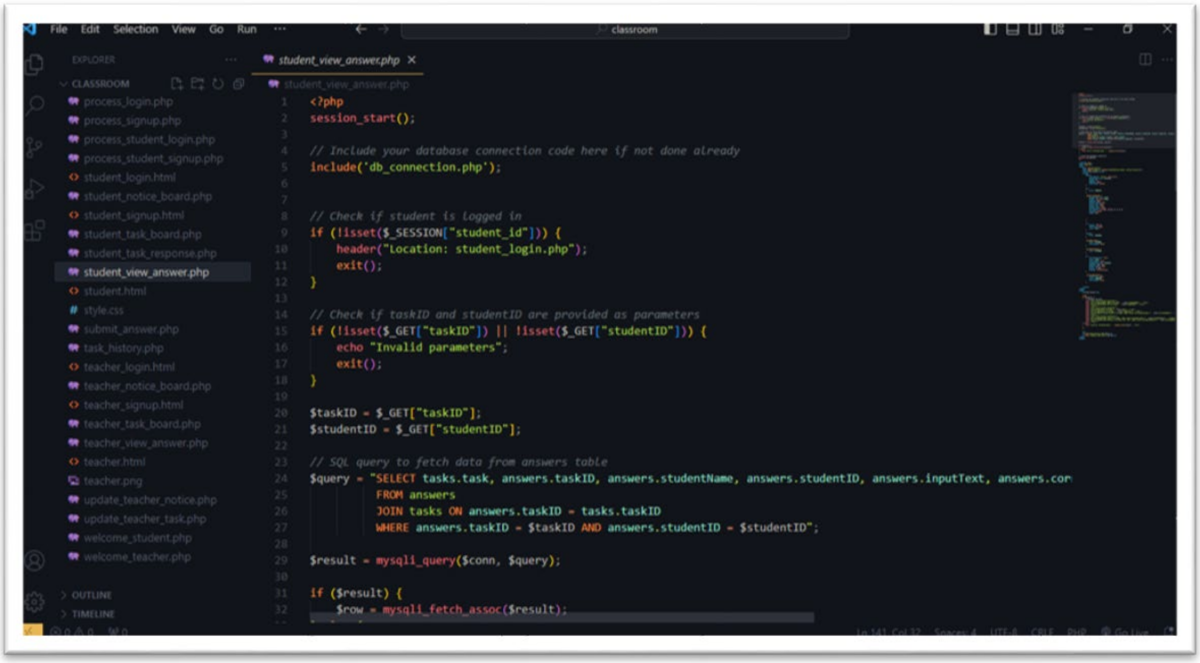
View Answer

Student Name: Md. Tariqur Rahman
Student ID: 62011031

Task: Write a 250-300 word essay on "The Effects of Social Media on Mental Health."
Answer:
Social media have many effect on mental health. People using it too much and feel depress sometime. It is important to use social media in a good way for don't have bad effect.
Corrected Answer:
Social media has many effects on mental health.People use it too much and feel depressed sometimes.It is important to use social media wisely to avoid negative effects.

Mistake Count: 7
Mistake Details:
"have" incorrect verb form for "social media". "effect" should be plural.
"using" should be "use" for correct verb form."depress" should be "depressed" as an adjective."sometime" should be "sometimes"
"for don't have" should be "to avoid"."bad effect" should be "negative effects"

Figure 44: frontend: student_view_answer.php



```php
<?php
session_start();

// Include your database connection code here if not done already
include('db_connection.php');


// Check if student is logged in
if (!isset($_SESSION["student_id"])) {
    header("Location: student_login.php");
    exit();
}

// Check if taskID and studentID are provided as parameters
if (!isset($_GET["taskID"]) || !isset($_GET["studentID"])) {
    echo "Invalid parameters";
    exit();
}

$taskID = $_GET["taskID"];
$studentID = $_GET["studentID"];

// SQL query to fetch data from answers table
$query = "SELECT tasks.task, answers.taskID, answers.studentName, answers.studentID, answers.inputText, answers.cor
          FROM answers
          JOIN tasks ON answers.taskID = tasks.taskID
          WHERE answers.taskID = $taskID AND answers.studentID = $studentID";

$result = mysqli_query($conn, $query);

if ($result) {
    $row = mysqli_fetch_assoc($result);
```

Figure 45: Backend: student_view_answer.php

Notice Board (student_notice_board.php):
Students can access notices posted by teachers through the **Notice Board**.

