

VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY  
UNIVERSITY OF TECHNOLOGY  
FACULTY OF COMPUTER SCIENCE AND ENGINEERING



WEB PROGRAMMING (CO2014)

---

Group Lab Assignment

# Web-Based Multiple-Choice Question Test Application

---

Lecturer: Nguyen Duc Thai, *HCMUT*

Students: Nguyen Tran Huy Viet - 2252906

Tran Trung Vinh - 2252914

Le Nhan Van - 2252899

# Contents

<b>1</b>	<b>Contribution</b>	<b>3</b>
<b>2</b>	<b>Project Features &amp; Requirements</b>	<b>4</b>
2.1	Phase 1: Requirement Analysis & Planning (Weeks 1-2)	4
2.2	Phase 2: Database & Backend Development (Weeks 3-5)	4
2.3	Phase 3: Frontend UI Development (Weeks 6-8)	4
2.4	Phase 4: Test Functionality Implementation (Weeks 9-10)	4
2.5	Phase 5: Testing & Deployment (Weeks 11-12)	4
<b>3</b>	<b>Requirement Analysis &amp; Planning</b>	<b>5</b>
3.1	Requirement Document	5
3.1.1	Domain Context	5
3.1.2	Stakeholders and Needs	5
3.1.3	Functional Requirement	5
3.1.4	Non-Functional Requirement	6
3.2	ERD (Entity-Relationship Diagram) for database design	7
3.3	Work breakdown structure.	8
<b>4</b>	<b>Database &amp; Backend Development</b>	<b>10</b>
4.1	Database schema with tables for users, questions, categories, tests, and results.	10
4.2	User authentication system	10
4.3	CRUD operations for admin	12
4.3.1	Create questions, tests	12
4.3.2	View questions, tests	14
4.3.3	Update and delete questions	14
4.3.4	Update student status	15
4.4	Implement image upload feature for questions	16
4.5	Admin panel to retrieve and select questions for a test	16
<b>5</b>	<b>Frontend UI Development</b>	<b>17</b>
5.1	Homepage, login, and test-taking pages with CSS	17
5.2	Category selection page before starting a test	19
5.3	Question display with answer selection functionality	20
5.4	Timer countdown on test page	20
5.5	Image display with each question	21
<b>6</b>	<b>Test Functionality Implementation</b>	<b>22</b>
6.1	Functional test-taking feature (answer selection and submission)	22
6.2	Store user test attempts in the database and allow users to review past tests.	23
<b>7</b>	<b>User manual</b>	<b>24</b>
7.1	Guest	24
7.2	Student	24
7.3	Admin	25
<b>8</b>	<b>How to set up the website</b>	<b>26</b>
<b>9</b>	<b>Conclusion</b>	<b>27</b>
<b>10</b>	<b>References</b>	<b>28</b>

## List of Figures

1	Enhanced Entity-Relationship Diagram of Ligma website . . . . .	7
2	Initial work break down structure . . . . .	8
3	Databse schema for the Ligma website . . . . .	10
4	Add test questions area . . . . .	12
5	Add questions . . . . .	13
6	Add test step 1: Fill out test information . . . . .	13
7	Question tabs . . . . .	14
8	Test tabs . . . . .	14
9	Question card with edit and delete options . . . . .	15
10	Student tab with search and sort . . . . .	15
11	Student account with denied access . . . . .	16
12	Add test step 2: choose questions for test . . . . .	16
13	Multiple Choice Website Homepage Layout (User) . . . . .	17
14	Sign In Section . . . . .	17
15	Sign Up Section . . . . .	18
16	Quiz Detail Page (User) . . . . .	18
17	Quiz Detail Page (Guest) . . . . .	19
18	Quiz Detail Page (Guest) . . . . .	19
19	Explore Page . . . . .	20
20	A section with question display with answer selections (Test taking page) . . . . .	20
21	Timer turns red when under 60 seconds left . . . . .	21
22	Question with the uploaded image . . . . .	21
23	Test information for students . . . . .	22
24	Test view for guests and users . . . . .	22
25	Test taking proccess for student . . . . .	23
26	Attempt review for student . . . . .	23



# 1 Contribution

Student	Student ID	Contribution	Percentage of work
Trần Trung Vĩnh	2252914	Ui/Ux desgin, front-end implementation and deploy	100%
Nguyễn Trần Huy Việt	2252906	Database desgin, database and backend implementation, writing reports	100%
Lê Nhân Văn	2252899		0%



## 2 Project Features & Requirements

### 2.1 Phase 1: Requirement Analysis & Planning (Weeks 1-2)

#### Deliverables

1. **Requirement Document** (features, use cases, user stories).
2. **ERD (Entity-Relationship Diagram)** for database design
3. **Work breakdown structure** (who does what in the group).

