UNIVERSITY OF JORDAN

Student Name	Std_ID	Section
Hussein Taha Yousef	0217884	1
Obey saber mosa	0215692	1
Raghad Nafi Azzouqa	0216289	2
Rama mamon albaddawi	0228601	2
Shahed Mohammad Idiabieh	0222930	2

King Abdullah II School for Information Technology Course Title: Human computer interaction

STUDY MATE

Supervisor name: Dr. Mohammad Abu shariaDate: 4th/June /2025

TABLE OF CONTENTS

1.0 INTRODUCTION

In today’s academic landscape, students often face difficulties in managing their study schedules, prioritizing tasks, and staying motivated throughout the semester. The lack of structured planning and time management tools tailored to university students can result in procrastination, stress, and decreased academic performance. Although some digital tools exist, many are either overly complex or not designed specifically with students’ needs in mind.

This project proposes the design and development of StudyMate, a study management application that empowers students to organize their academic life effectively. The system will follow Human-Computer Interaction (HCI) principles, ensuring a user-centered design that promotes ease of use, accessibility, and engagement. The main features include course tracking, a Pomodoro timer for study sessions, a daily to-do list, and visual feedback through performance analytics.

This report presents a full development life cycle of the system, starting from need analysis and system requirements, moving through prototyping and evaluation, and concluding with future improvement suggestions. The project emphasizes designing with the user in mind, supported by usability testing and continuous refinement.

1.1 NEED ANALYSIS AND DESCRIPTION

Need:

University students often struggle with time management and organizing their academic tasks, leading to poor productivity and elevated stress levels during peak academic periods.

Description:

StudyMate addresses this gap by offering a streamlined and intuitive interface that helps students plan their study time, set goals, and track progress. By integrating tools such as a Pomodoro timer, course task planner, and progress dashboard, the system encourages structured study behavior and promotes academic efficiency. The human-centered design ensures that users can easily interact with the system, adapt to its functionalities, and incorporate it into their daily routines.

1.2 PROJECT CONSTRAINTS

- The development tools and resources must remain within the team's available free and open-source options, with a maximum budget of 0 JD (since development is done by students).

- The full system prototype and documentation must be completed and submitted by May 27, 2025

1.3 System Environment

Need:

Ensure that students and team members involved in development and testing are familiar with the tools and platforms used in designing and using the system.

Description:

The system will be prototyped using accessible, cross-platform tools such as Canva (for interface design) and optionally HTML/CSS/JavaScript (for high-fidelity prototyping). Basic training or onboarding materials (such as a user manual or tutorial video) will be provided for demonstration and testing. The system will be optimized for use on laptops and mobile devices, ensuring accessibility for the target student audience.

1.4 PROJECT SOFTWARE AND HARDWARE REQUIREMENTS

◈ Software Requirements:

The project will utilize Canva as the main tool for medium- and high-fidelity interface design, allowing for collaborative design and prototyping. For optional high-fidelity coding, HTML, CSS, and JavaScript may be used for basic UI interaction. Documentation will be prepared using Microsoft Word, and presentation materials will be created using PowerPoint and Canva. Version control and collaboration may be managed through GitHub or Google Drive.

◈ Hardware Requirements:

The development and testing of the system will be conducted on standard personal laptops and mobile devices with internet access. Minimum specifications include 4GB RAM, dual-core processors, and updated web browsers such as Chrome or Firefox. All tools and platforms selected are lightweight and accessible, ensuring smooth operation without the need for specialized hardware.

1.5 PROJECT SCHEDULE

- The project will be executed over six main phases spread across the semester timeline:

- Week 1–2: Topic Selection & Team Formation – Finalizing the idea and assigning roles to team members.

- Week 3–4: Requirements Analysis – Identifying user needs and system features through basic surveys.

- Week 5–6: Design Phase – Creating low- and medium-fidelity prototypes using Figma and paper sketches.

- Week 7–8: Implementation – Building high-fidelity interfaces and optional coded UI using HTML/CSS.

- Week 9: Evaluation – Conducting usability tests (heuristic and cooperative evaluation).

- Week 10: Final Report & Presentation – Compiling the documentation, preparing the poster, and final submission.

- The team will follow a weekly progress review to ensure alignment with the project deadline on May 27, 2025.

2.0 PROJECT BACKGROUND AND EXISTING TECHNOLOGIES

In recent years, students have increasingly turned to digital tools to support their academic lives, including apps for scheduling, task management, and time tracking. While many applications such as Notion, Google Calendar, Trello, and Forest offer various productivity features, they often fall short in fully addressing the specific needs of university students in terms of simplicity, personalization, and academic goal tracking. Additionally, these tools can sometimes be overwhelming due to feature overload or steep learning curves.

Study Mate emerges as a focused solution tailored specifically for students, combining essential study tools into one unified platform with a clean and user-friendly interface. Unlike existing technologies, Study Mate prioritizes simplicity, motivational features, and a study-specific workflow that aligns with students' academic habits. It builds on the strengths of existing tools while addressing their limitations, guided by Human-Computer Interaction principles to ensure an engaging, intuitive, and accessible experience.

3.0 SOFTWARE REQUIREMENTS DOCUMENT

3.1 TARGETED USERS

The primary users of the system are undergraduate university students, aged between 18 to 25 years, who are enrolled in various academic programs. These users typically deal with multiple courses, assignments, and exams simultaneously. They are digital natives, comfortable with mobile and desktop platforms, and actively seek tools that improve their academic productivity. The system is also designed to accommodate students with basic digital literacy, making the interface intuitive and easy to navigate for all user level.

3.2 REQUIREMENTS GATHERING AND CUSTOMER FEEDBACK TECHNIQUES

To ensure the system meets user needs, several requirements elicitation techniques were applied. These include:

- Online questionnaires distributed among students at the University of Jordan to identify common pain points in time and task management.

- Short interviews conducted with a sample of 5 students to gather qualitative insights about their study habits.

- Observation of how students currently use tools like calendars, notes apps, or to-do lists.

- The feedback collected guided the identification of core features and user preferences, which were reflected in the system design phase

3.3 FUNCTIONAL REQUIREMENTS

FR-ID	Description	Stakeholders
FR1	Users shall be able to add, edit, and delete subjects/courses.	All stakeholders
FR2	Users shall be able to create and manage daily to-do tasks.	All stakeholders
FR3	Users shall be able to launch and control a Pomodoro-style study timer.	All stakeholders
FR4	The system shall track the number of hours studied per subject.	All stakeholders
FR5	The system shall display simple analytics and progress reports.	All stakeholders
FR6	The system shall provide an intuitive and responsive user interface	All stakeholders

3.4 NON-FUNCTIONAL REQUIREMENTS

These requirements describe the quality aspects of the system:

- **Portability/Accessibility**: The system shall be accessible via mobile and desktop platforms
- **Performance**: The system shall respond to user input in less than 1 second.
- **Internationalization/Usability**: The system shall support both English and Arabic (optional).
- **Usability**: The design shall follow UI/UX best practices based on HCI principles.
- **User experience and maintainability**: The system shall maintain consistency in layout and navigation across all screens.

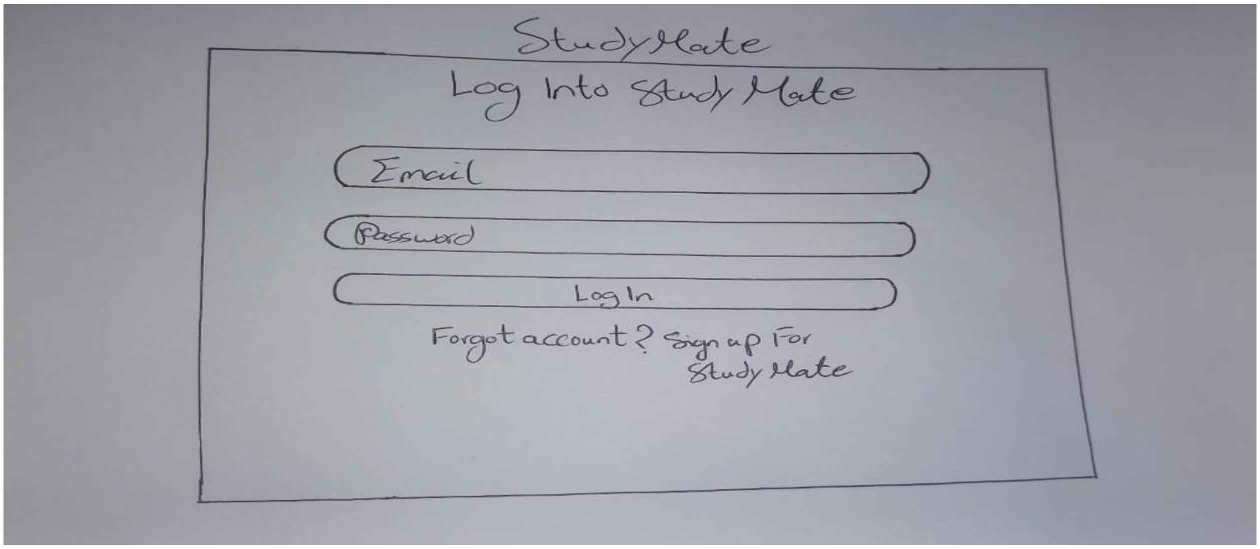
3.5 USABILITY AND USER EXPERIENCE GOALS

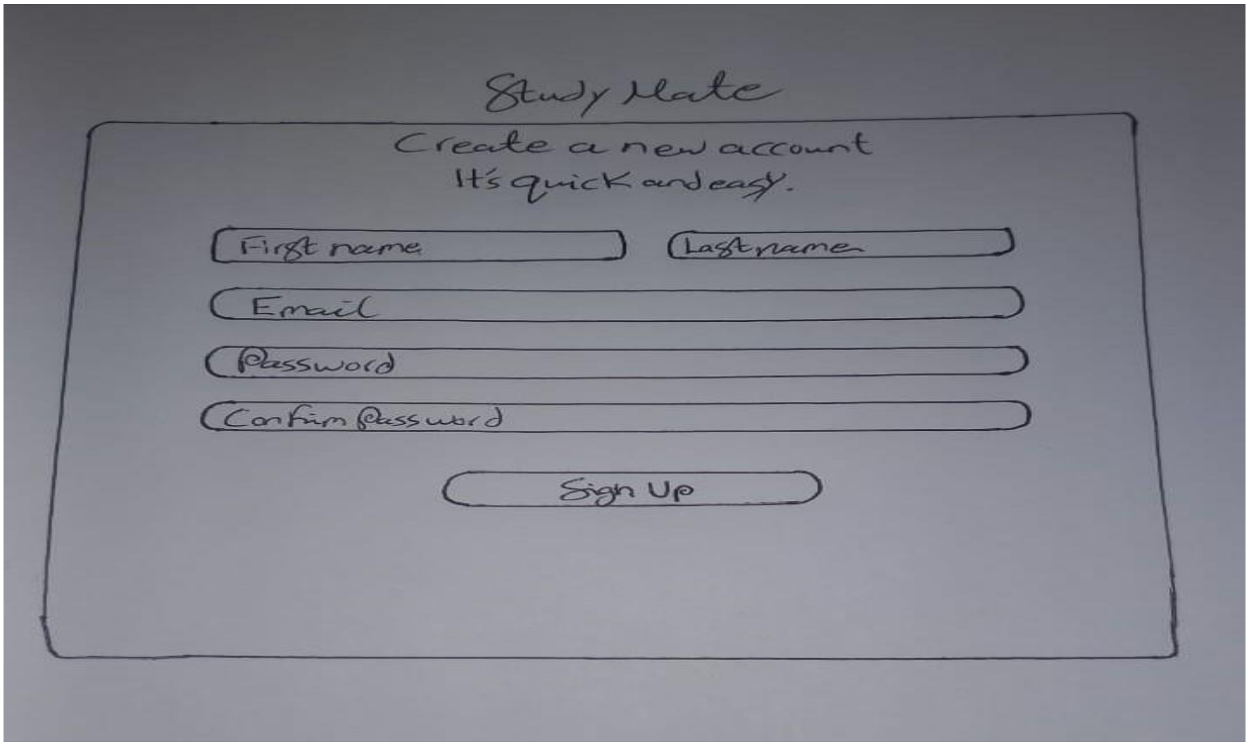
The system is designed with the following usability and UX goals:

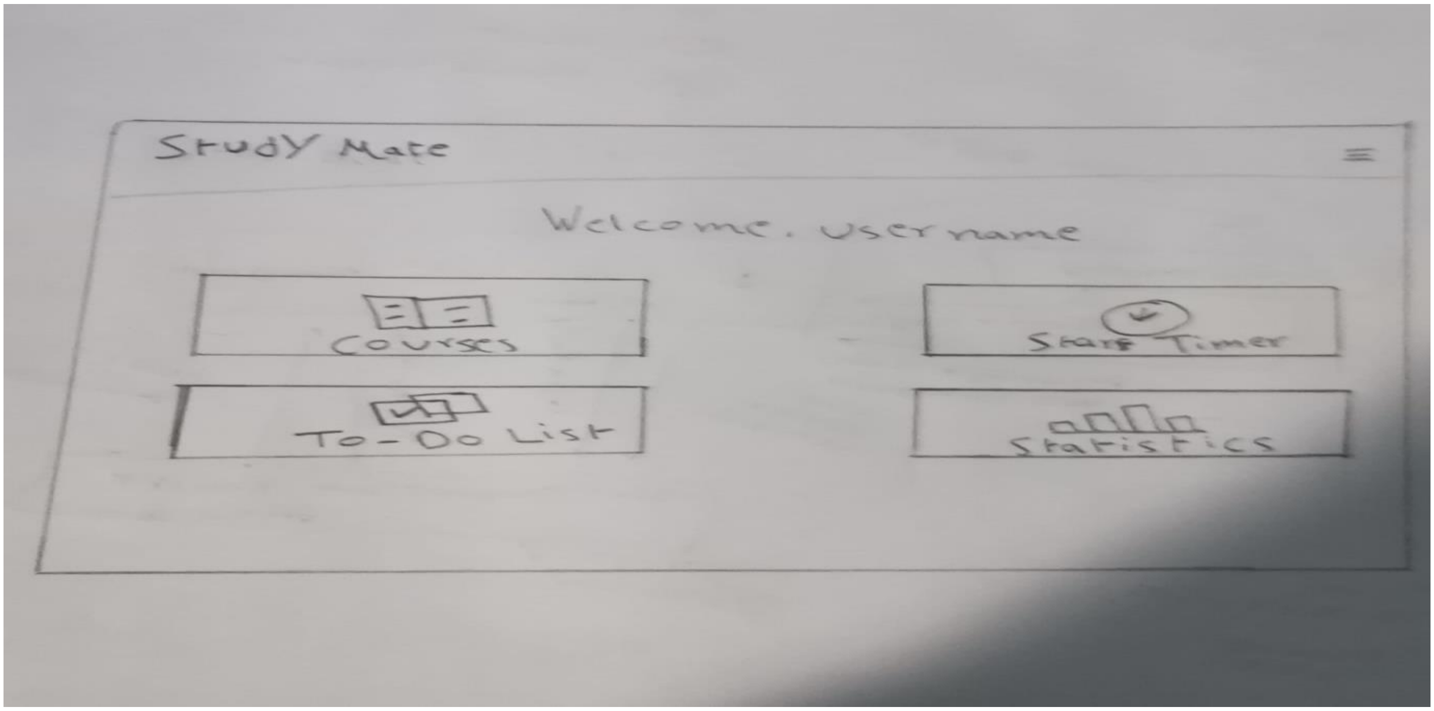
- **Simplicity**: Interfaces should be clean and avoid clutter.
- **Learnability**: New users should be able to understand the interface within minutes.
- **Efficiency**: Users should accomplish their tasks with minimal steps.
- **Feedback**: System responses (e.g., timer notifications) should be clear and timely.
- **Engagement**: The design should encourage continued usage through positive interaction and progress visibility