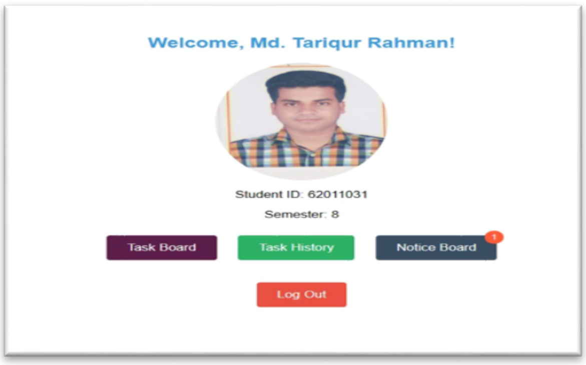
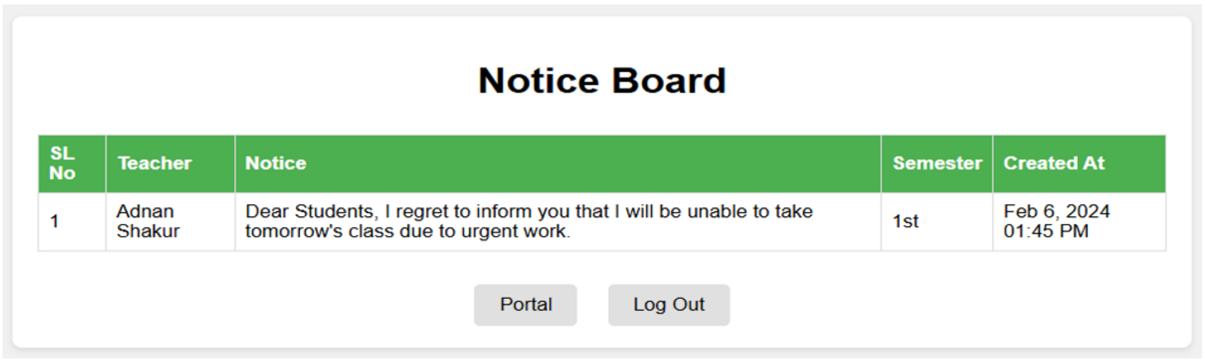


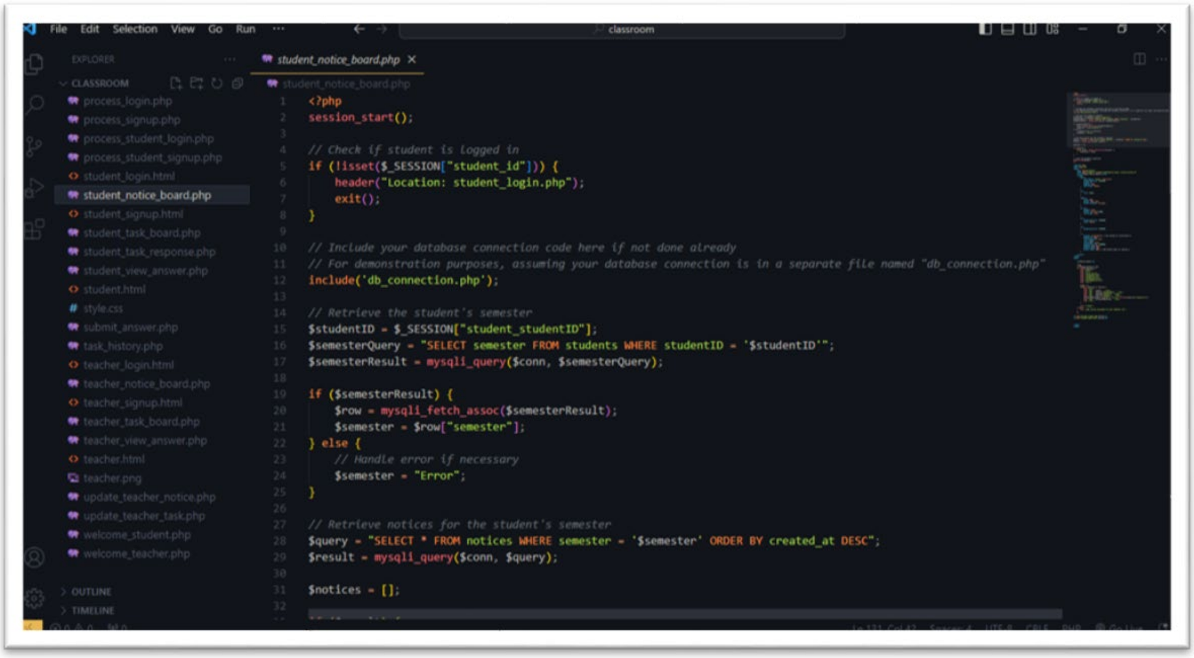
Figure 46: check notice



Notice Board

SL No	Teacher	Notice	Semester	Created At
1	Adnan Shakur	Dear Students, I regret to inform you that I will be unable to take tomorrow's class due to urgent work.	1st	Feb 6, 2024 01:45 PM