### 2.2 Phase 2: Database & Backend Development (Weeks 3-5)

#### Deliverables

1. **Database schema** with tables for **users, questions, categories, tests, and results**.
2. **User authentication system** (Admin, Registered Users, Guests).
3. **CRUD operations for admin** (Add/Edit/Delete Questions, Categories & Users).
4. **Implement image upload feature** for questions.
5. **Admin panel to retrieve and select questions for a test** (new feature).

### 2.3 Phase 3: Frontend UI Development (Weeks 6-8)

#### Deliverables

1. **Homepage, login, and test-taking pages** with CSS.
2. **Responsive layout** using Flexbox/Grid.
3. **Category selection page** before starting a test.
4. **Question display with answer selection functionality**.
5. **Timer countdown** on test page.
6. **Image display** with each question.

### 2.4 Phase 4: Test Functionality Implementation (Weeks 9-10)

#### Deliverables

1. **Functional test-taking feature** (answer selection and submission).
2. **Store user test attempts in the database**.
3. **Allow users to review past tests**

### 2.5 Phase 5: Testing & Deployment (Weeks 11-12)

#### Deliverables

1. **Testing Report** (Bug fixes, Performance, Security)
2. **Final working web application** deployed on a server.
3. **Final Project Report & User Manual**.

## Source code

The entire source code for this assignment can be found on [this github repository](https://github.com/HuyVietFeb4/MCQ-Web) or go to the next url: <https://github.com/HuyVietFeb4/MCQ-Web>

## 3 Requirement Analysis & Planning

### 3.1 Requirement Document

#### 3.1.1 Domain Context

In this assignment, we developed a web-based application named "Ligma" for administering multiple-choice question tests. Ligma is designed to provide a free platform for students of all levels—from primary school to undergraduate studies—to practice and enhance their knowledge across various subjects, including mathematics, history, literature, calculus, and computer architecture. Each week, the website features a new set of tests for students to take. Users can create accounts, access tests, and review their results to track their progress. Administrators have the ability to manage the question bank by adding new questions, categorizing and sorting them, and creating tailored tests for students on a weekly basis.

#### 3.1.2 Stakeholders and Needs

- Students: from all levels can access a comprehensive set of features designed to support their learning journey. They have the ability to view detailed test overviews and statistics, take new tests, and review their past attempts to analyze their progress and identify areas for improvement.
- Admins: have full control over managing the content and user accounts within the platform. They can create, edit, and delete their own questions, curate tests by selecting questions from the question bank, and modify or remove their own tests as needed. Additionally, admins have the authority to disable student accounts when necessary.
- Guests: can explore the platform with limited access. They have the ability to log in, sign up for an account, and view available tests, offering them a glimpse of the resources and features offered.

#### 3.1.3 Functional Requirement

##### Students

- View tests overview and tests statistics include average points, total attempts of the tests
- Take and submit tests in a given times frames
- View past attempts and scores
- Search and sort tests according to names or category
- Sign up to create an account and sign in
- Sent contact us form
- Select an answer and submit.
- Students cannot go back to previous questions once submitted.