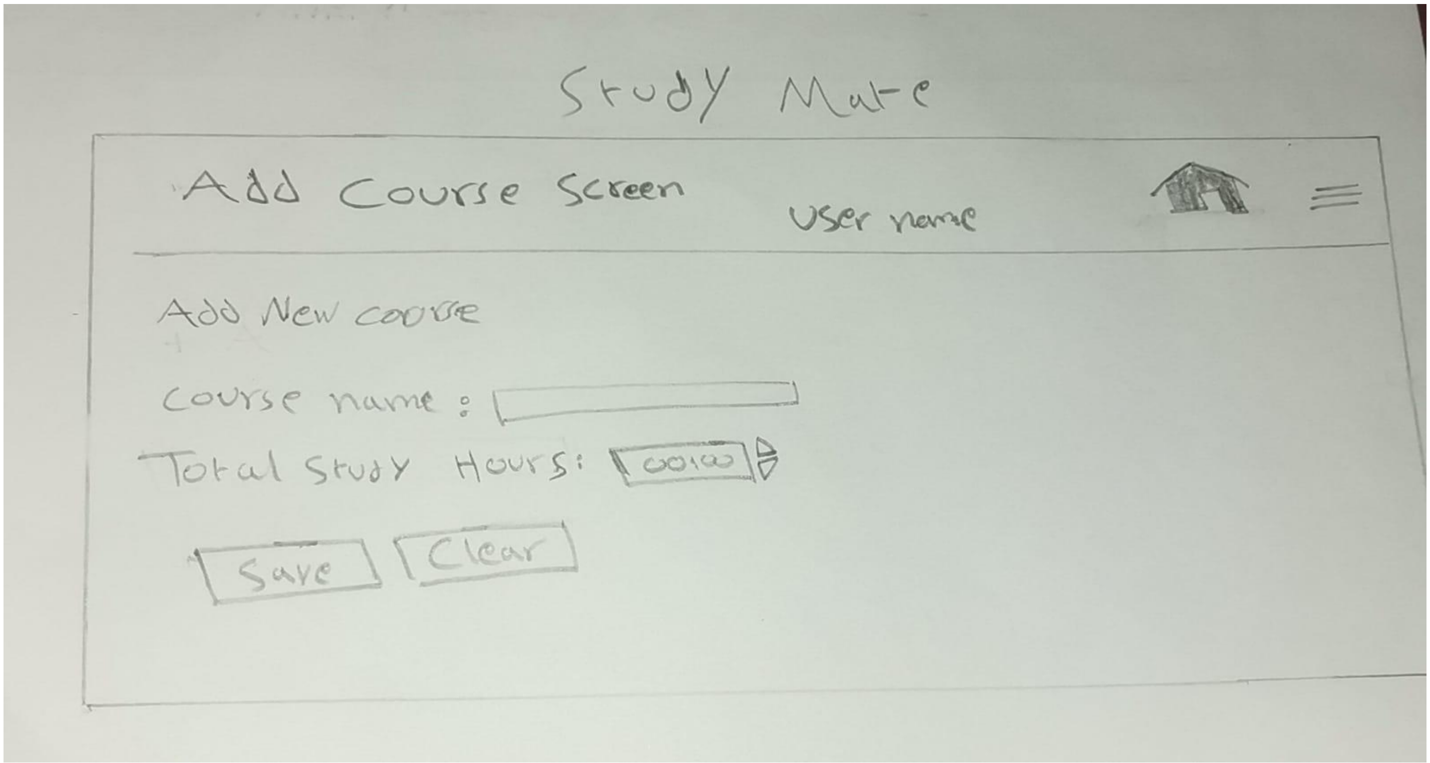
4.0 SYSTEM DESIGN DOCUMENT

4.1 Low-Fidelity Prototyping

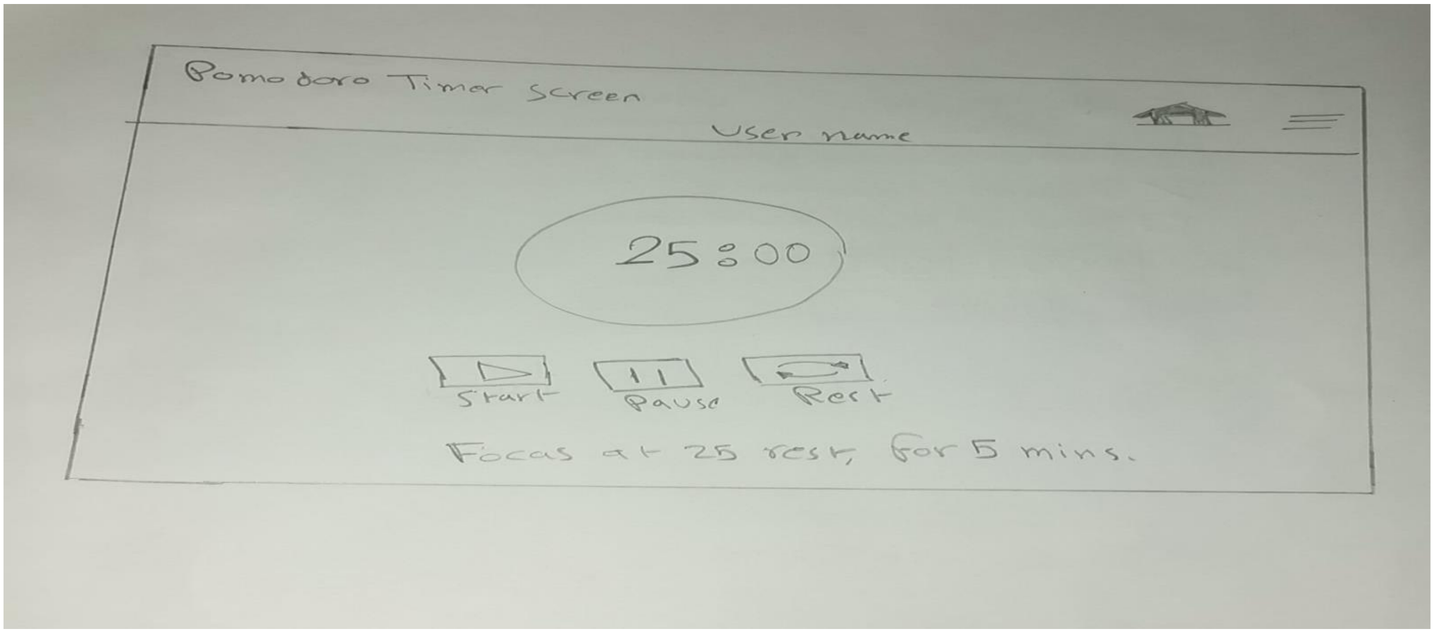


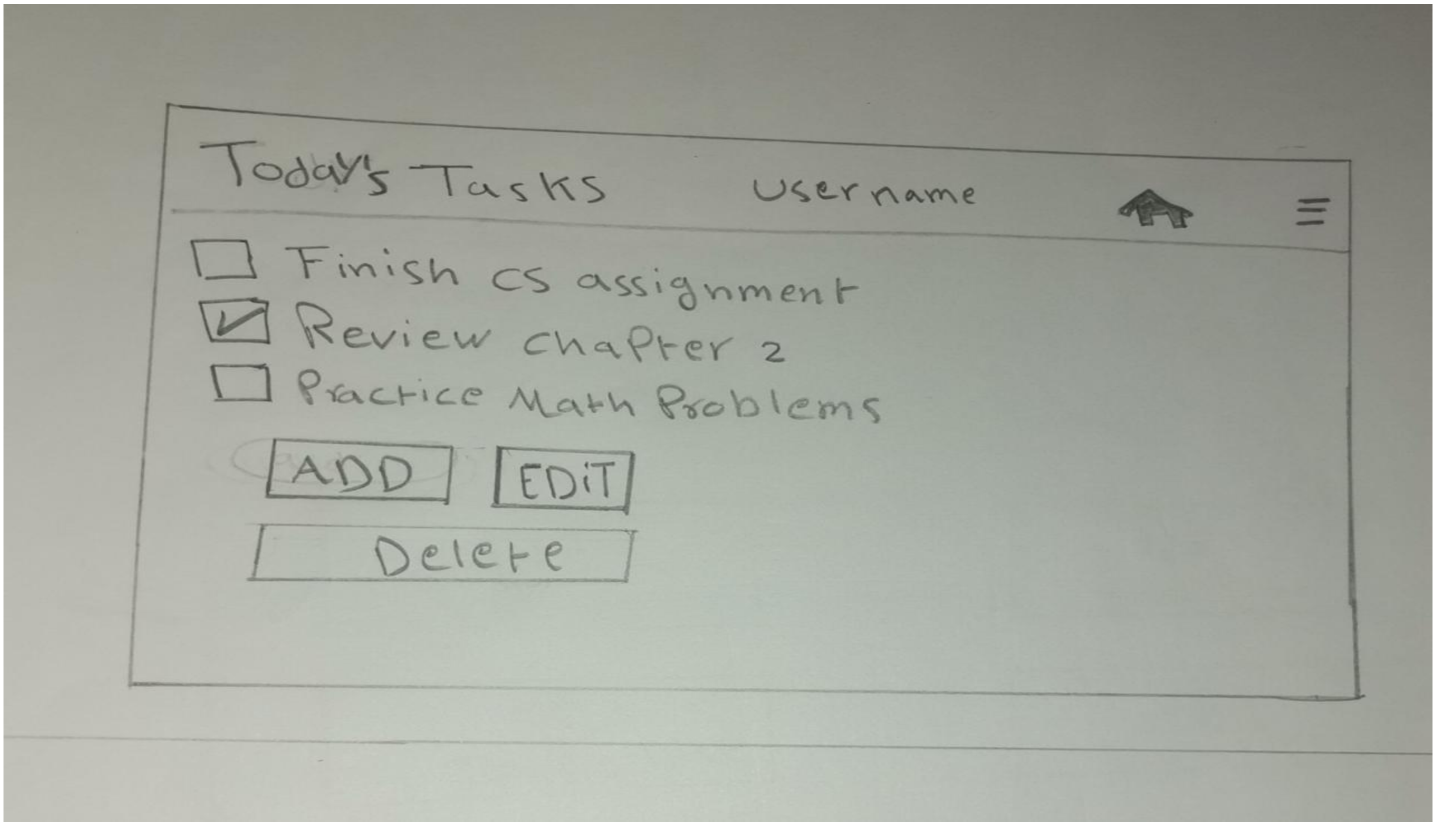


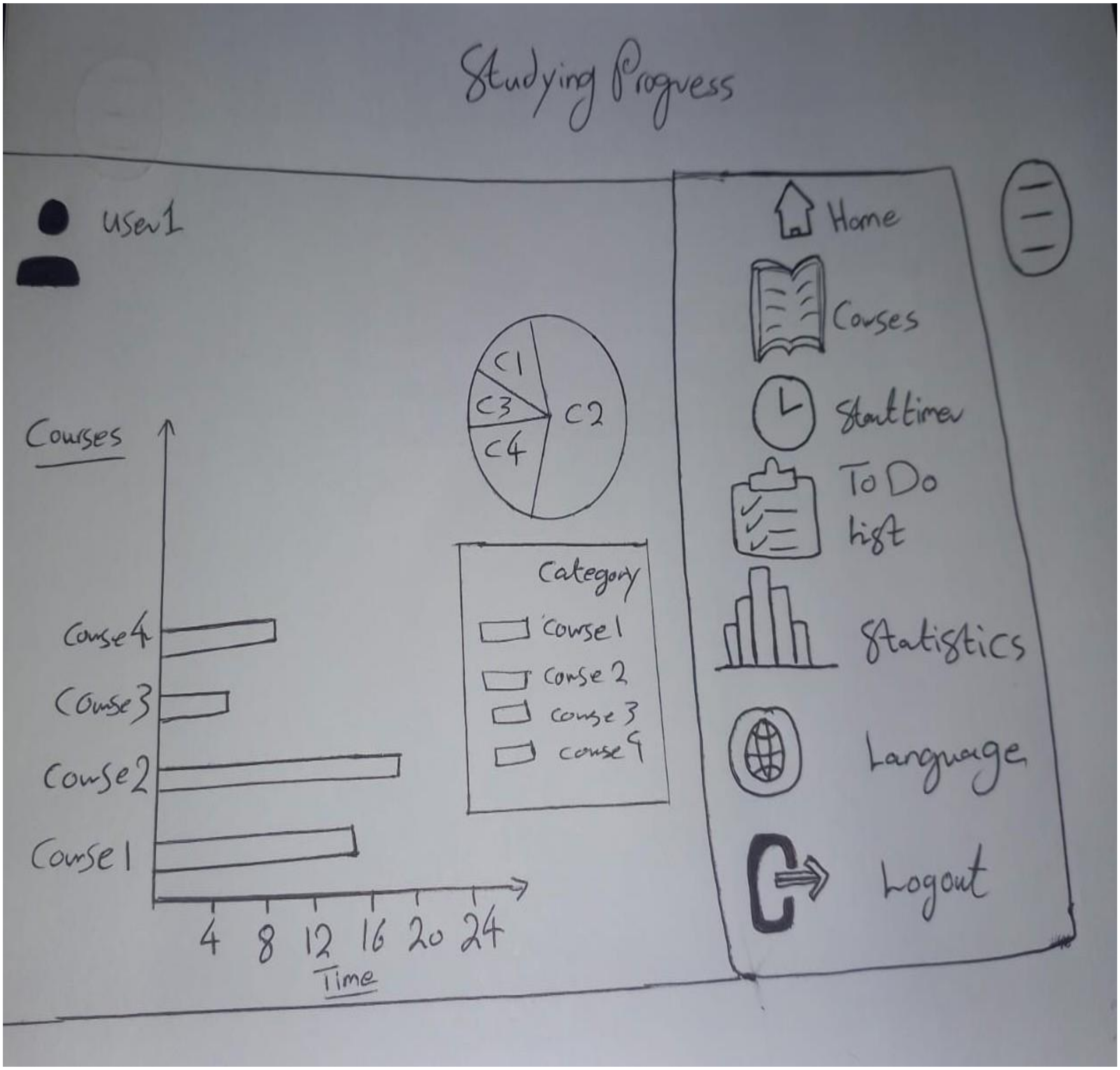

Study Mate
≡
Welcome, User name
Courses
Start Timer
To-Do List
Statistics


Study Mate
Add Course Screen
User name
≡
Add New course
Course name :
Total Study Hours: 00:00
Save
Clear