Figure 47: frontend: student_notice_board.php

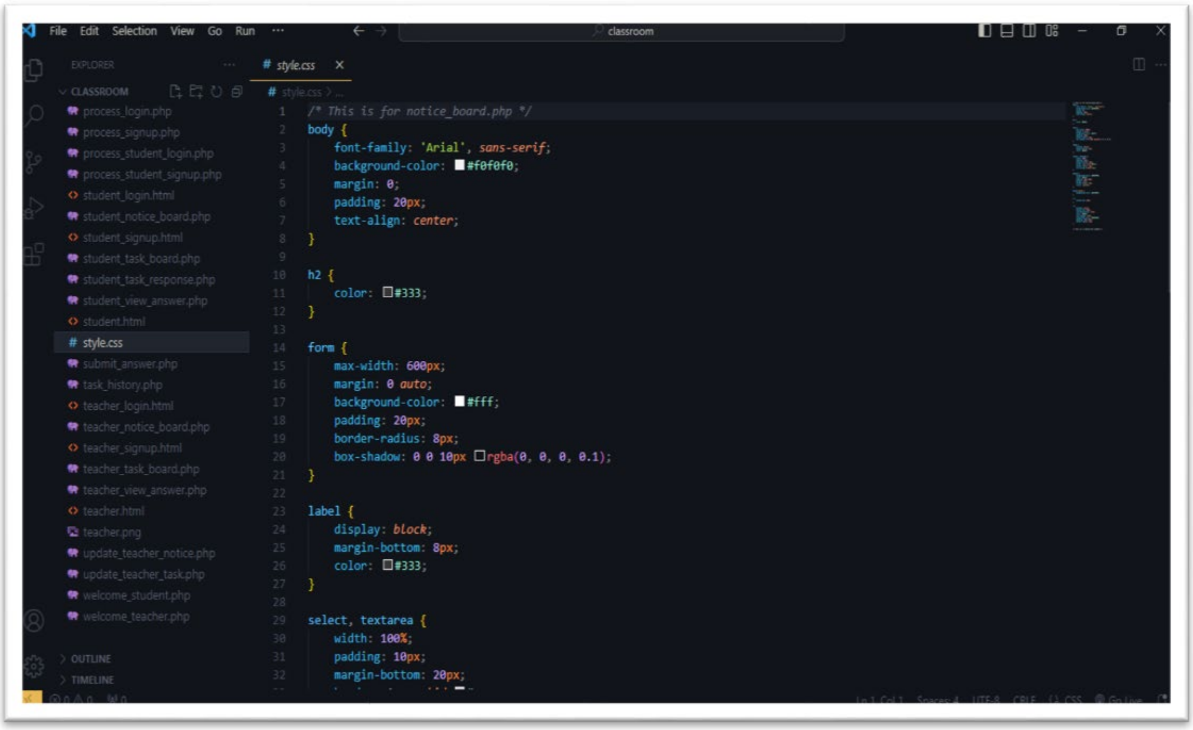


```php
<?php
session_start();

// Check if student is logged in
if (!isset($_SESSION["student_id"])) {
    header("Location: student_login.php");
    exit();
}

// Include your database connection code here if not done already
// For demonstration purposes, assuming your database connection is in a separate file named "db_connection.php"
include('db_connection.php');

// Retrieve the student's semester
$studentID = $_SESSION["student_studentID"];
$semesterQuery = "SELECT semester FROM students WHERE studentID = '$studentID'";
$semesterResult = mysqli_query($conn, $semesterQuery);

if ($semesterResult) {
    $row = mysqli_fetch_assoc($semesterResult);
    $semester = $row["semester"];
} else {
    // Handle error if necessary
    $semester = "Error";
}

// Retrieve notices for the student's semester
$query = "SELECT * FROM notices WHERE semester = '$semester' ORDER BY created_at DESC";
$result = mysqli_query($conn, $query);

$notices = [];
```

Figure 48: Backend: student_notice_board.php

Design: style.css

This file contains the CSS for styling the entire application, ensuring a visually appealing and user-friendly interface.



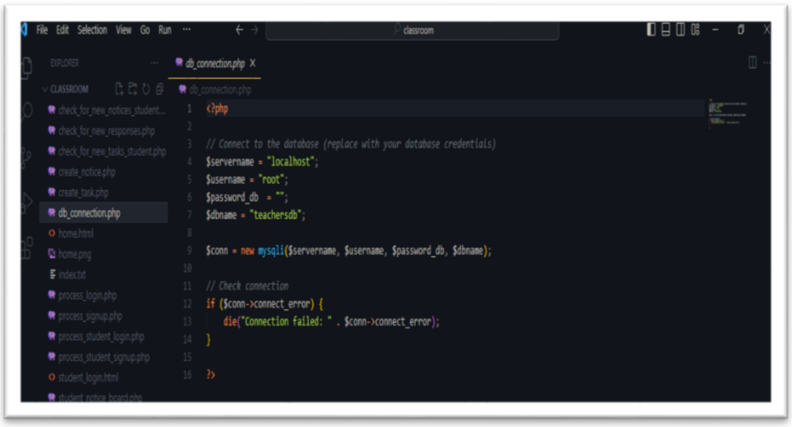
```css
/* This is for notice_board.php */
body {
    font-family: 'Arial', sans-serif;
    background-color: #f0f0f0;
    margin: 0;
    padding: 20px;
    text-align: center;
}

h2 {
    color: #333;
}

form {
    max-width: 600px;
    margin: 0 auto;
    background-color: #fff;
    padding: 20px;
    border-radius: 8px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}

label {
    display: block;
    margin-bottom: 8px;
    color: #333;
}

select, textarea {
    width: 100%;
    padding: 10px;
    margin-bottom: 20px;
```

Figure49: style.css

Database:
The project uses a MySQL database (teachersdb) with nine tables:

1. teachers
2. students
3. tasks
4. notices
5. answers
6. submissions
7. teacher_notifications
8. student_notifications
9. notice_board_notifications

The db_connection.php script handles the database connection.