##### Admin

- View tests overview and tests statistics include average points, total attempts of the tests
- Create, edit and delete tests
- Create, edit and delete questions
- Search and sort tests and questions according to names or category
- Sign up to create an account and sign in
- Sent contact us form
- Can add picture for questions when create/edit
- Each questions only has one answers

##### Guests

- Create an account
- View tests overview and tests statistics include average points, total attempts of the tests
- Sent contact us form

#### **3.1.4 Non-Functional Requirement**

- The website should be compatible with major browsers (e.g., Chrome, Firefox, Safari) and responsive on various devices, including desktops, tablets, and mobile phones.
- Passwords are securely stored using the RIPEMD-128 hash function combined with a salting technique. Salting adds a unique random value to each password before hashing, preventing attacks such as dictionary exploits and rainbow tables.
- The questions must only have 4 options
- The test must have at least 5 unique questions
- The picture in questions must be JPEG, PNG, GIF formats with the max size of 2 MB

### 3.2 ERD (Entity-Relationship Diagram) for database design

The database design for the website is organized and functional, supporting user registration, role assignment, and activity tracking. To manage user accounts, the "Users" table stores essential registration details like Password, Email, User\_name, and User\_ID. The User\_ID serves as a key to determine user roles, categorizing them into either "Students" or "Admins" based on their signup details. The "Admins" table contains Admin\_ID, while the "Students" table includes Student\_ID, and these roles dictate what actions each user can perform.

Admins are empowered to create multiple questions, each featuring four options. These options are defined in the "Options" entity, a weak entity where the primary key combines Option\_ID (ranging from 1 to 4) and Question\_ID from the parent "Questions" entity. Admins can select questions from the "Questions" database to assemble tests, which they exclusively own.

Students participate in these tests, generating new "Test\_attempt" entries for each attempt. Every test attempt includes "Question\_attempt" entities that meticulously record student performance on each question. This structure ensures detailed tracking of user activities, test creation, and student progress.

Additionally, the "Contact" entity captures user contact information, storing attributes like Contact\_ID, Email, Contact\_date, Description, and Title. The interrelations between these entities create a cohesive database design for comprehensive user and test management.