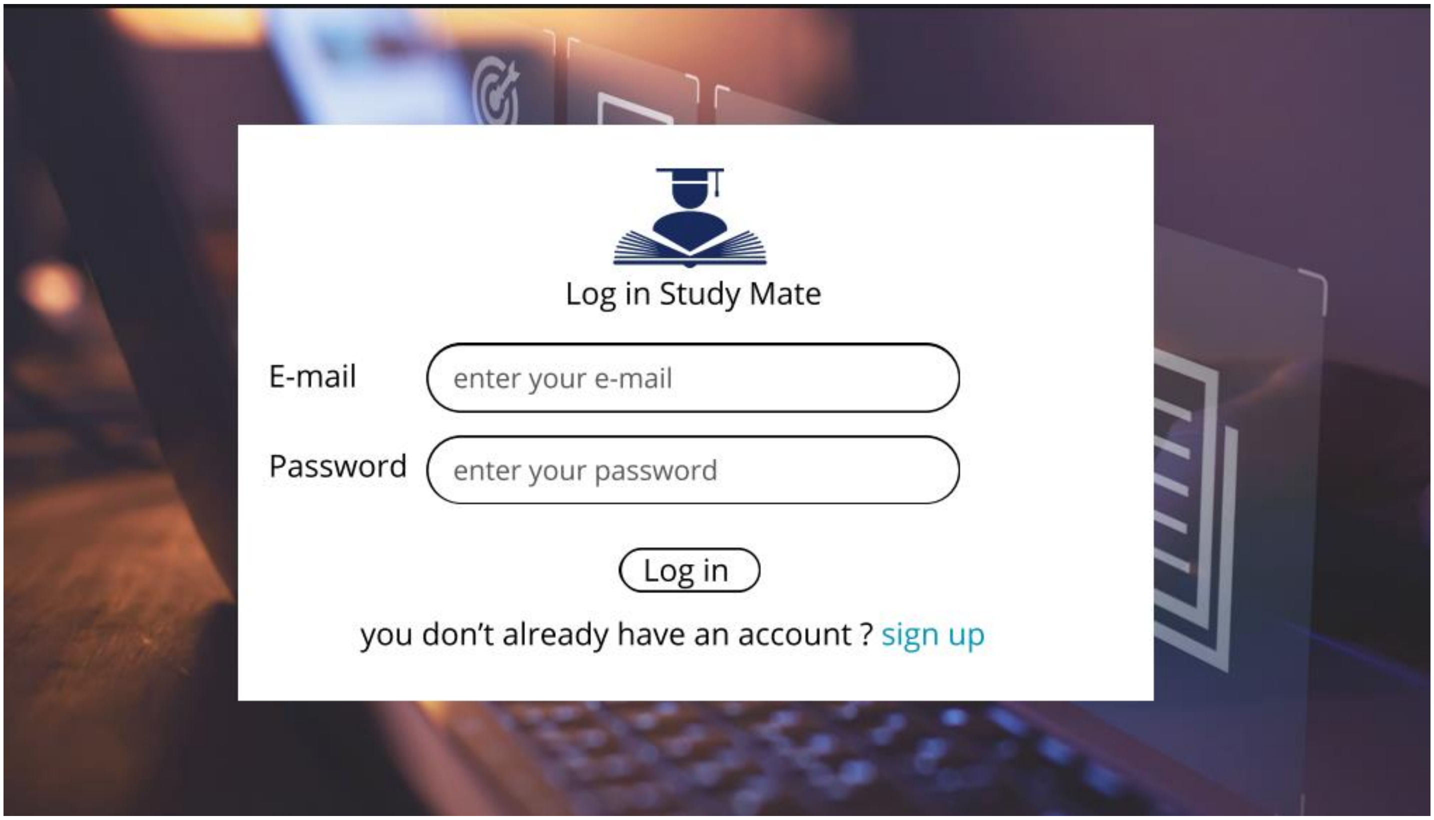
.


Pomodoro Timer Screen
User name
25:00
Start
Pause
Rest
Focus at 25 rest, for 5 mins.


Today's Tasks
Username
□ Finish CS assignment
☑ Review chapter 2
□ Practice Math Problems
ADD
EDIT
Delete