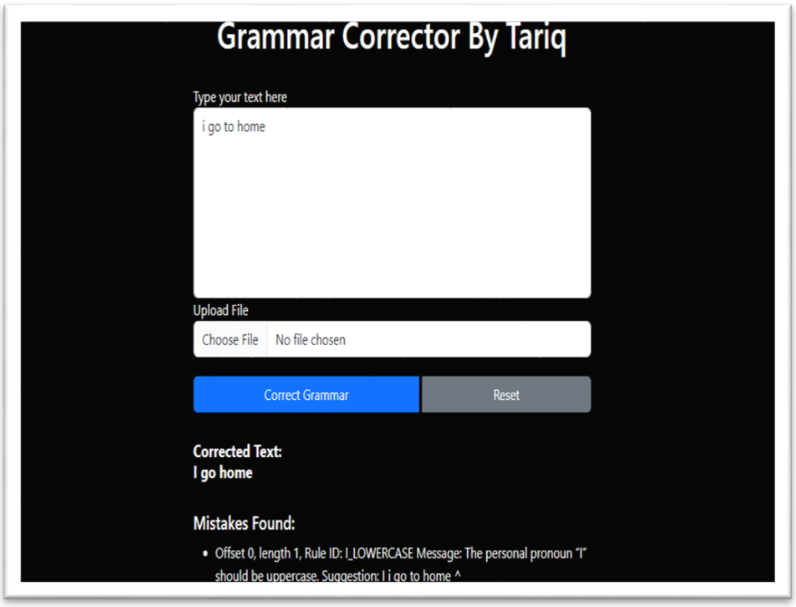
```php
<?php

// Connect to the database (replace with your database credentials)
$servername = "localhost";
$username = "root";
$password_db  = "";
$dbname = "teachersdb";

$conn = new mysqli($servername, $username, $password_db, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

?>
```

Figure 50: Databse-backend: db_connection.php

Grammar Checker App:
The Grammar Checker application is built using Python for the backend and HTML for the frontend. It consists of the following essential files:

1. **index.html**
2. **Model.py**
3. **App.py**

index.html:
The *index.html* file defines a user-friendly web interface for the Grammar and Spell Checker application, incorporating Bootstrap for enhanced styling and responsiveness. The interface includes two forms: one for direct text input and another for file uploads. Upon submission, the corrected text and identified grammatical errors are displayed on the page, processed through the Flask application.



i go to home

No file chosen

Corrected Text:
I go home

Mistakes Found:

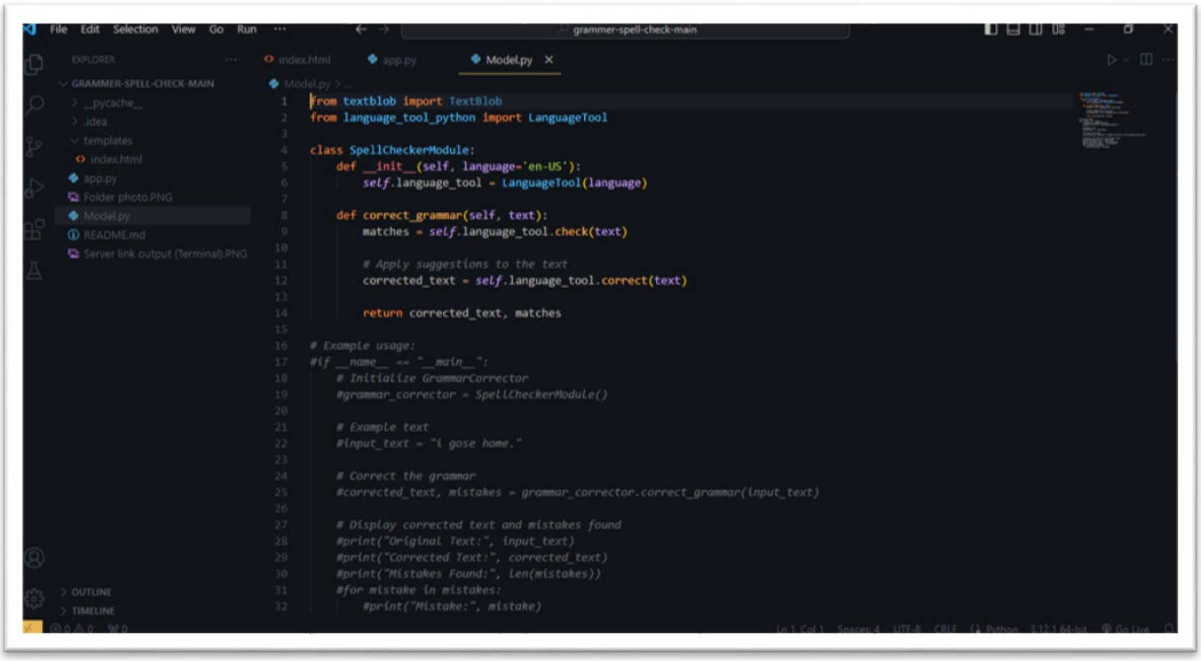
- Offset 0, length 1, Rule ID: I_LOWERCASE Message: The personal pronoun "I" should be uppercase. Suggestion: I i go to home ^

Figure 51: Frontend: index.html grammar correction tool

Model.py:
In the *model.py* file, I install the required modules, including Flask, LanguageTool, SpellChecker, and TextBlob. This Python script defines a **SpellCheckerModule** class that serves as both a spell and grammar checker. It utilizes the **TextBlob** library for spell checking and the **LanguageTool** library for grammar checking.