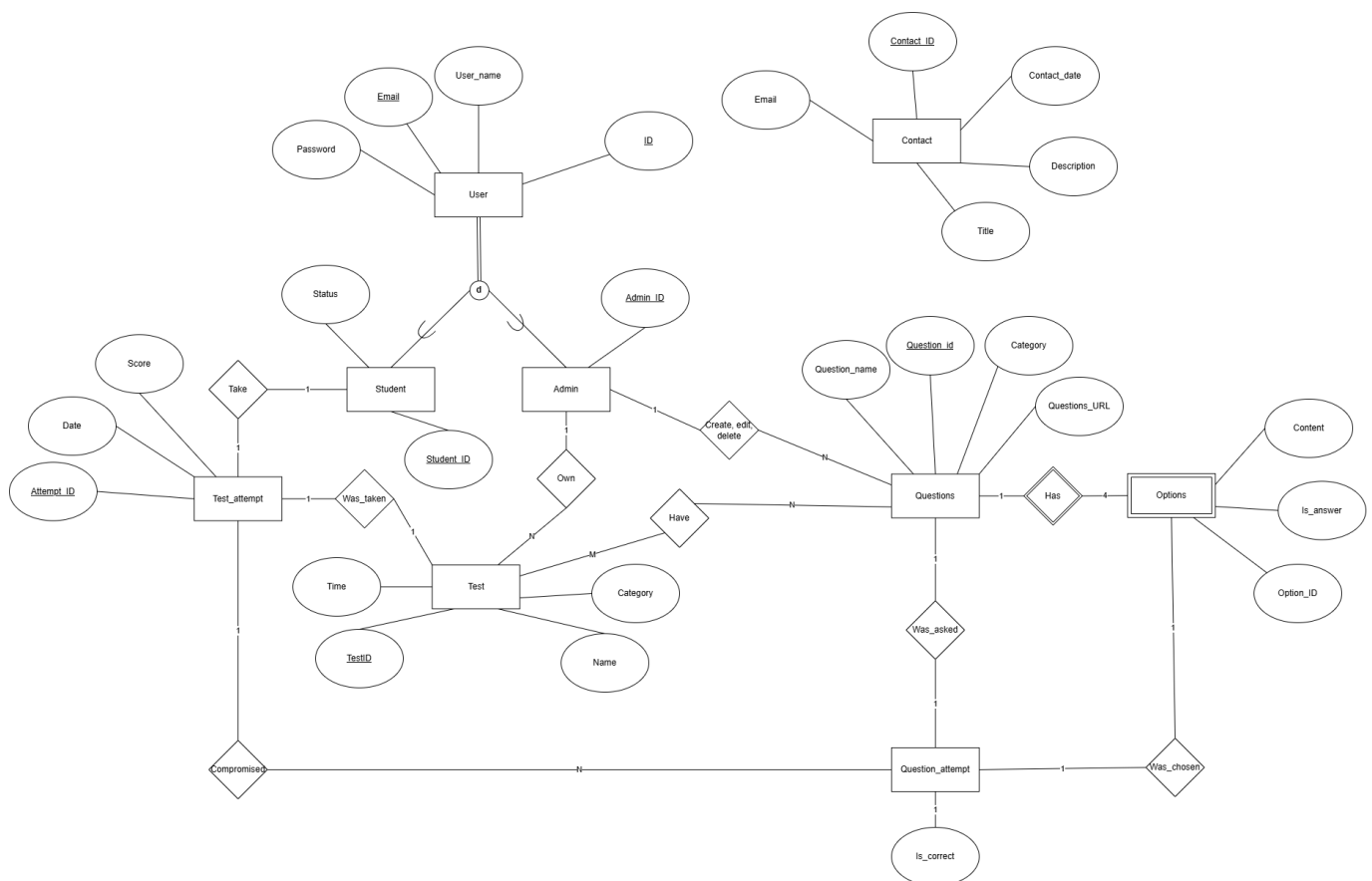


Figure 1: Enhanced Entity-Relationship Diagram of Ligma website



### 3.3 Work breakdown structure.

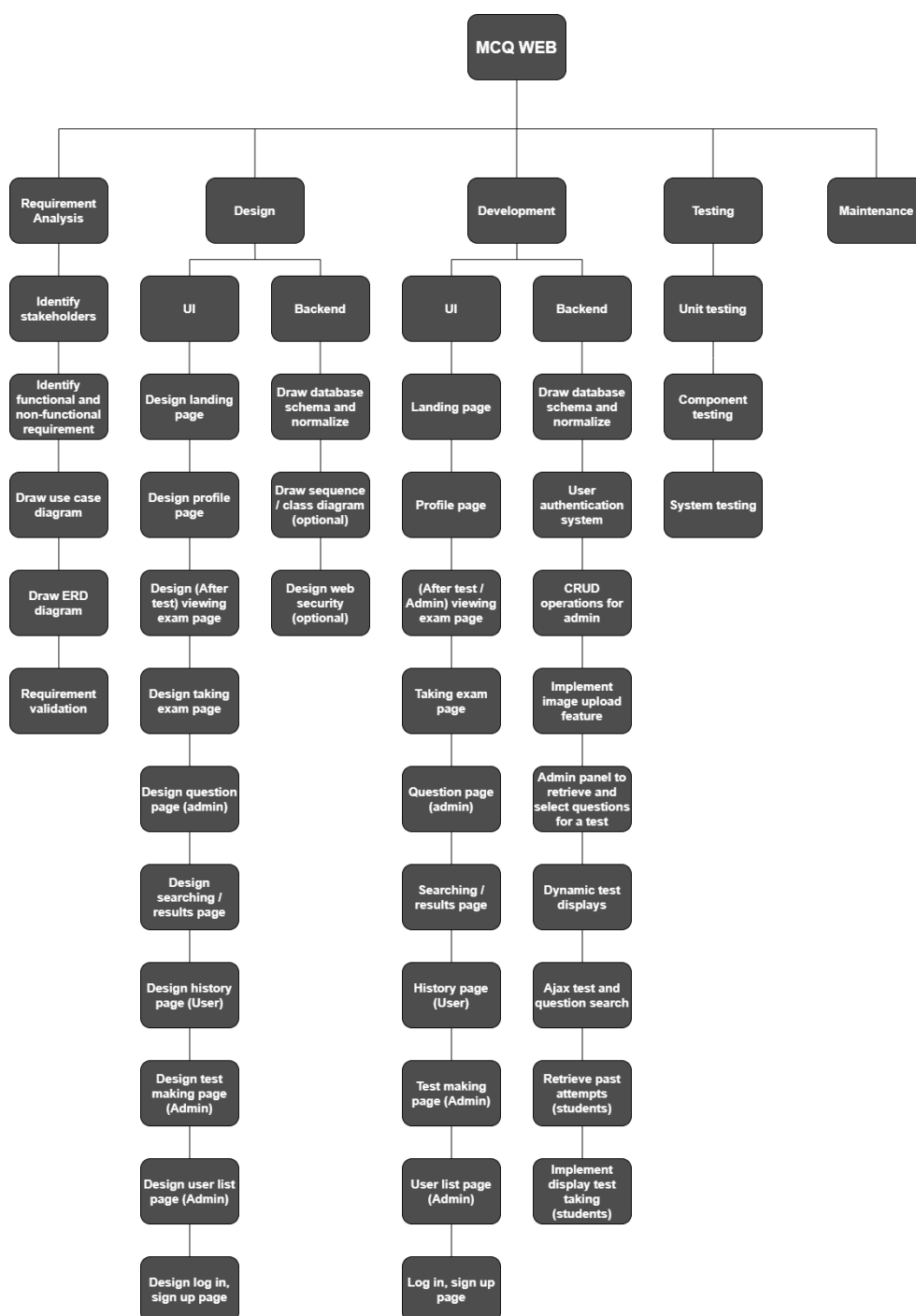


Figure 2: Initial work break down structure



Initially, Nhan Van was assigned to handle requirement analysis and UI/UX development, Trung Vinh focused on UI/UX design and a bit of backend development, and Huy Viet took charge of database design and implementation. The testing responsibilities were intended to be shared equally among all three members. However, with Nhan Van withdrawing from the subject and stepping back from the development process very late to the course without doing anything meaningful, the tasks were redistributed between Trung Vinh and Huy Viet. Here is the final work breakdown structure:

- Huy Viet: Database design, backend design and implementation, requirement analysis, writing report
- Trung Vinh: UI/UX design and implementation, deploy website on servers.

That means only the work of requirement analysis, design and development have been done for this website in the short period of time of just a little bit more than 1 week.