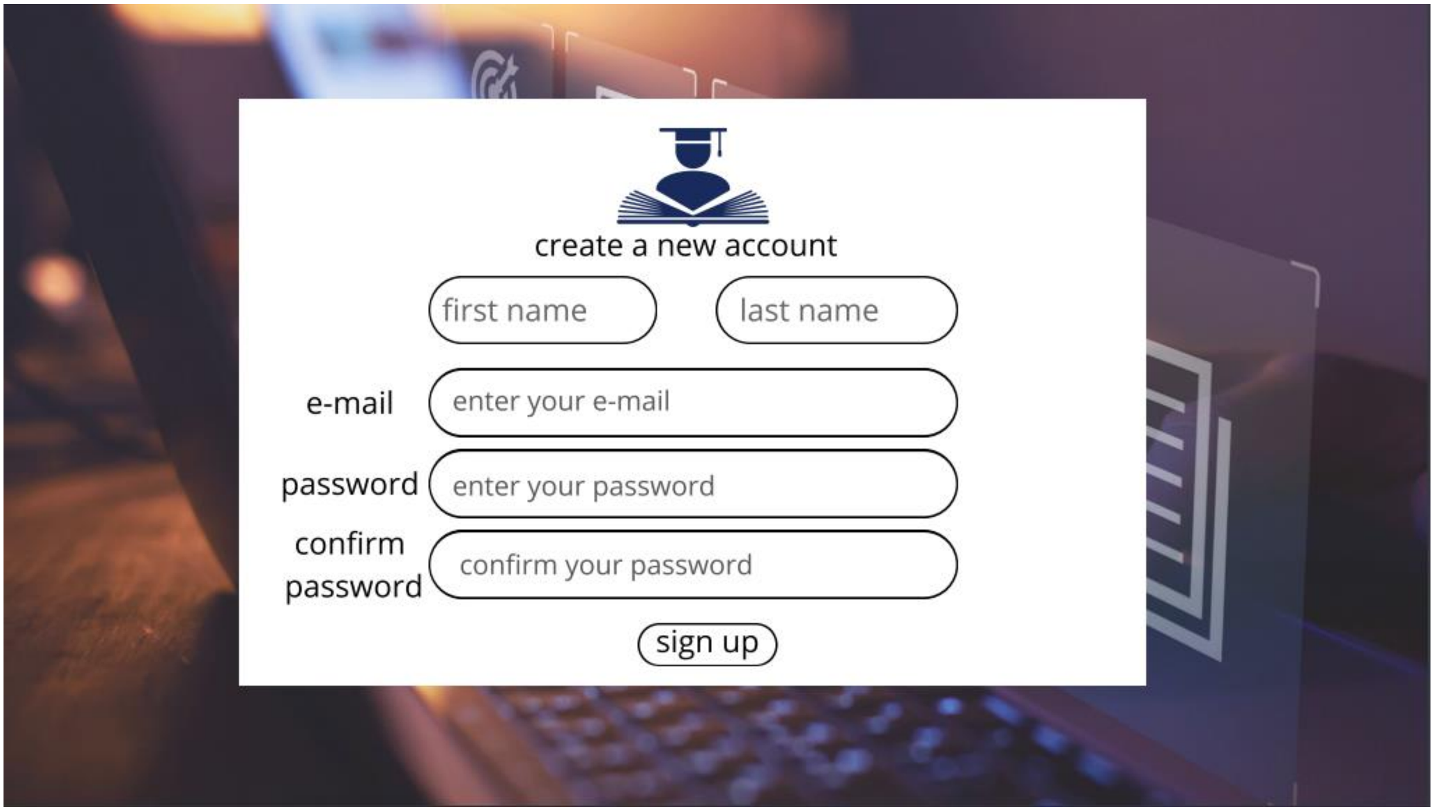


4.2 MEDIUM-FIDELITY PROTOTYPING

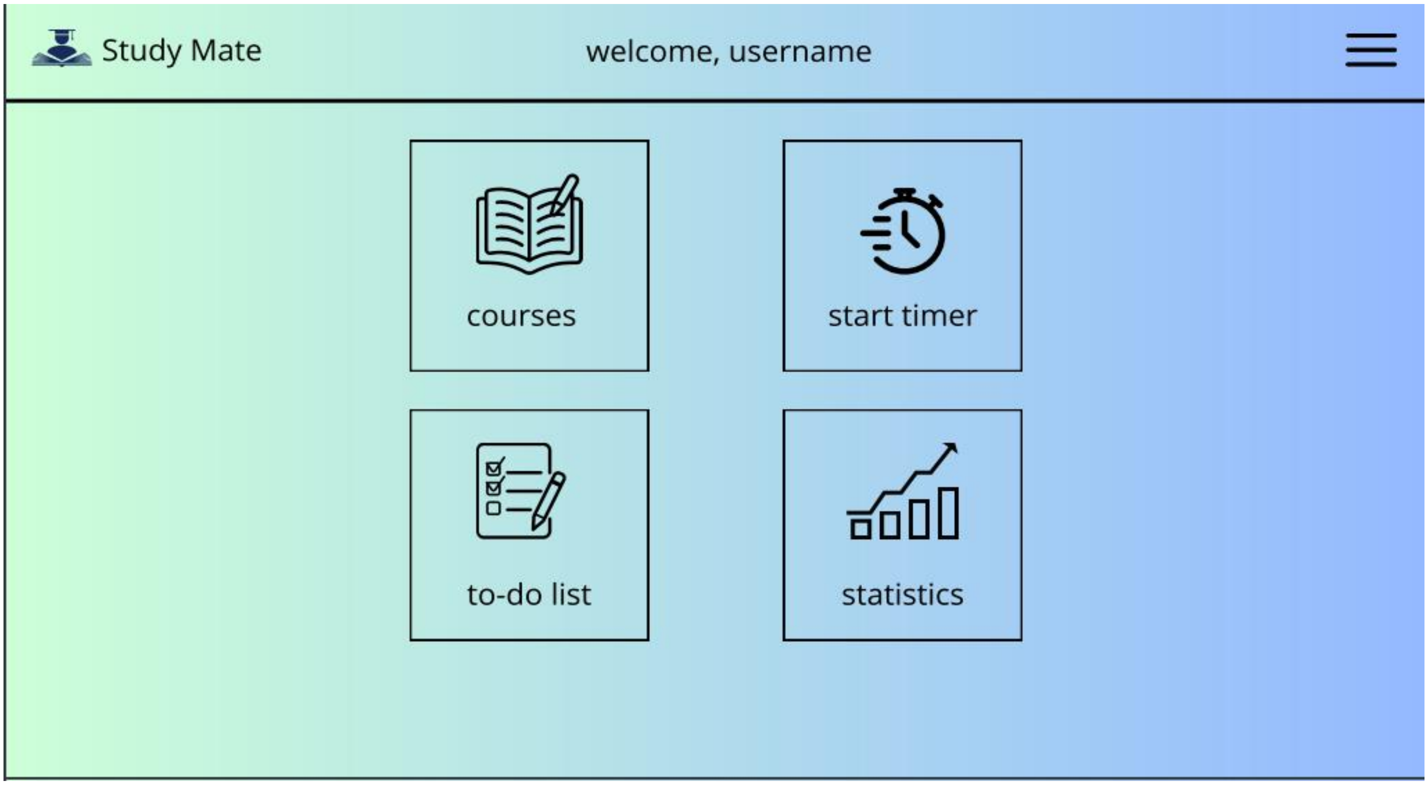


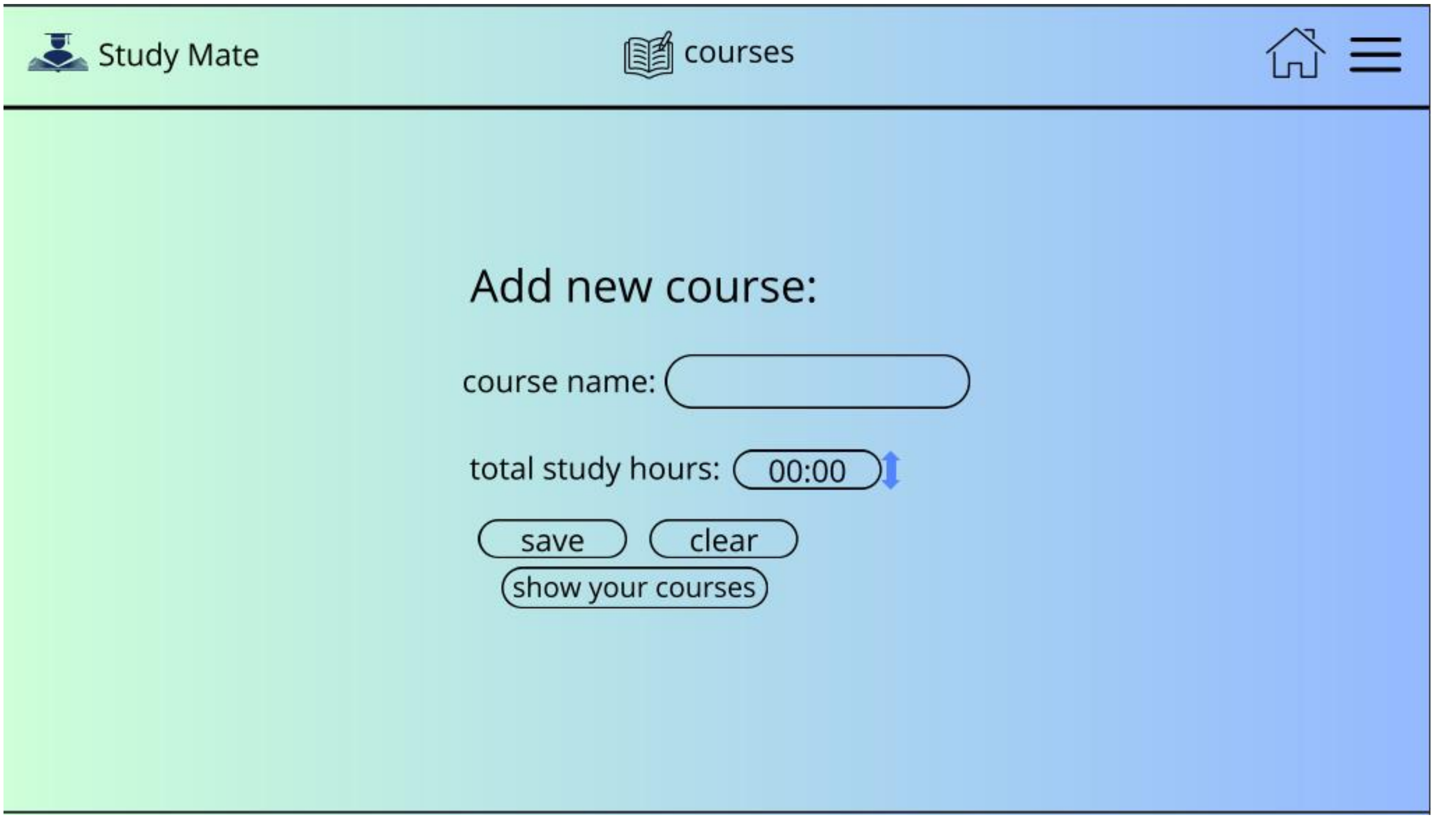
Log in Study Mate

you don't already have an account ? sign up



create a new account

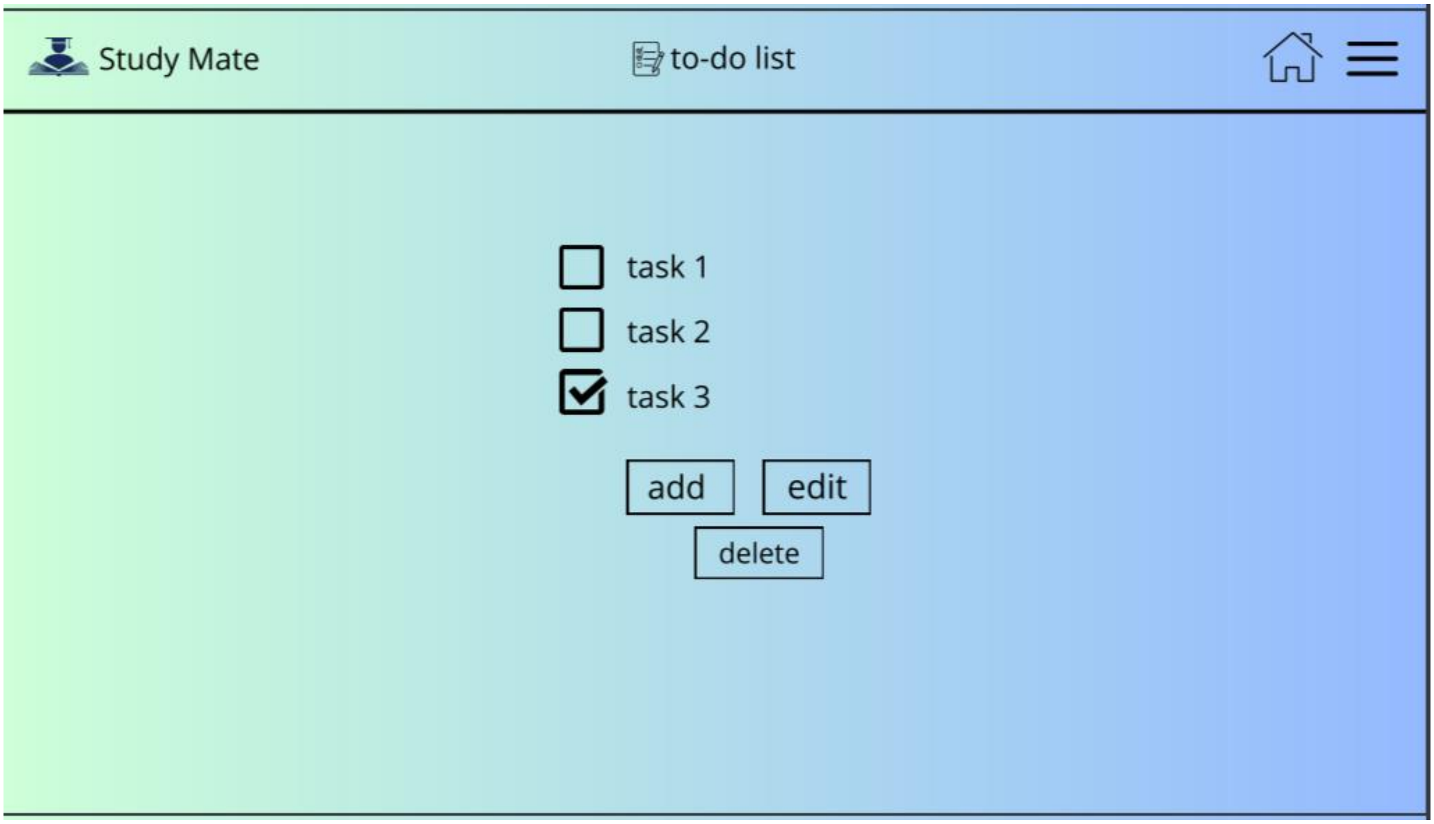

Study Mate
welcome, username
courses
start timer
to-do list
statistics

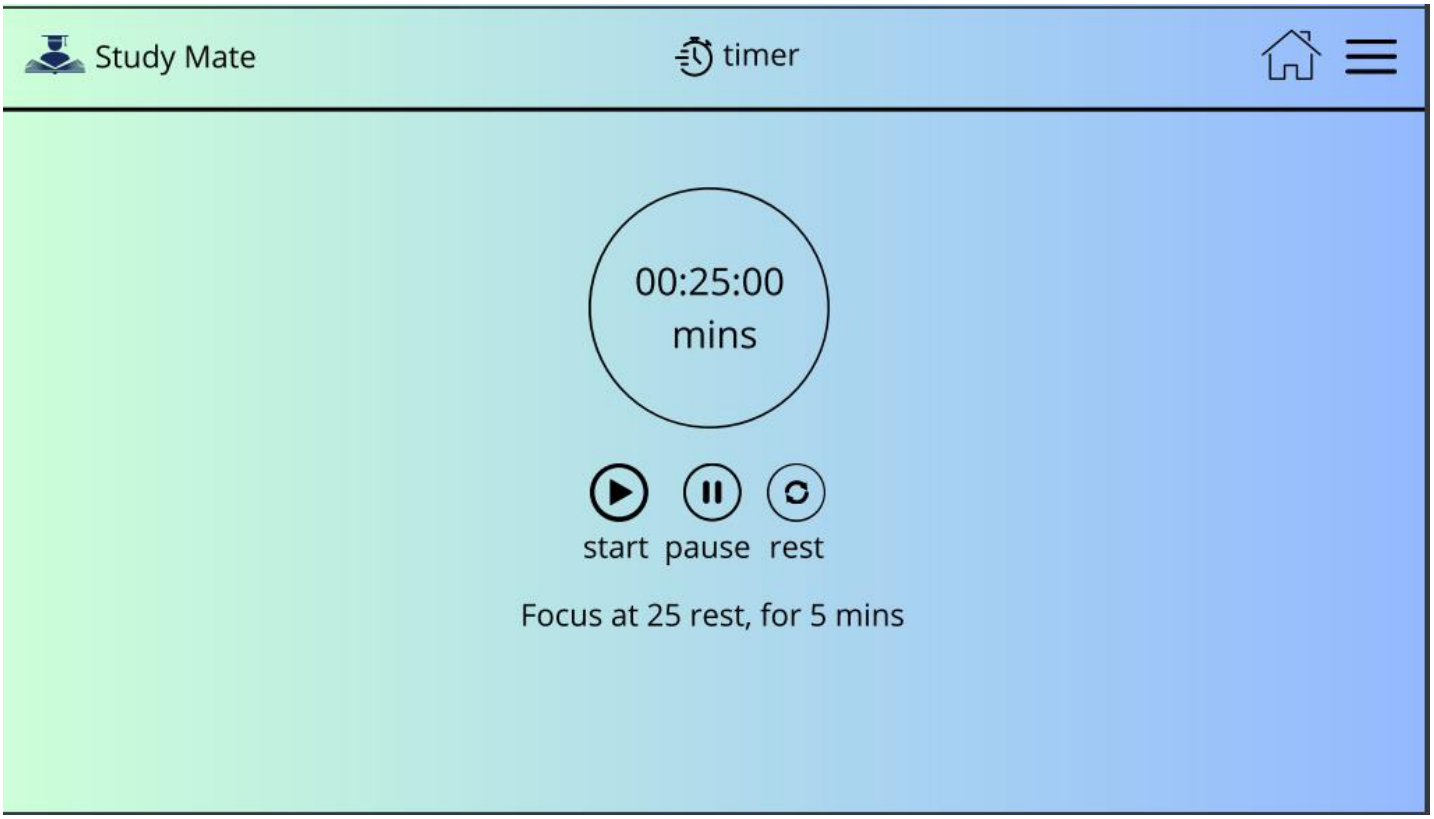


Add new course:

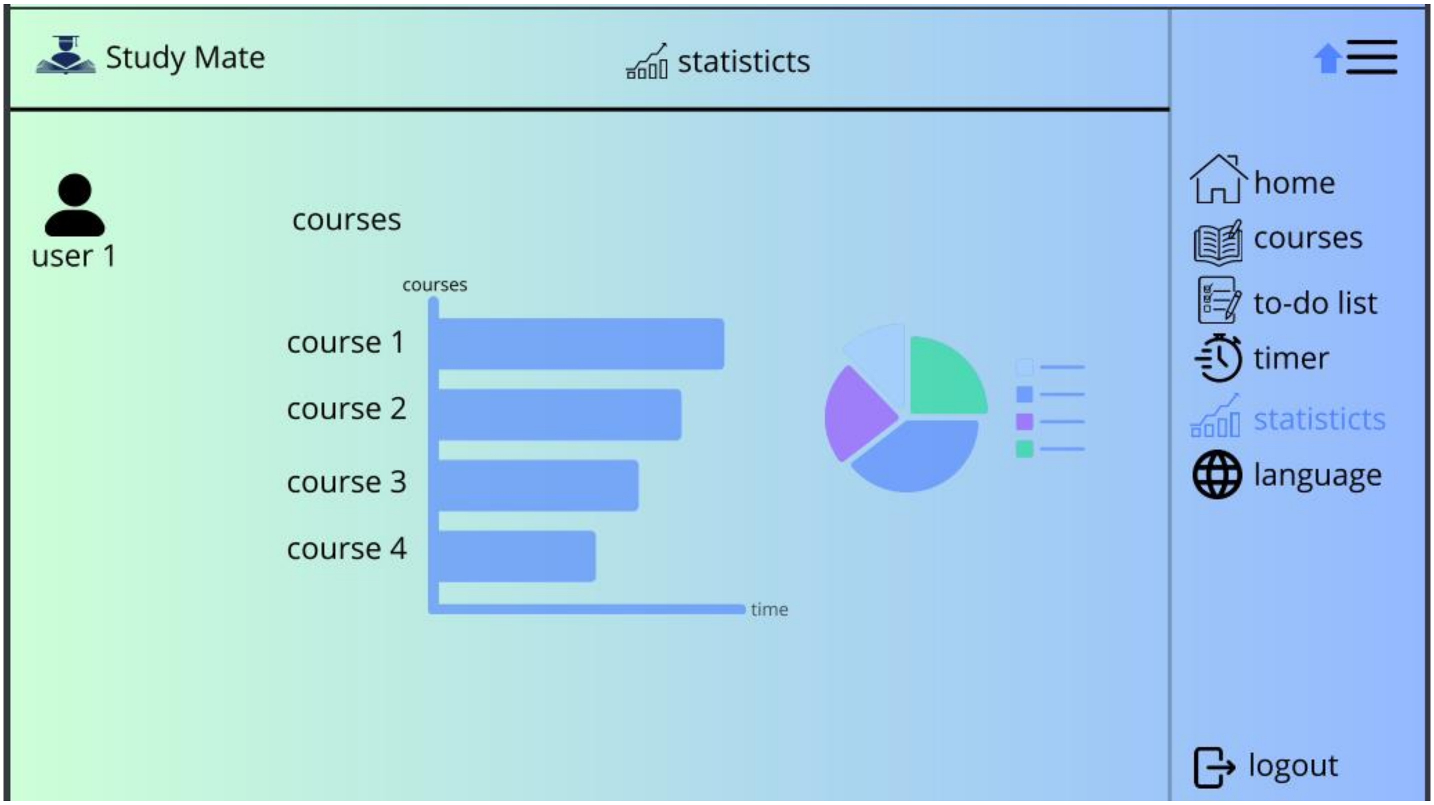
course name:

total study hours: 00:00

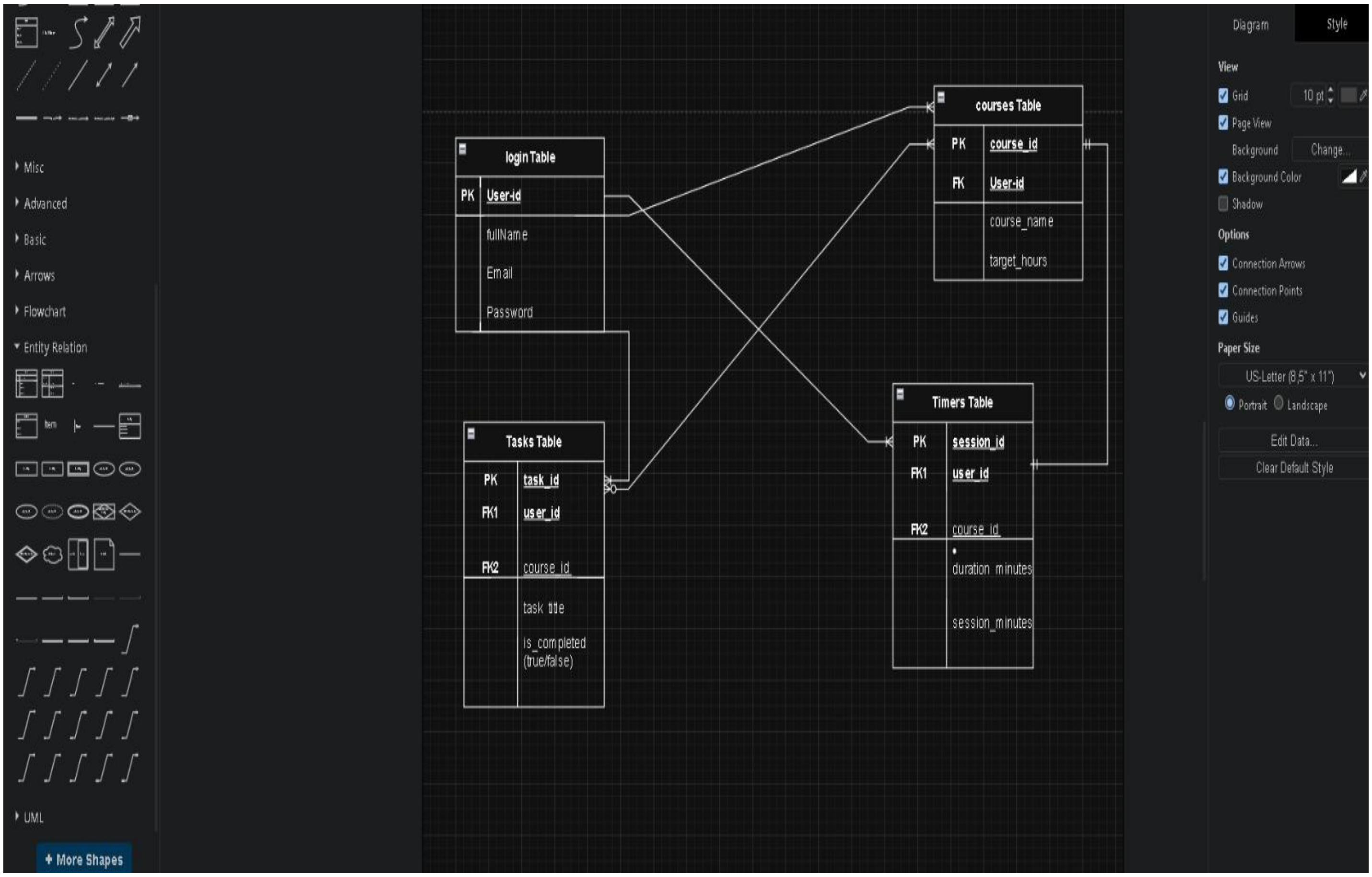




Focus at 25 rest, for 5 mins


Study Mate
statistícts
user 1
courses
courses
course 1
course 2
course 3
course 4
time
home
courses
to-do list
timer
statistícts
language
logout