- The **correct_spell** method tokenizes the input text into words, corrects each word's spelling using **TextBlob**, and then reconstructs a corrected sentence.
- The **correct_grammar** method checks the input text for grammatical errors using **LanguageTool**, collects rule IDs for identified mistakes, and returns both the list of rule IDs and their count.

While the **correct_spell** method contains a comment suggesting an example input, the provided code does not currently utilize it.



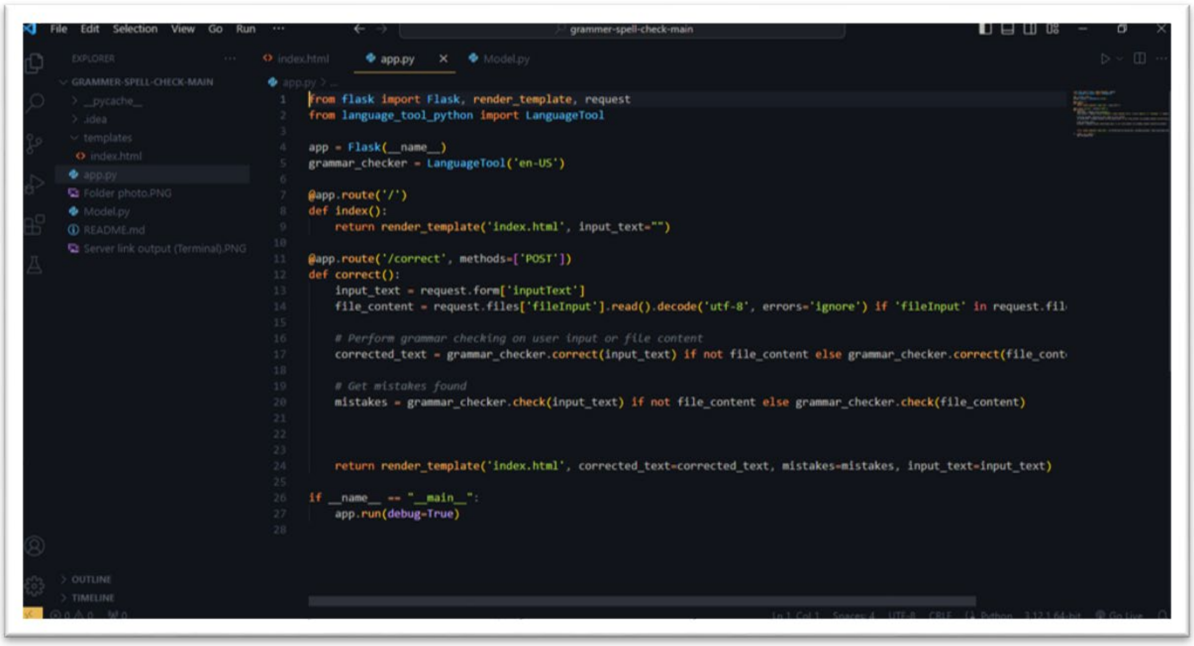
```python
from textblob import TextBlob
from language_tool_python import LanguageTool

class SpellCheckerModule:
    def __init__(self, language='en-US'):
        self.language_tool = LanguageTool(language)

    def correct_grammar(self, text):
        matches = self.language_tool.check(text)

        # Apply suggestions to the text
        corrected_text = self.language_tool.correct(text)

        return corrected_text, matches

# Example usage:
#if __name__ == "__main__":
    # Initialize GrammarCorrector
    #grammar_corrector = SpellCheckerModule()

    # Example text
    #input_text = "i gose home."

    # Correct the grammar
    #corrected_text, mistakes = grammar_corrector.correct_grammar(input_text)

    # Display corrected text and mistakes found
    #print("Original Text:", input_text)
    #print("Corrected Text:", corrected_text)
    #print("Mistakes Found:", len(mistakes))
    #for mistake in mistakes:
        #print("Mistake:", mistake)
```

Figure 52: Backend: Model.py

App.py:
This Python script uses the **Flask** web framework to create a simple web application. It includes three routes:

- **'/'** – Renders an HTML form for user input.
- **'/spell'** – Handles spell-checking requests.
- **'/grammar'** – Handles grammar-checking requests.