## 4 Database & Backend Development

### 4.1 Database schema with tables for users, questions, categories, tests, and results.

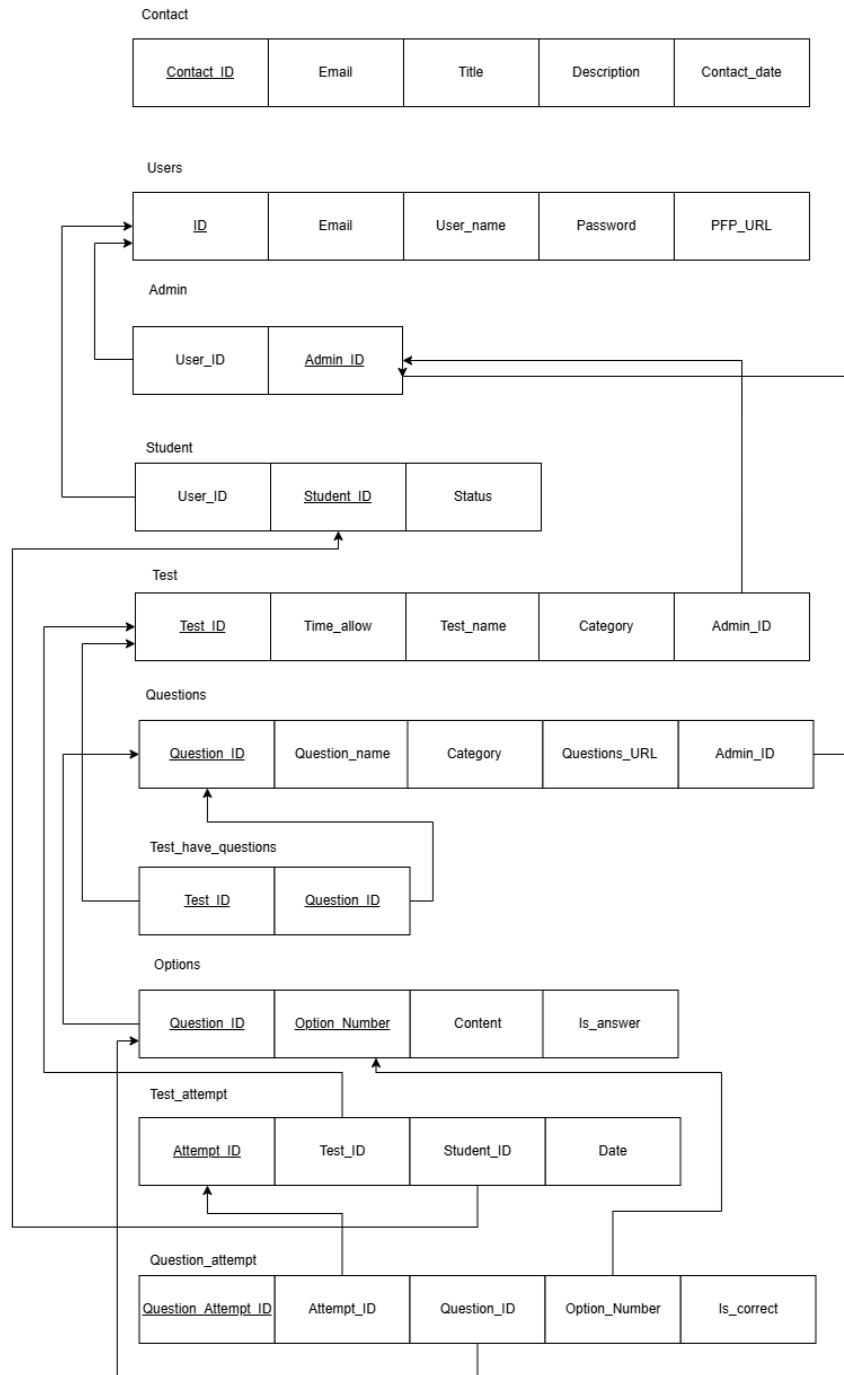


Figure 3: Database schema for the Ligma website

### 4.2 User authentication system

The user authentication system consists of two key features: sign-up and sign-in. The sign-up feature is exclusively available for student accounts, allowing students to register through the system. In contrast, admin accounts are created directly in the database via SQL insertion. This approach ensures that only authorized admin accounts can be established, effectively preventing the creation of any unauthorized or unwanted accounts.

The sign-up feature of the website implements both front-end and back-end validation to ensure secure and accurate

data entry. The front-end validation checks the format of user input and prevents empty fields, ensuring users provide complete and correctly formatted information. On the back end, validation is performed again to confirm that no fields are left empty. Additionally, the system encodes any potential JavaScript or HTML code to mitigate the risk of cross-site scripting (XSS) attacks. This is achieved through the `sanitize_input()` function, which processes and secures user-provided data. The implementation of `sanitize_input()` is as follows:

Listing 4.1: PHP `sanitize_input()` function to encode any potential JavaScript or HTML code

```
1 function sanitize_input($data) {
2     $data = trim($data);
3     $data = stripslashes($data);
4     $data = htmlspecialchars($data);
5     return $data;
6 }
```

Once all inputs are validated to ensure correct formatting, the system checks whether the provided email has already been registered. If the email is found in the database, an error message is generated to inform the user. However, if the email is not registered, the system proceeds to create the account. To enhance security, the password is hashed using the RIPEMD-128 algorithm along with a unique salt. This approach strengthens protection against unauthorized access and password-related vulnerabilities.

Listing 4.2: PHP code snippet of hashing passwords to store to the database

```
1 function sanitize_input($data) {
2     $salt1 = "%$32*~";
3     $salt2 = "fwfbgh#(";
4     $token = hash('ripemd128', "$salt1$password$salt2");
5     add_student($connection, $email, $username, $token);
6 }
```

The sign-in feature is identical for both admin and user accounts, requiring only two input fields: email and password. The system first validates the input on both the front end and the back end to ensure correctness. It then checks whether the provided email is already registered. If the email is not found in the database, the user is prompted to sign up. If the email is valid, the system proceeds to authenticate the password using the previously mentioned validation methods. Once authentication is successful, the system assigns the appropriate user role within the session based on the role associated with the email in the database.