4.3 Database design



login Table

PK	User-id
	fullName
	Email
	Password

courses Table

PK	course_id
FK	User-id
	course_name
	target_hours

Tasks Table

PK	task_id
FK1	user_id
FK2	course_id
	task title
	is_completed (true/false)

Timers Table

PK	session_id
FK1	user_id
FK2	course_id
	• duration minutes
	session_minutes

View

☑ Grid 10 pt

☑ Page View

Background Change...

☑ Background Color

□ Shadow

Options

☑ Connection Arrows

☑ Connection Points

☑ Guides

Paper Size

US-Letter (8.5" x 11")

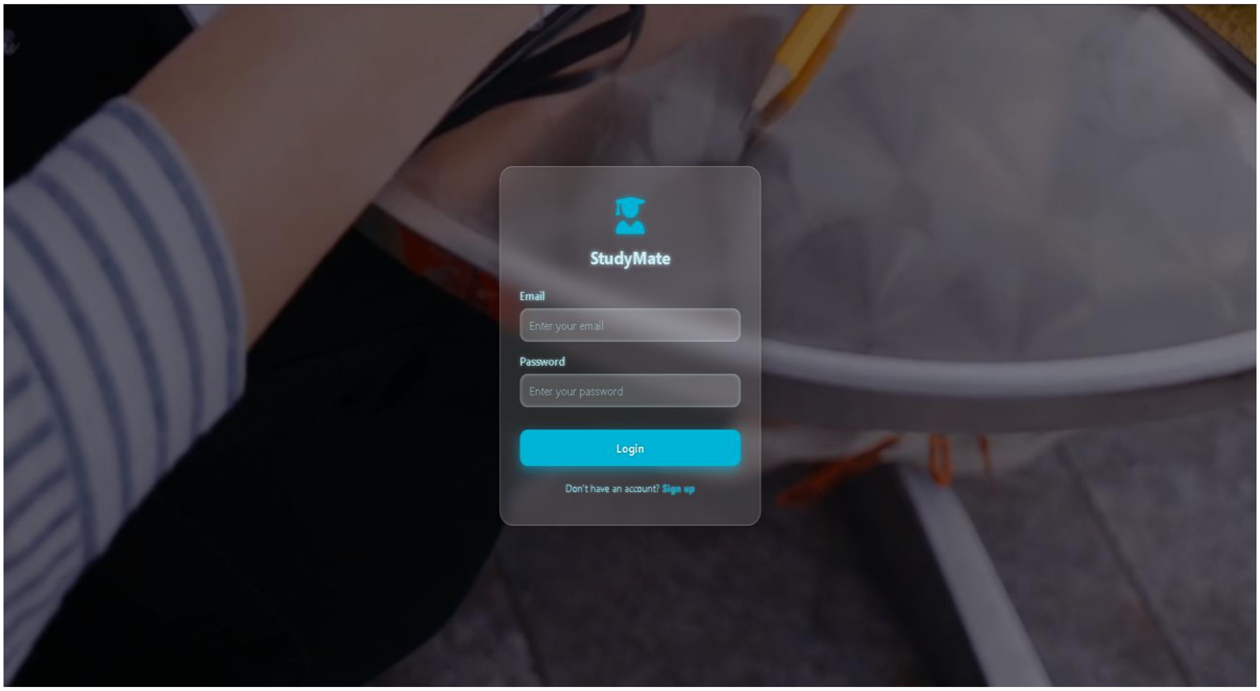
● Portrait ○ Landscape

Edit Data...

Clear Default Style

5.0 SYSTEM IMPLEMENTATION

5.1 High-Fidelity Prototyping



StudyMate

Email

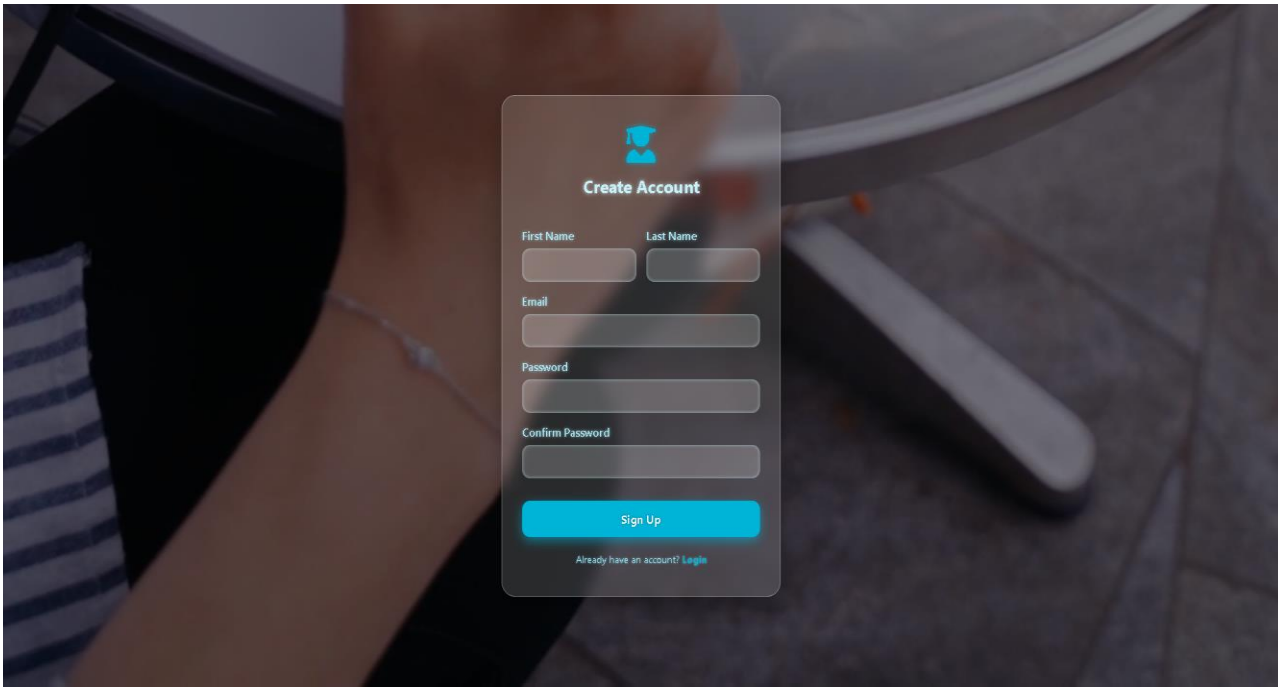
Enter your email

Password

Enter your password

Login

Don't have an account? Sign up



Create Account

First Name

Last Name

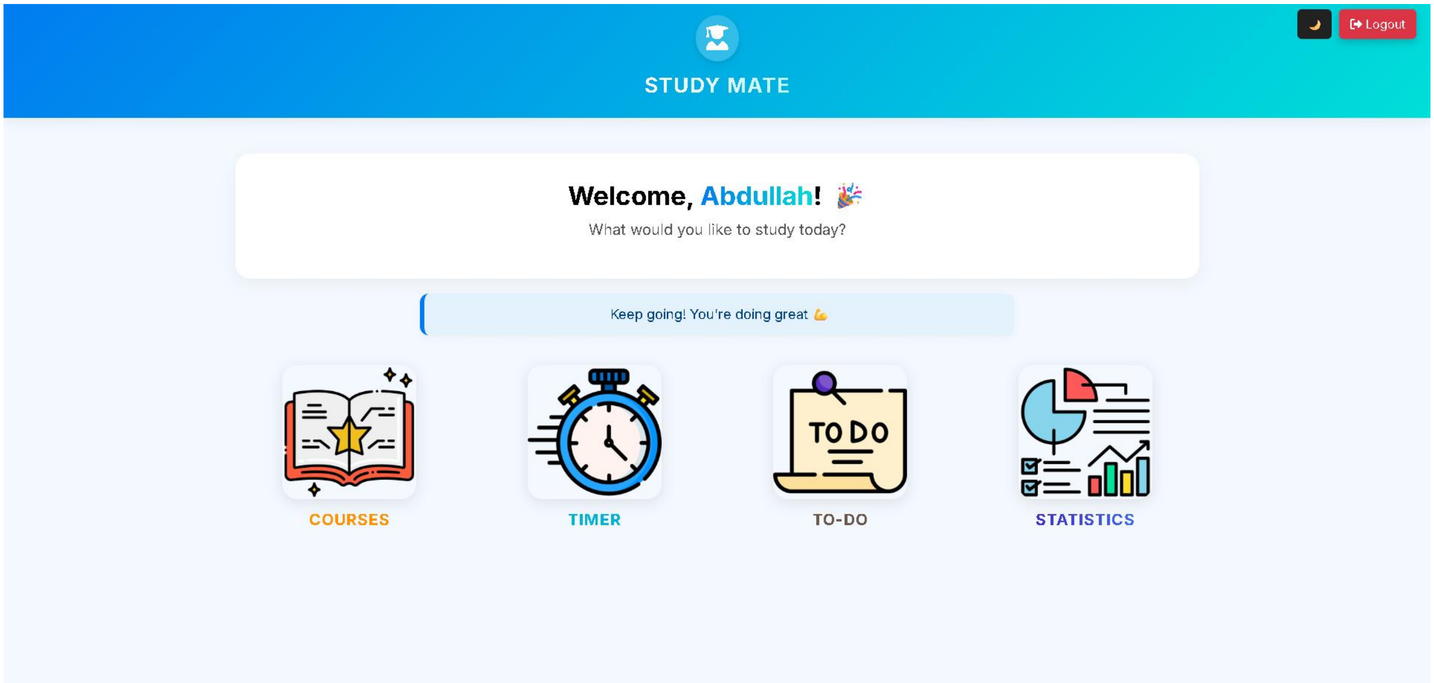
Email

Password

Confirm Password

Sign Up

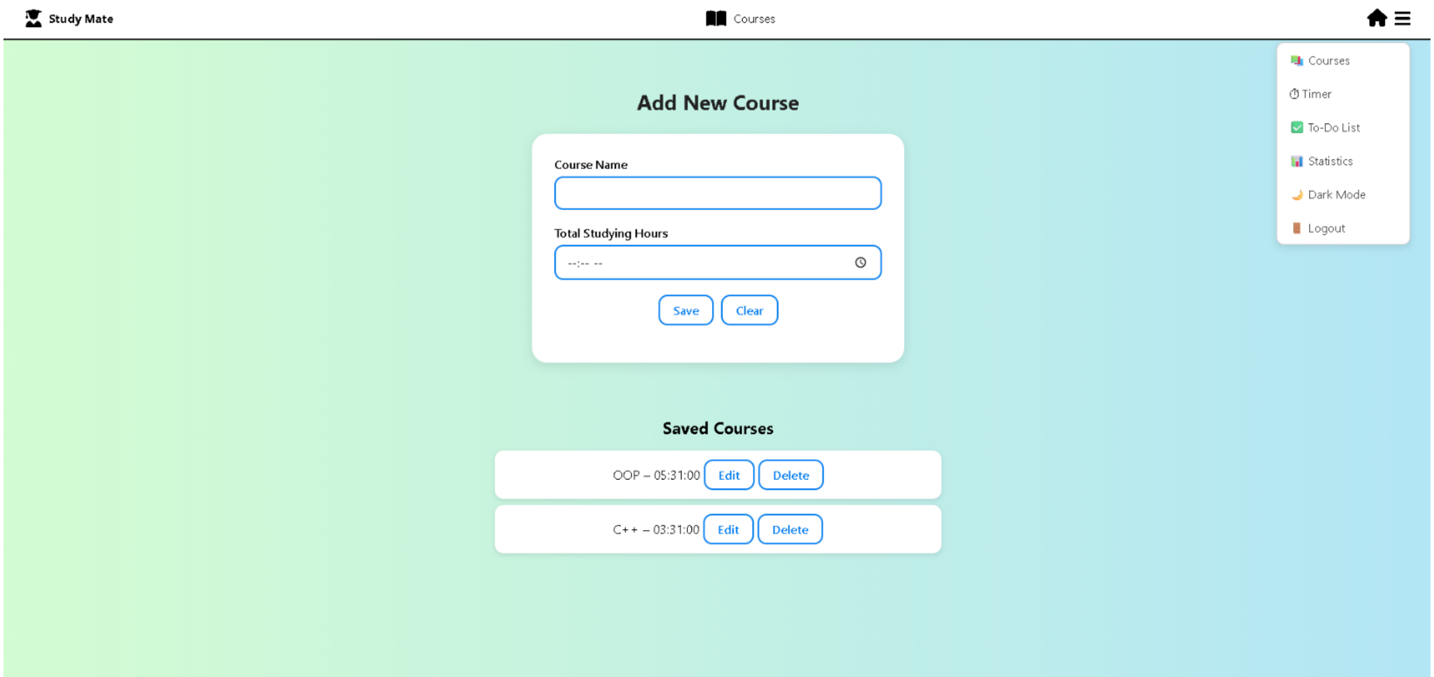
Already have an account? Login



Welcome, Abdullah! 🎉

What would you like to study today?