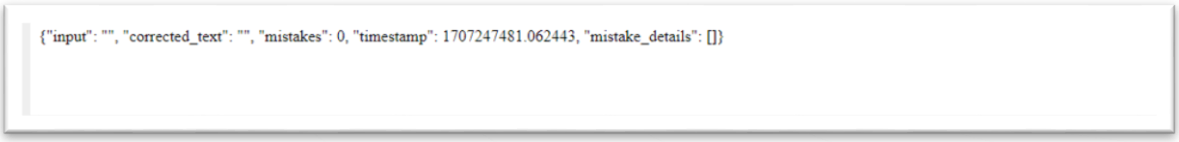
The **SpellCheckerModule** processes the text input, applies corrections, and provides the corrected version, which is then displayed on the web page.



```python
from flask import Flask, render_template, request
from language_tool_python import LanguageTool

app = Flask(__name__)
grammar_checker = LanguageTool('en-US')

@app.route('/')
def index():
    return render_template('index.html', input_text="")

@app.route('/correct', methods=['POST'])
def correct():
    input_text = request.form['inputText']
    file_content = request.files['fileInput'].read().decode('utf-8', errors='ignore') if 'fileInput' in request.fil

    # Perform grammar checking on user input or file content
    corrected_text = grammar_checker.correct(input_text) if not file_content else grammar_checker.correct(file_cont

    # Get mistakes found
    mistakes = grammar_checker.check(input_text) if not file_content else grammar_checker.check(file_content)



    return render_template('index.html', corrected_text=corrected_text, mistakes=mistakes, input_text=input_text)

if __name__ == "__main__":
    app.run(debug=True)
```

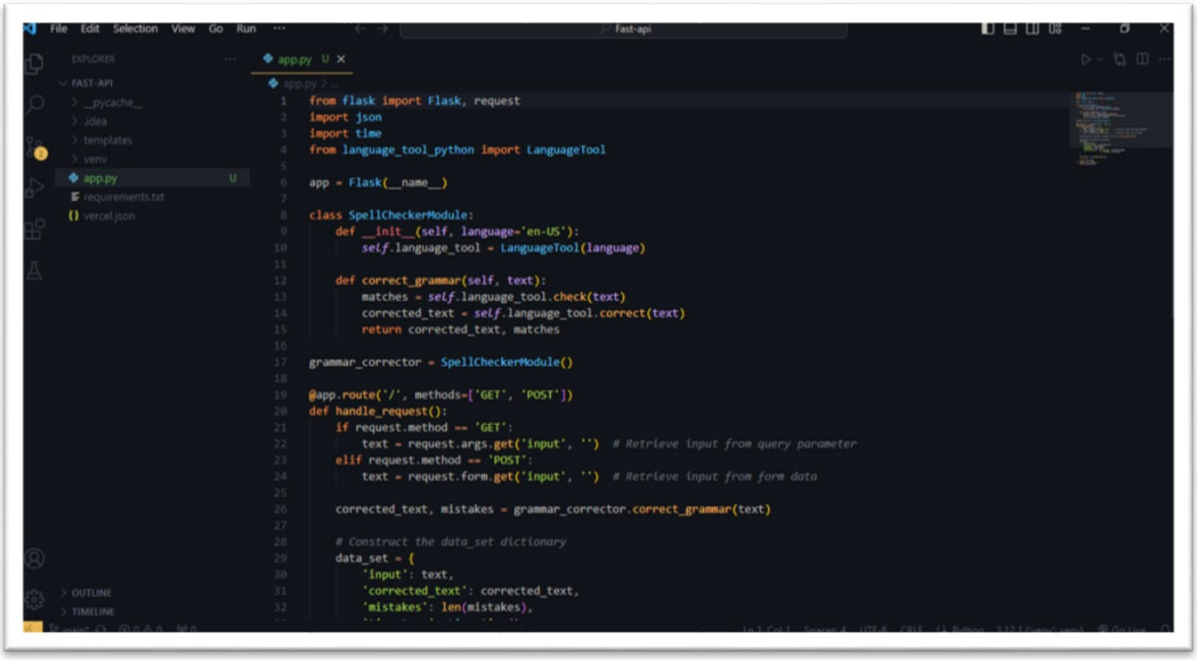
Figure 53: Backend: App.py

Grammar Checker API:
For my PHP project, I have developed an API for the Grammar Checker. In this process, I modified the *model.py* file and renamed it *app.py*. To meet the project requirements, the API is designed to receive requests for corrected text, mistake count, and mistake messages from the input text.



```
{"input": "", "corrected_text": "", "mistakes": 0, "timestamp": 1707247481.062443, "mistake_details": []}
```

Figure 54: API Result



```python
from flask import Flask, request
import json
import time
from language_tool_python import LanguageTool

app = Flask(__name__)

class SpellCheckerModule:
    def __init__(self, language='en-US'):
        self.language_tool = LanguageTool(language)

    def correct_grammar(self, text):
        matches = self.language_tool.check(text)
        corrected_text = self.language_tool.correct(text)
        return corrected_text, matches

grammar_corrector = SpellCheckerModule()

@app.route('/', methods=['GET', 'POST'])
def handle_request():
    if request.method == 'GET':
        text = request.args.get('input', '')  # Retrieve input from query parameter
    elif request.method == 'POST':
        text = request.form.get('input', '')  # Retrieve input from form data

    corrected_text, mistakes = grammar_corrector.correct_grammar(text)

    # Construct the data_set dictionary
    data_set = {
        'input': text,
        'corrected_text': corrected_text,
        'mistakes': len(mistakes),
```

Figure 55: Backend: Grammar-Checker-API(app.py)

In this script, I use the **Flask**, **LanguageTool**, and **SpellChecker** modules. The application runs on the local server at http://127.0.0.1:5000/.