Keep going! You're doing great 💪



Add New Course

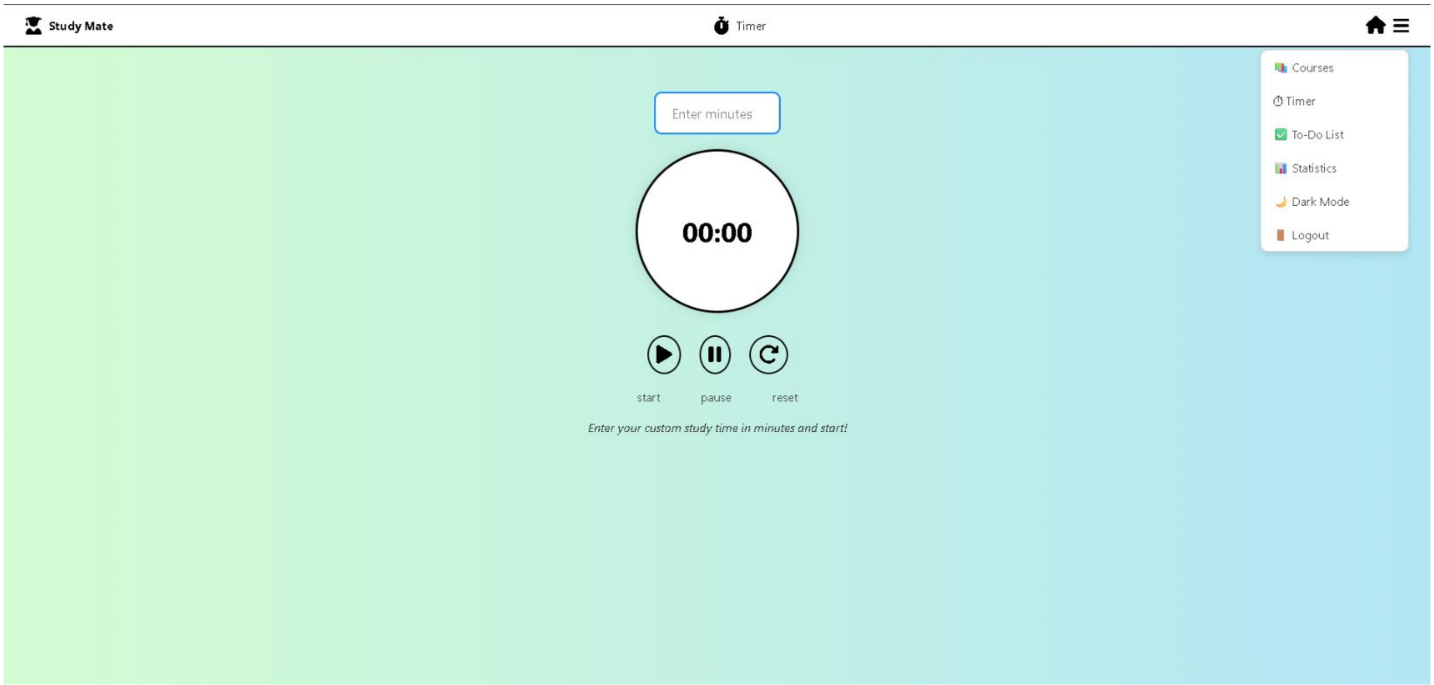
Course Name

Total Studying Hours --:-- --

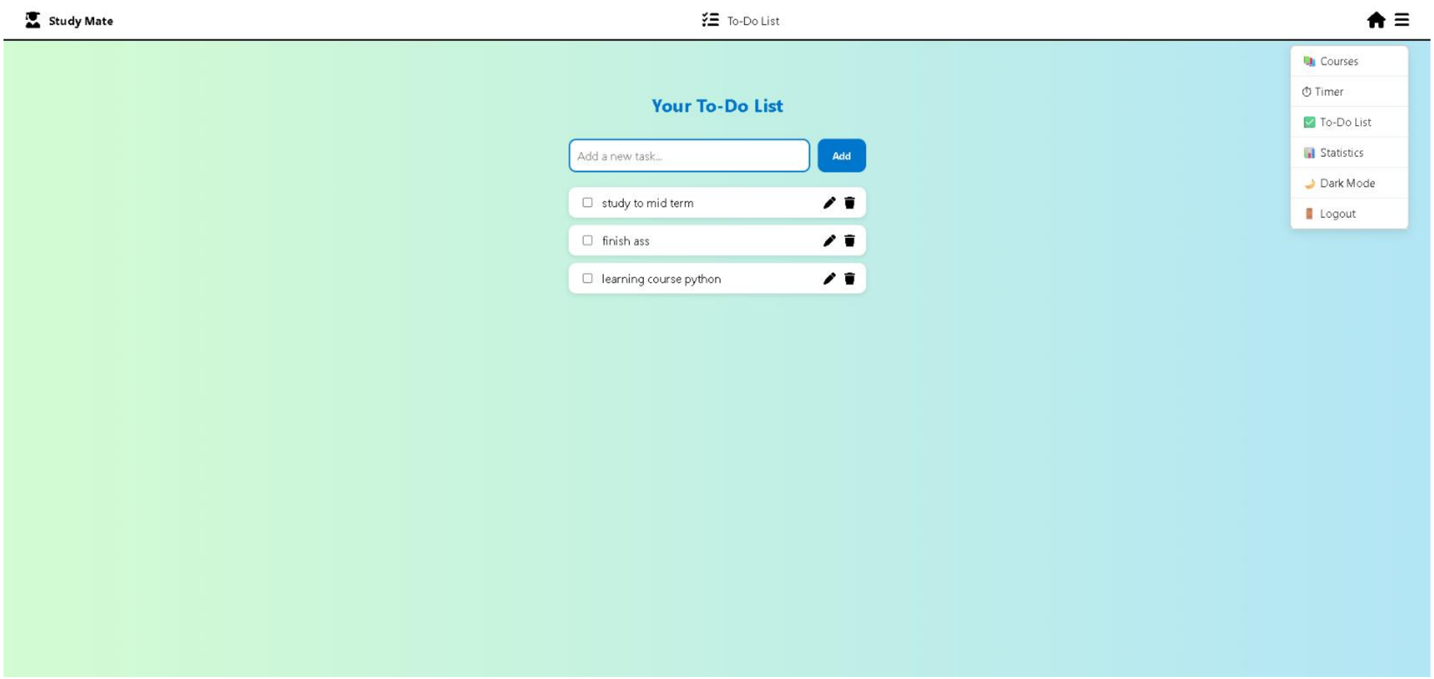
Saved Courses

OOP – 05:31:00 Edit Delete

C++ – 03:31:00 Edit Delete

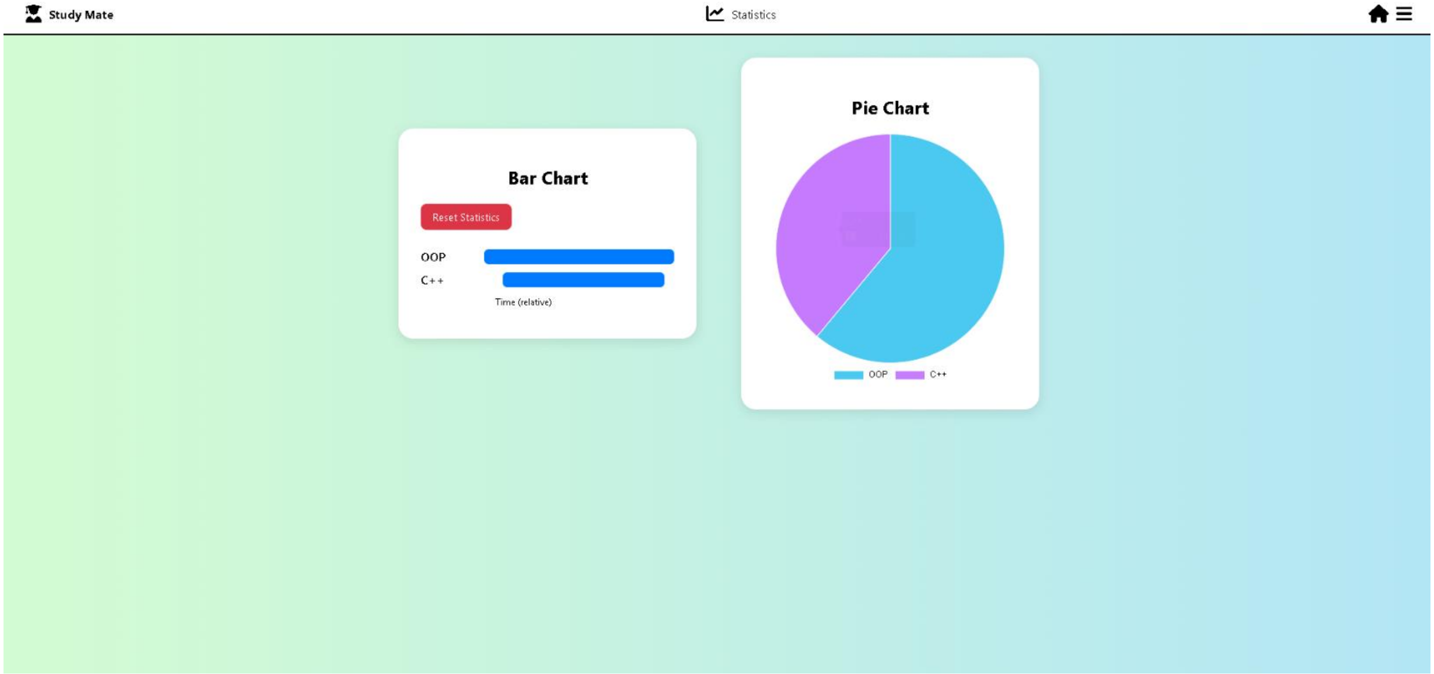


Enter your custom study time in minutes and start!

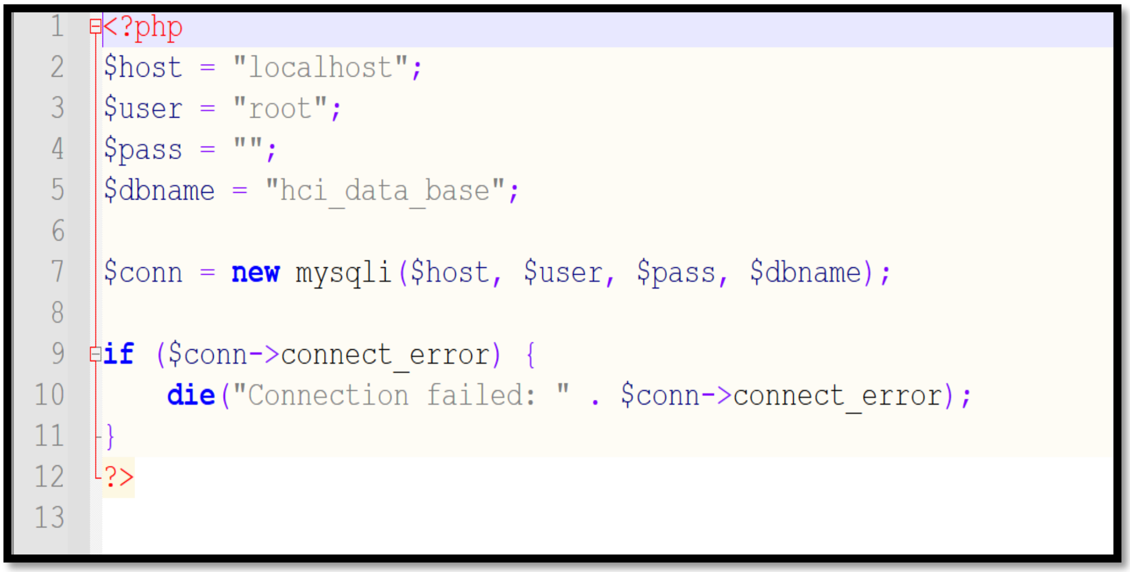


Your To-Do List

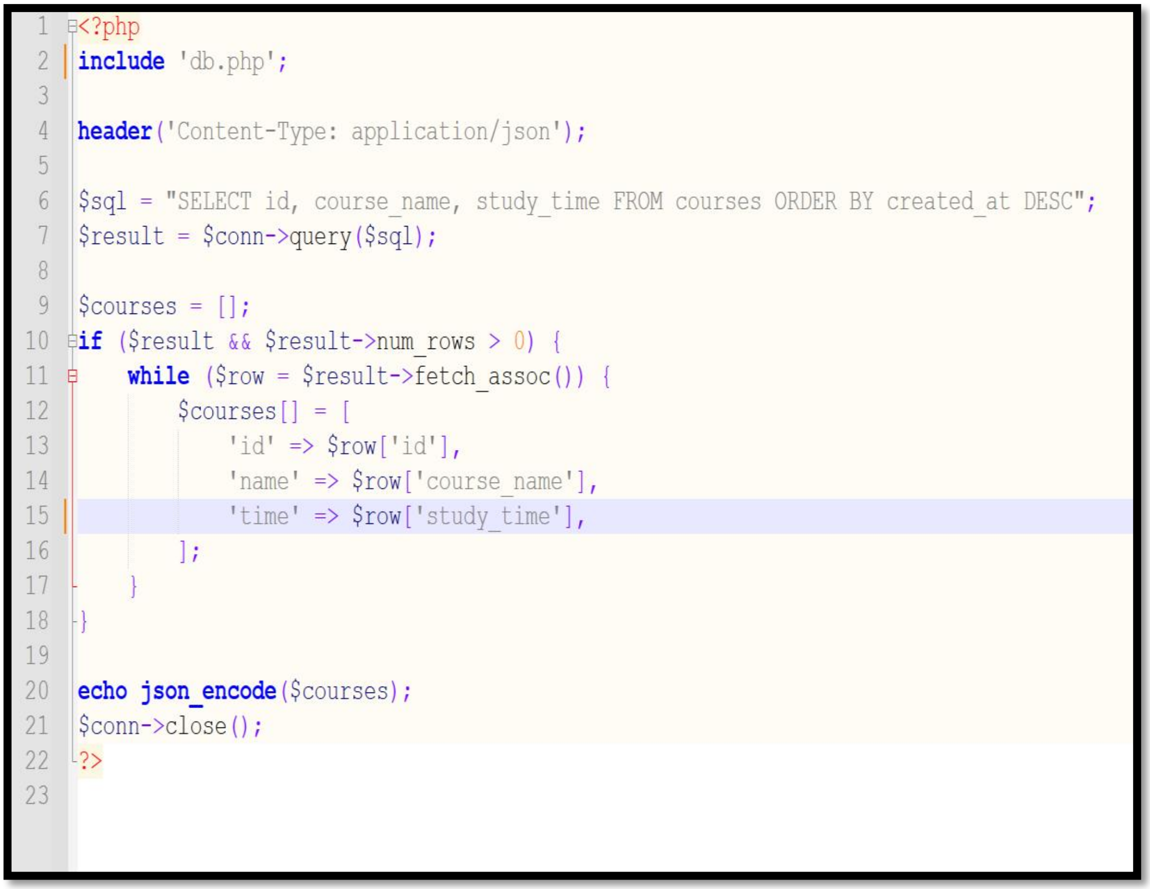
- □ study to mid term
- □ finish ass
- □ learning course python


Study Mate
Statistics
Bar Chart
Reset Statistics
OOP
C++
Time (relative)
Pie Chart
OOP
C++

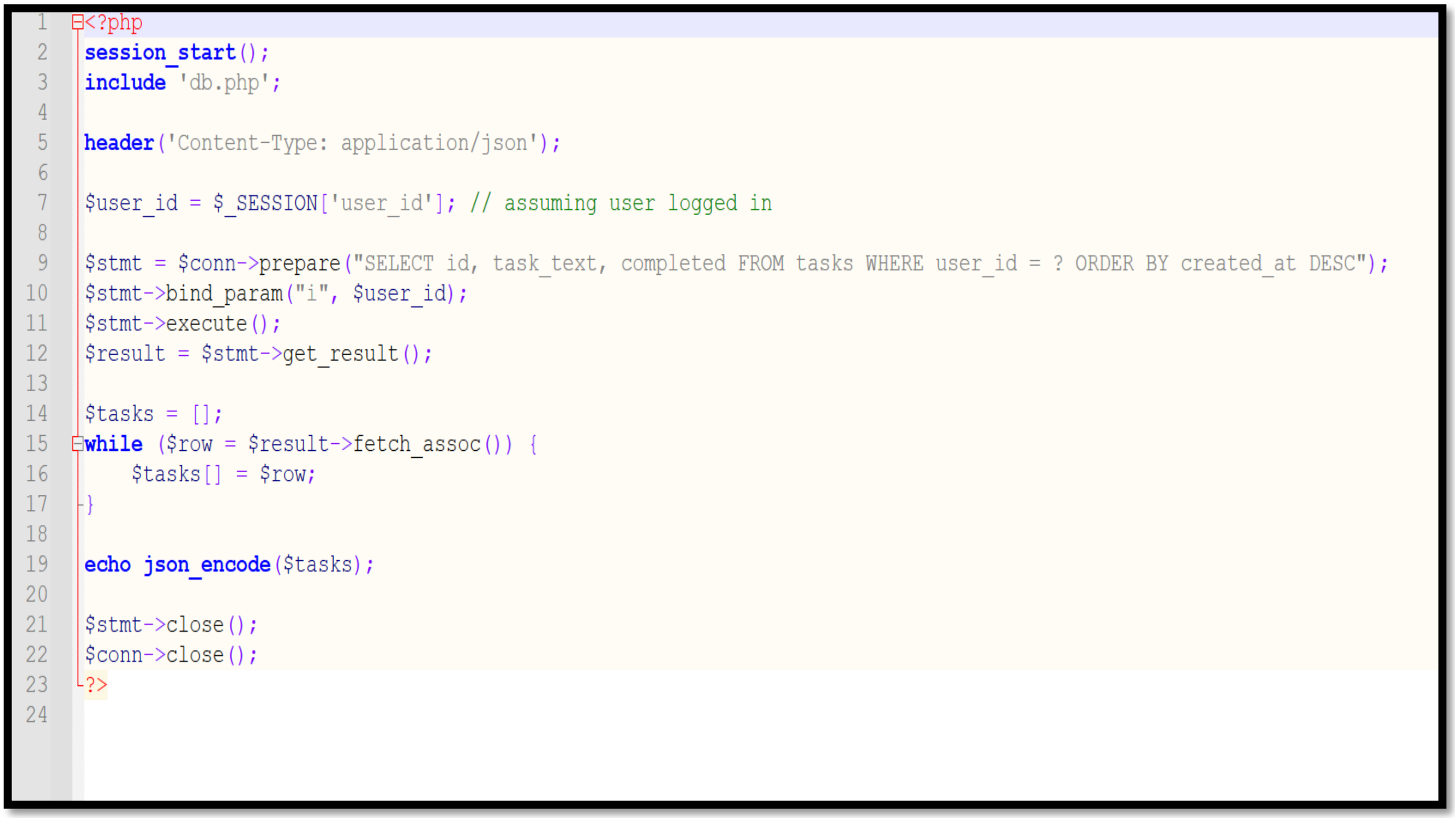
5.2 database implementation



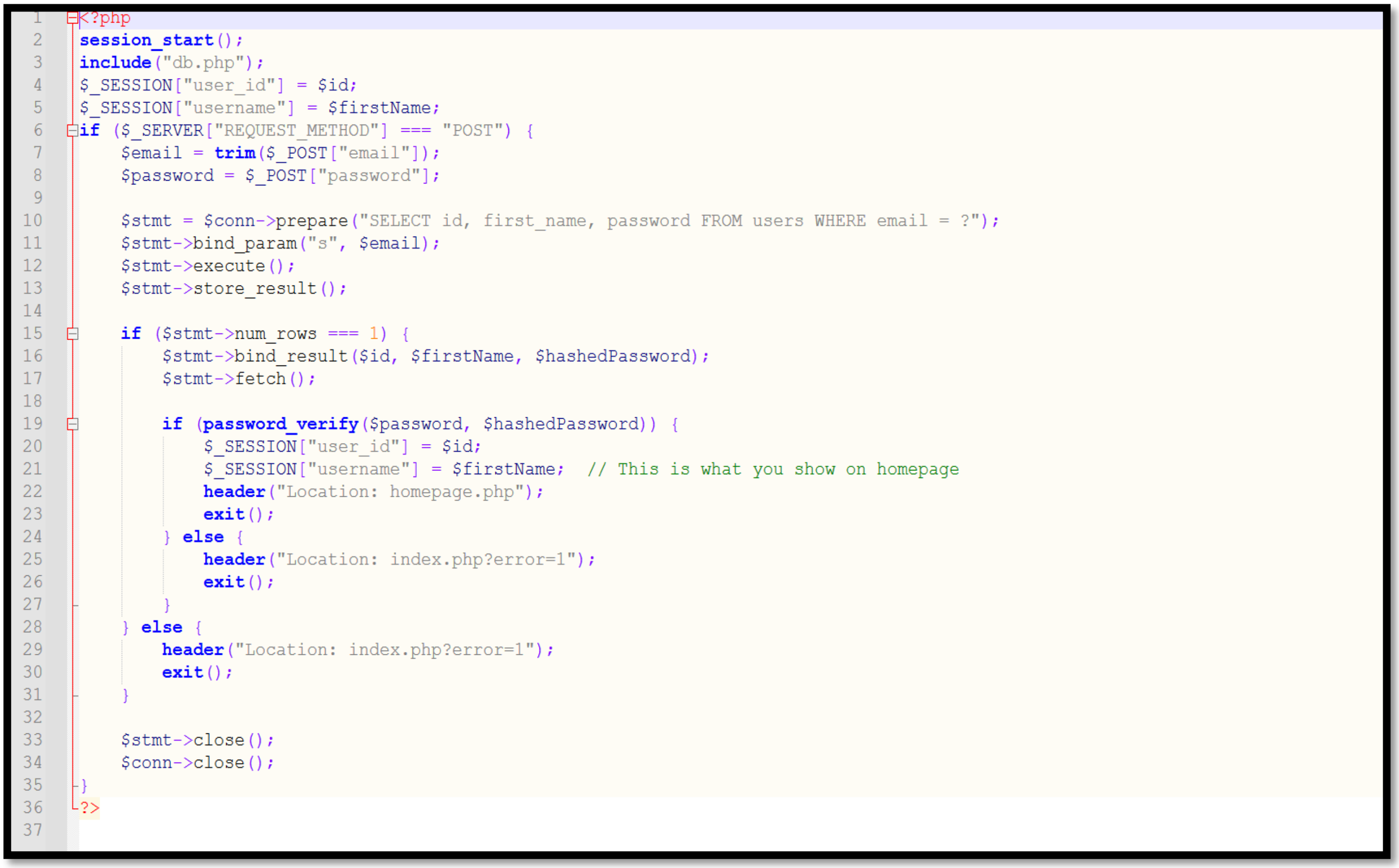
```php
<?php
$host = "localhost";
$user = "root";
$pass = "";
$dbname = "hci_data_base";

$conn = new mysqli($host, $user, $pass, $dbname);

if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
?>
```



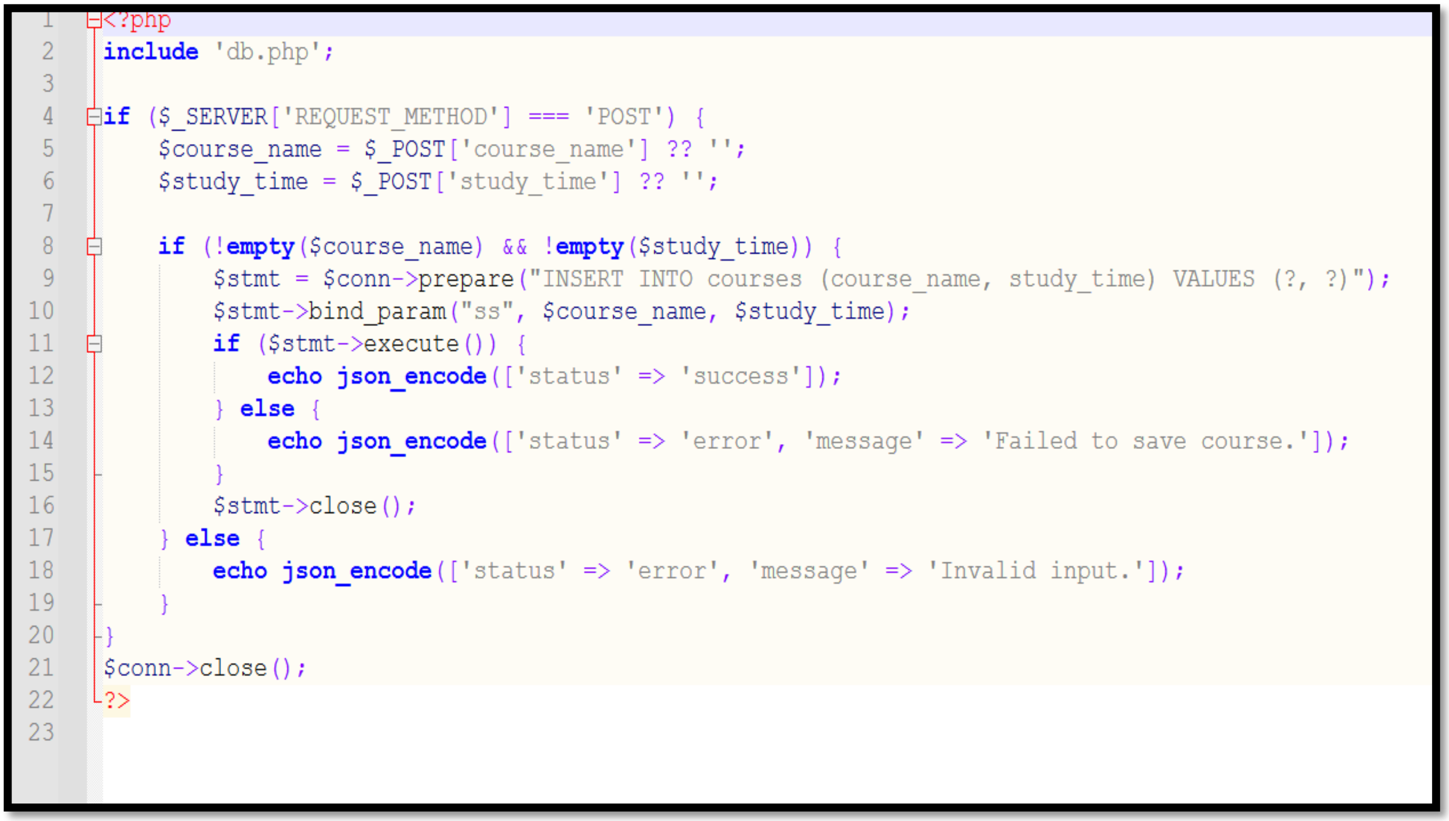
```php
<?php
include 'db.php';

header('Content-Type: application/json');

$sql = "SELECT id, course_name, study_time FROM courses ORDER BY created_at DESC";
$result = $conn->query($sql);

$courses = [];
if ($result && $result->num_rows > 0) {
    while ($row = $result->fetch_assoc()) {
        $courses[] = [
            'id' => $row['id'],
            'name' => $row['course_name'],
            'time' => $row['study_time'],
        ];
    }
}

echo json_encode($courses);
$conn->close();
?>
```