V. Results

A total of 50 students participated in the survey, which consisted of both Likert-scale questions and open-ended responses. The students were asked to evaluate the grammar correction tool based on several factors, including its impact on grammar proficiency, ease of use, time efficiency, and overall engagement. The following table presents the results from the Likert-scale portion of the survey:

Question	Strongly Disagree (1)	Disagree (2)	Neutral (3)	Agree (4)	Strongly Agree (5)	Average Rating
How helpful was the grammar correction tool in improving your grammar skills?	2 (4%)	4 (8%)	10 (20%)	18 (36%)	16 (32%)	4.0
How easy was it to understand the feedback provided by the tool?	1 (2%)	3 (6%)	7 (14%)	20 (40%)	19 (38%)	4.2
Did the tool help you identify recurring grammar mistakes in your writing?	0 (0%)	2 (4%)	5 (10%)	23 (46%)	20 (40%)	4.3
How confident do you feel about your grammar after using the tool?	0 (0%)	3 (6%)	9 (18%)	21 (42%)	17 (34%)	4.1
To what extent did the grammar correction tool engage you in your learning process?	3 (6%)	5 (10%)	10 (20%)	18 (36%)	14 (28%)	3.9
How much time did the tool save you in correcting grammar mistakes compared to manual corrections?	1 (2%)	4 (8%)	12 (24%)	19 (38%)	14 (28%)	4.0
How likely are you to recommend the grammar correction tool to other students in future English courses?	0 (0%)	2 (4%)	7 (14%)	21 (42%)	20 (40%)	4.2

VI. Data Analysis and Interpretation

The data analysis reveals several key insights into the effectiveness and impact of the grammar correction tool on students' learning experience:

- **Tool Effectiveness**: The majority of students rated the tool positively in terms of its ability to improve grammar skills, with an average rating of **4.0**. **68%** of students agreed or strongly agreed that the tool helped them improve their grammar proficiency.
- **Ease of Use**: The tool was found to be relatively easy to use, with an average rating of **4.2** for feedback clarity. **78%** of students found the feedback provided by the tool to be easy to understand.
- **Identifying Recurring Mistakes**: The tool was especially effective in helping students identify recurring grammar mistakes, with a high rating of **4.3**. This suggests that the tool's feedback was both insightful and actionable for students.
- **Confidence in Grammar**: Students reported increased confidence in their grammar after using the tool, with **76%** of students expressing greater confidence in their grammar skills, as reflected by an average rating of **4.1**.
- **Engagement**: The tool received a moderate rating for engagement (**3.9**), indicating that while the tool was helpful, some students may have felt that additional interactive features could further enhance their learning experience.
- **Time Efficiency**: The tool was regarded as time-saving, with an average rating of **4.0**, suggesting that it reduced the amount of time spent on manual corrections.
- **Recommendation**: The tool received a high recommendation rate, with **82%** of students stating they would recommend it to others in future courses, evidenced by an average rating of **4.2**.

These results suggest that the grammar correction tool has had a positive impact on students' grammar learning and their overall experience in the English Composition course. The tool's ability to provide clear and timely feedback, identify recurring errors, and save time on manual corrections were some of the key strengths highlighted by students.

Qualitative Feedback

In addition to the quantitative data, students were asked to provide qualitative feedback regarding the tool's usability and overall effectiveness. Key themes emerged from their responses:

1. **Usability and Feedback Clarity**: Students appreciated the clarity and simplicity of the tool's feedback. Many students reported that the feedback helped them understand their errors and provided clear explanations of how to correct them.

o "The immediate feedback was very helpful, especially for common grammar mistakes like subject-verb agreement."

o "The suggestions for corrections were very clear and easy to follow."

2. **Suggestions for Improvement**: Several students suggested that the tool could benefit from more advanced grammar explanations and examples tailored to individual mistakes.

o "It would be helpful if the tool could offer more detailed explanations of why certain corrections are made."

o "I would love to see more examples of correct usage based on my errors."

3. **Impact on Confidence**: Many students expressed that the tool helped them feel more confident in their writing, particularly when it came to correcting grammatical errors.

o "I now feel more confident when writing because I know how to spot and fix my mistakes."

o "The tool helped me improve my writing and grammar, which made me feel more secure in my assignments."

VII. Limitations and Future Improvements

While the web-based grammar correction tool offers significant benefits in improving grammatical accuracy and facilitating automated feedback, it has certain limitations. One key drawback is its focus on grammar correction without addressing higher-order writing concerns such as coherence, organization, and rhetorical effectiveness. While students receive real-time grammatical feedback, the tool does not yet provide suggestions on sentence structure, academic tone, or lexical variety, which are essential for developing advanced writing skills. Additionally, automated feedback may not always capture the nuances of complex writing tasks, making instructor intervention necessary for more sophisticated corrections.

Another limitation is the tool's reliance on existing NLP libraries, which may not always detect context-specific errors or idiomatic expressions accurately. Furthermore, its effectiveness has only been tested within a single institutional setting, meaning its adaptability to different educational contexts and proficiency levels remains an area for further investigation. Additionally, security measures have not yet been fully implemented, as this was a pilot project. Once integrated into an institutional setting, robust security protocols will be necessary to protect student data, prevent unauthorized access, and ensure compliance with institutional policies.

Notably, this tool currently uses a free API for grammar correction. If a paid API were used, we could expect more accurate and sophisticated results, potentially enhancing its overall effectiveness in providing high-quality grammatical feedback.

To enhance the tool's effectiveness and expand its capabilities, the following improvements should be considered:

1. **Advanced AI Features**: Enhancing contextual grammar analysis to detect errors beyond basic rule-based corrections, allowing for deeper linguistic understanding.
2. **Expanded Feedback Mechanisms**: Providing more detailed explanations and tailored examples to help students not only identify mistakes but also understand the rationale behind corrections.
3. **Accessibility Enhancements**: Incorporating text-to-speech functionality to support students with disabilities and those who prefer auditory learning methods.
4. **Progress Tracking and Achievement Badges**: Implementing a progress report system where students can track their grammatical improvements over time and earn achievement badges for milestones, fostering motivation and engagement.
5. **Student and Teacher Analytics Dashboard**: Developing a comprehensive analytics dashboard that allows both students and teachers to monitor improvement over time.

o Students can track their progress on a monthly basis, comparing their current performance with previous months.

o Teachers can view a summary of each student's progress at a glance, including the percentage of improvement compared to prior months.

o This feature would provide valuable insights into individual and collective learning trends, enabling personalized instruction and targeted interventions.

6. **Enhanced Security Measures**: Implementing secure authentication, encrypted data storage, and access controls to ensure student data privacy and compliance with institutional guidelines.

This project was initially developed as a pilot study to assess its feasibility and effectiveness in addressing grammar instruction challenges in large classrooms. While institutional funding could facilitate further enhancements, making the tool more sophisticated and widely applicable, the primary objective was to explore its potential in an educational setting. This study provides a foundational framework and also aims to inspire future EdTech researchers to explore and build upon this work, further advancing digital solutions for language learning.

VIII. Conclusion

This study highlights the potential of technology-driven solutions in enhancing grammar instruction, particularly in large classrooms where personalized feedback is challenging. By integrating Natural Language Processing (NLP) technologies, the proposed tool streamlines the error correction process, increases student engagement, and reduces the burden on educators. While the tool has shown positive results in improving grammatical accuracy, addressing its current limitations—such as expanding beyond grammar correction to include style suggestions and adaptive feedback mechanisms—will further enhance its effectiveness. Moving forward, refining its features to accommodate diverse learning needs and integrating it into a broader writing support system will ensure its long-term impact in digital language learning. Ultimately, this research underscores the role of technology in bridging gaps in language education, paving the way for more interactive and personalized learning experiences.

References

[1] Alfaki, I. M. (2015). University Students' English Writing Problems: Diagnosis and Remedy. *International Journal of English Language Teaching, 3*(3), 40–52.

[2] Biswas, M. S. H. (2021). Investigating The Problems of Writing in the English Language: Perspective of Tertiary-Level Students. *International Journal of Scientific and Research Publications, 11*(6), 327–346.

[3] Cavaleri, M., & Dianati, S. (2016). You Want Me to Check Your Grammar Again? The Usefulness of an Online Grammar Checker as Perceived by Students. *Journal of Academic Language and Learning, 10*(1), A223–A236.

[4] Crovitz, D., Devereaux, M. D., & Morgan, C. M. (2022). *Next Level Grammar for A Digital Age: Teaching with Social Media and Online Tools for Rhetorical Understanding and Critical Creation.* Routledge.

[5] Mamadiyorova, S. O. Q., & Rahmonqulova, A. S. (2024). The Challenges of Teaching Grammar in Language Classes. *Modern Education and Development, 15*(9), 267–271.

[6] Maulidina, P., & Wibowo, H. (2022). The Use of Grammarly Tools to Enrich Students' Writing Ability. *Journal of Applied Linguistics and Language Research, 18*(2), 179–189.

[7] Mishra, P., & Koehler, M. J. (2006). Technological Pedagogical Content Knowledge: A Framework for Teacher Knowledge. *Teachers College Record, 108*(6), 1017–1054. Https://Doi.Org/10.1111/J.1467- 9620.2006.00684.X

[8] Nurmatova, Z. A. (2025). Teaching Grammar to Non-Native Speakers: Challenges and Strategies. *International Journal of Artificial Intelligence, 5*(1), 303–306.

[9] Obari, H. (2024). Language Learning at The Brink of Singularity: AI's Impact on Educational Paradigms. In *Proceedings of The International CALL Research Conference 2024* (Pp. 197–206). Https://Doi.Org/10.29140/9780648184485-30

[10] Solfiyatuzzahro, Santihastuti, A., & Erfan. (2019). Grammatical Accuracy Using Teacher's Written Corrective Feedback. *Eralingua: Jurnal Pendidikan Bahasa Asing dan Sastra, 3*(2), 75–85.