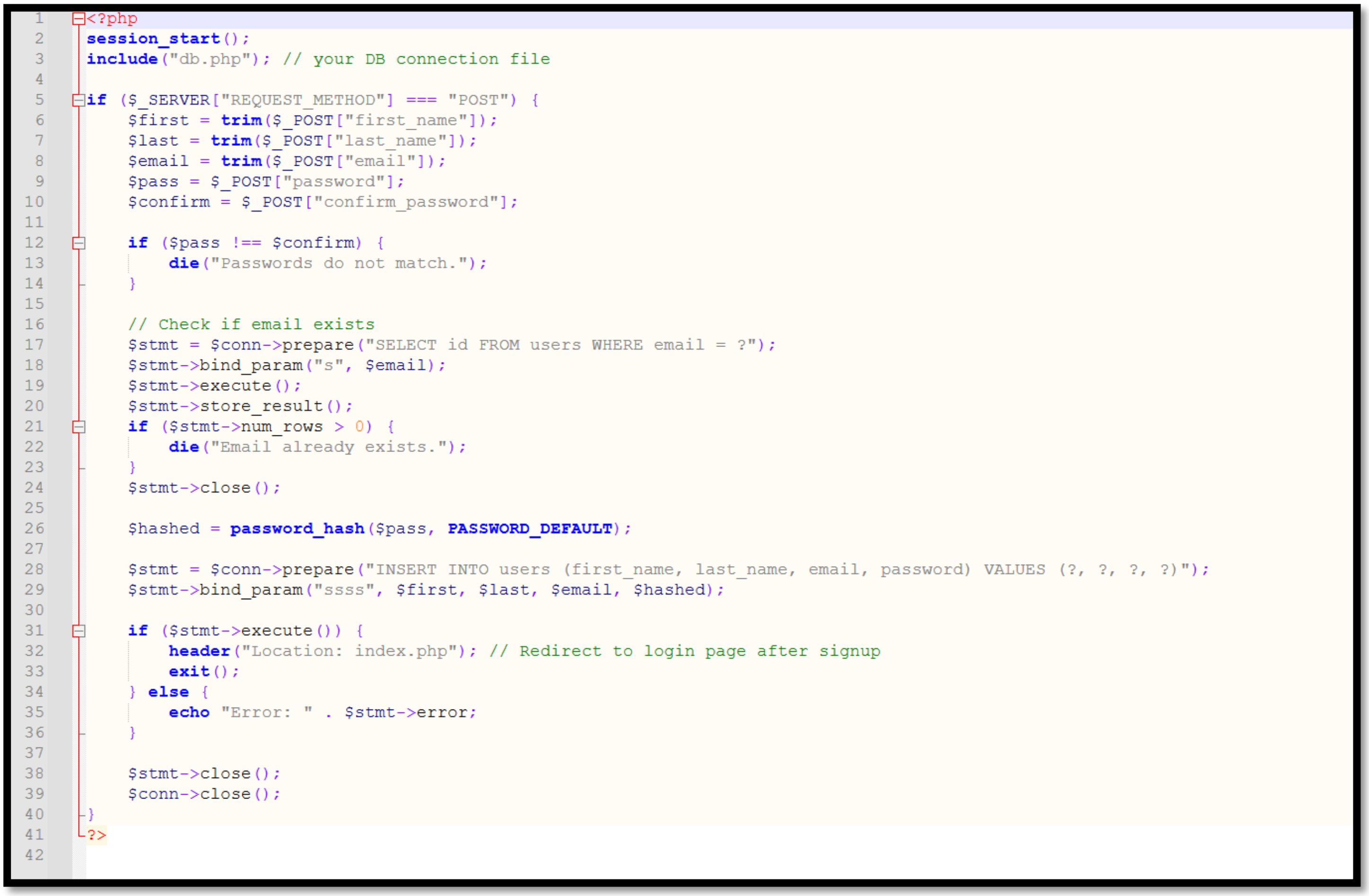
```php
<?php
session_start();
include 'db.php';

header('Content-Type: application/json');

$user_id = $_SESSION['user_id']; // assuming user logged in

$stmt = $conn->prepare("SELECT id, task_text, completed FROM tasks WHERE user_id = ? ORDER BY created_at DESC");
$stmt->bind_param("i", $user_id);
$stmt->execute();
$result = $stmt->get_result();

$tasks = [];
while ($row = $result->fetch_assoc()) {
    $tasks[] = $row;
}

echo json_encode($tasks);

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



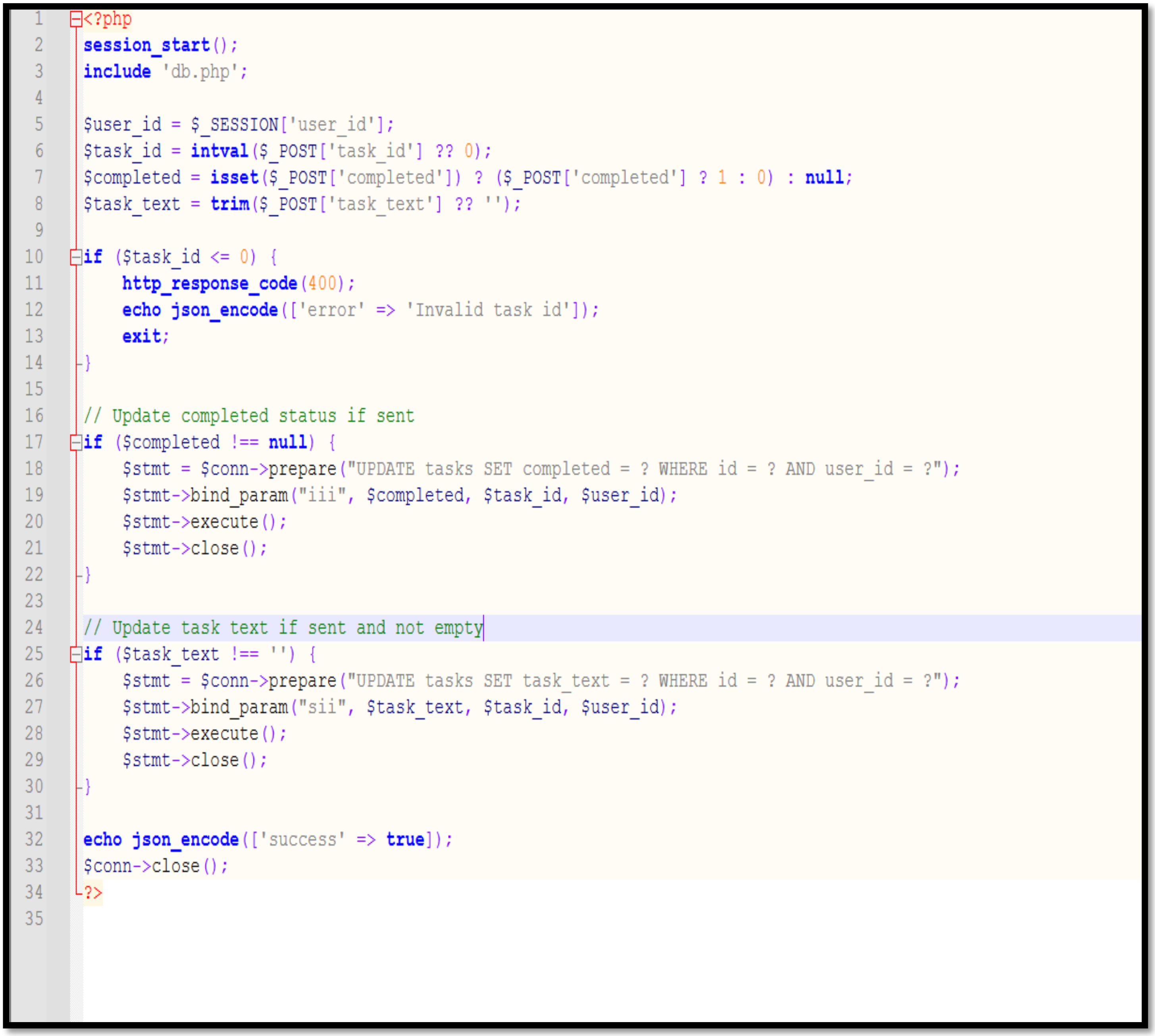
```php
<?php
session_start();
include("db.php");
$_SESSION["user_id"] = $id;
$_SESSION["username"] = $firstName;
if ($_SERVER["REQUEST_METHOD"] === "POST") {
    $email = trim($_POST["email"]);
    $password = $_POST["password"];

    $stmt = $conn->prepare("SELECT id, first_name, password FROM users WHERE email = ?");
    $stmt->bind_param("s", $email);
    $stmt->execute();
    $stmt->store_result();

    if ($stmt->num_rows === 1) {
        $stmt->bind_result($id, $firstName, $hashedPassword);
        $stmt->fetch();

        if (password_verify($password, $hashedPassword)) {
            $_SESSION["user_id"] = $id;
            $_SESSION["username"] = $firstName;  // This is what you show on homepage
            header("Location: homepage.php");
            exit();
        } else {
            header("Location: index.php?error=1");
            exit();
        }
    } else {
        header("Location: index.php?error=1");
        exit();
    }

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



```php
<?php
include 'db.php';

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $course_name = $_POST['course_name'] ?? '';
    $study_time = $_POST['study_time'] ?? '';

    if (!empty($course_name) && !empty($study_time)) {
        $stmt = $conn->prepare("INSERT INTO courses (course_name, study_time) VALUES (?, ?)");
        $stmt->bind_param("ss", $course_name, $study_time);
        if ($stmt->execute()) {
            echo json_encode(['status' => 'success']);
        } else {
            echo json_encode(['status' => 'error', 'message' => 'Failed to save course.']);
        }
        $stmt->close();
    } else {
        echo json_encode(['status' => 'error', 'message' => 'Invalid input.']);
    }
}
$conn->close();
?>
```



```php
<?php
session_start();
include("db.php"); // your DB connection file

if ($_SERVER["REQUEST_METHOD"] === "POST") {
    $first = trim($_POST["first_name"]);
    $last = trim($_POST["last_name"]);
    $email = trim($_POST["email"]);
    $pass = $_POST["password"];
    $confirm = $_POST["confirm_password"];

    if ($pass !== $confirm) {
        die("Passwords do not match.");
    }

    // Check if email exists
    $stmt = $conn->prepare("SELECT id FROM users WHERE email = ?");
    $stmt->bind_param("s", $email);
    $stmt->execute();
    $stmt->store_result();
    if ($stmt->num_rows > 0) {
        die("Email already exists.");
    }
    $stmt->close();

    $hashed = password_hash($pass, PASSWORD_DEFAULT);

    $stmt = $conn->prepare("INSERT INTO users (first_name, last_name, email, password) VALUES (?, ?, ?, ?)");
    $stmt->bind_param("ssss", $first, $last, $email, $hashed);

    if ($stmt->execute()) {
        header("Location: index.php"); // Redirect to login page after signup
        exit();
    } else {
        echo "Error: " . $stmt->error;
    }

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



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

$user_id = $_SESSION['user_id'];
$task_id = intval($_POST['task_id'] ?? 0);
$completed = isset($_POST['completed']) ? ($_POST['completed'] ? 1 : 0) : null;
$task_text = trim($_POST['task_text'] ?? '');

if ($task_id <= 0) {
    http_response_code(400);
    echo json_encode(['error' => 'Invalid task id']);
    exit;
}

// Update completed status if sent
if ($completed !== null) {
    $stmt = $conn->prepare("UPDATE tasks SET completed = ? WHERE id = ? AND user_id = ?");
    $stmt->bind_param("iii", $completed, $task_id, $user_id);
    $stmt->execute();
    $stmt->close();
}

// Update task text if sent and not empty
if ($task_text !== '') {
    $stmt = $conn->prepare("UPDATE tasks SET task_text = ? WHERE id = ? AND user_id = ?");
    $stmt->bind_param("sii", $task_text, $task_id, $user_id);
    $stmt->execute();
    $stmt->close();
}

echo json_encode(['success' => true]);
$conn->close();
?>
```

6.0 SYSTEM TESTING AND INSTALLATION

6.1 HEURISTIC EVALUATION:

Heuristic evaluation is a usability inspection method where a small group of evaluators examine the interface and judge its compliance with established usability principles (heuristics), such as those defined by Jakub Nielsen.

-METHOD:

- Three team members acted as evaluators.
- They reviewed the system based on Nielsen’s 10 heuristics.
- Each evaluator independently explored the interface and recorded usability issues.

- KEY FINDINGS:

Heuristic	Issue Identified	Severity	Recommendation
Visibility of system status	No feedback shown after saving a course	Medium	Add success message or toast notification
Error prevention	No input validation in task form	High	Add required field indicators
Consistency and standards	Some button styles inconsistent across pages	Low	Unify styling using shared CSS classes
Match between system and world	"Pomodoro" term not explained	Low	Add tooltip or help icon

CONCLUSION:

Most usability issues were minor and easily fixable. The system aligns well with most usability principles after applying the recommendations.

6.2 COOPERATIVE EVALUATION

Cooperative evaluation involves real users interacting with the system while providing live feedback. The goal is to understand user behavior, expectations, and potential confusion.

PARTICIPANTS:

- users (university students)
- Each was asked to complete common tasks such as:

1.Adding a new course

2.Starting the study timer

3.Viewing statistic

PROCEDURE:

- Users were observed while using the system without guidance.
- They were encouraged to **"think aloud"** during the process.
- Observers noted confusion points, errors, or hesitation.

OBSERVATIONS:

Task	User Feedback	Issue?	Suggested Fix
Add Course	Simple and clear	No	—
Start Timer	Timer start not clearly indicated	Yes	Highlight active timer visually
View Statistics	Stats were helpful but too simple	Yes	Add more visual graphs if time allows

CONCLUSION:

Cooperative testing showed that users found the system intuitive, with only minor improvements needed to enhance clarity and feedback.

7.0 CONCLUSION AND FUTURE WORK

7.1 Overall Weaknesses

Despite the strengths of the current system, a few weaknesses and limitations were identified during development and evaluation:

- **No Real Authentication Logic**: While the system has login and sign-up interfaces, user authentication is not connected to a secure back-end or hashed passwords.
- **No Role-Based Access**: All users have the same view and access; there are no distinct roles like admin, student, or instructor.
- **Limited Data Persistence**: Most of the data relies on Local Storage or temporary database interaction, with no robust session management.
- **Basic Visual Analytics**: The statistics provided are simple text-based summaries and could be more informative with visual graphs.
- **Lack of Multi-Language Support**: Currently, the system supports English only, which may limit usability for Arabic-speaking users.

These limitations do not prevent the system from functioning, but they highlight areas for future development and optimization.

7.2 Overall, Strengths

The current version of StudyMate demonstrates several strengths that contribute to its usability and effectiveness:

- **User-Friendly Interface**: Simple, clean, and responsive design makes navigation intuitive even for first-time users.
- **Effective Task Management**: To-Do list and course tracking help students organize their study routines efficiently.
- **Interactive Study Tools**: The built-in Pomodoro timer enhances focus and time management.
- **Visual Feedback**: The statistics screen provides students with clear feedback about their study progress.
- **Scalable Architecture**: The modular design allows for easy expansion or feature addition in the future.

These strengths show that the core objectives of the system were successfully me

7.3 FUTURE WORK

Based on user feedback and potential improvements, the following ideas are proposed for future versions:

- **Mobile App Integration** for studying on the go.
- **Advanced Analytics** like charts, graphs, and detailed time logs.
- **Reminder System** with push notifications.
- **Multi-language Support**, including Arabic for native users.
- **Admin & Teacher Roles** for monitoring and managing students’ study activity.

8.0 REFERENCES

- Ahmed Abu sharia, Second Semester, 2025
- Nielsen, J. (1994). *10 Usability Heuristics for User Interface Design*. Nielsen Norman Group. Retrieved from https://www.nngroup.com/articles/ten-usability-heuristics/
- W3Schools. (2025). *HTML, CSS, JavaScript tutorials*. Retrieved from https://www.w3schools.com/
- Bootstrap. (2025). *Bootstrap v5.3 Documentation*. Retrieved from https://getbootstrap.com/
- phpMyAdmin. (2025). *phpMyAdmin Documentation*. Retrieved from https://www.phpmyadmin.net/
- Cover. (2025). *Free Background Videos for Websites*. Retrieved from https://coverr.co/
- Pixels. (2025). *Free stock videos for web and mobile*. Retrieved from https://www.pexels.com/videos
- Mozilla Developer Network (MDN). (2024). *Web technology documentation*. Retrieved from https://developer.mozilla.org/
- Stack Overflow. (2025). *Community-driven Q&A for coding issues*. Retrieved from https://stackoverflow.com/
- OpenAI ChatGPT. (2025). *AI-powered assistant for project guidance, documentation writing, and coding help*. Retrieved from https://chat.openai.com