Listing 4.3: PHP code snippet to authorize input and assign roles up sign in

```
1 if ($token == $user["Password_hash"]) {
2     $check_student_query = "SELECT * FROM Student where User_ID = ?";
3     $check_admin_query = "SELECT * FROM Admins where User_ID = ?";
4     $check_student_statement = $connection->prepare($check_student_query);
5     $check_admin_statement = $connection->prepare($check_admin_query);
6     $check_student_statement->bind_param("i", $user['User_ID']);
7     $check_student_statement->execute();
8     $student_result = $check_student_statement->get_result();
9     $check_admin_statement->bind_param("i", $user['User_ID']);
10    $check_admin_statement->execute();
11    $admin_result = $check_admin_statement->get_result();
12    if($student_result->num_rows == 1) {
13        $student = $student_result->fetch_assoc();
14        session_regenerate_id(true);
15        $_SESSION['User_ID'] = $user['User_ID'];
16        $_SESSION['User_name'] = $user['User_name'];
17        $_SESSION['Email'] = $user['Email'];
18        $_SESSION['PFP_URL'] = $user['PFP_URL'];
19        $_SESSION['Student_ID'] = $student['Student_ID'];
20        $_SESSION['is_admin'] = FALSE;
21        $_SESSION['Student_status'] = $student['Student_status'];
22        if($student['Student_status'] == 'banned') {
23            header("Location: $base_url/pages/index.php?page=you_have_been_banned");
24            exit;
25        }
26    }
```

```

27     else {
28         $admin = $admin_result->fetch_assoc();
29         session_regenerate_id(true);
30         $_SESSION['User_ID'] = $user['User_ID'];
31         $_SESSION['User_name'] = $user['User_name'];
32         $_SESSION['Email'] = $user['Email'];
33         $_SESSION['PFP_URL'] = $user['PFP_URL'];
34         $_SESSION['Admin_ID'] = $admin['Admin_ID'];
35         $_SESSION['is_admin'] = TRUE;
36     }
37     $connection->close();
38     header("Location:␣$base_url/pages/index.php?page=landing_page");
39 }
40 else {
41     $_SESSION['error_message'] = "Invalid␣passwords␣or␣email,␣try␣again.";
42     $connection->close();
43     header("Location:␣$base_url/pages/index.php?page=sign_in");
44 }

```

### 4.3 CRUD operations for admin

CRUD operations for admin include:

- Create questions, tests
- View questions, tests
- Update questions
- Delete questions
- Update student status

Let's go through each features:

#### 4.3.1 Create questions, tests

To create questions or tests, the admin must first navigate to the "Your Tests" tab or "Your Questions" tab. On the right side of the screen, a plus (+) button is available. When clicked, it presents two options: "Create Questions" and "Create Tests." The admin can then choose the desired option to proceed with creating content.

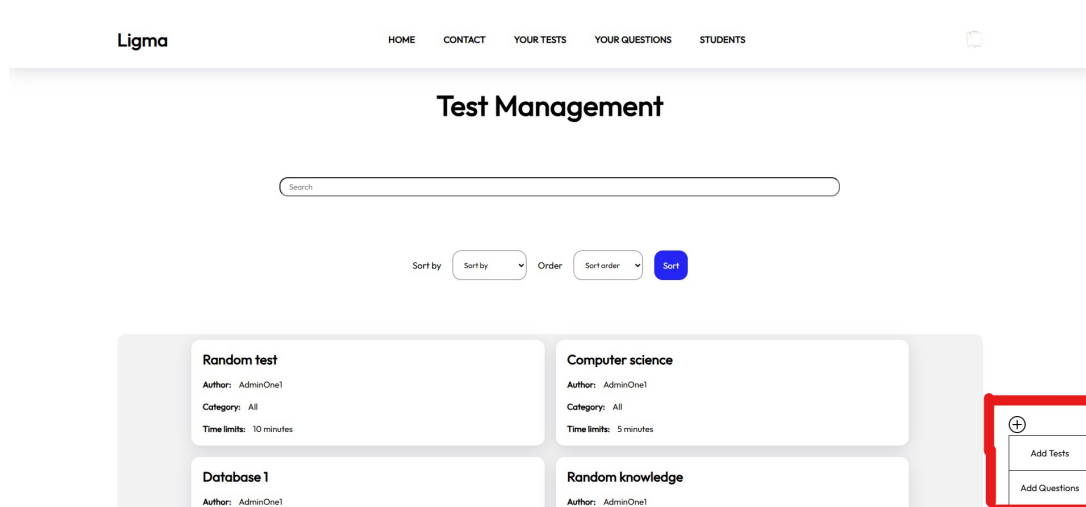
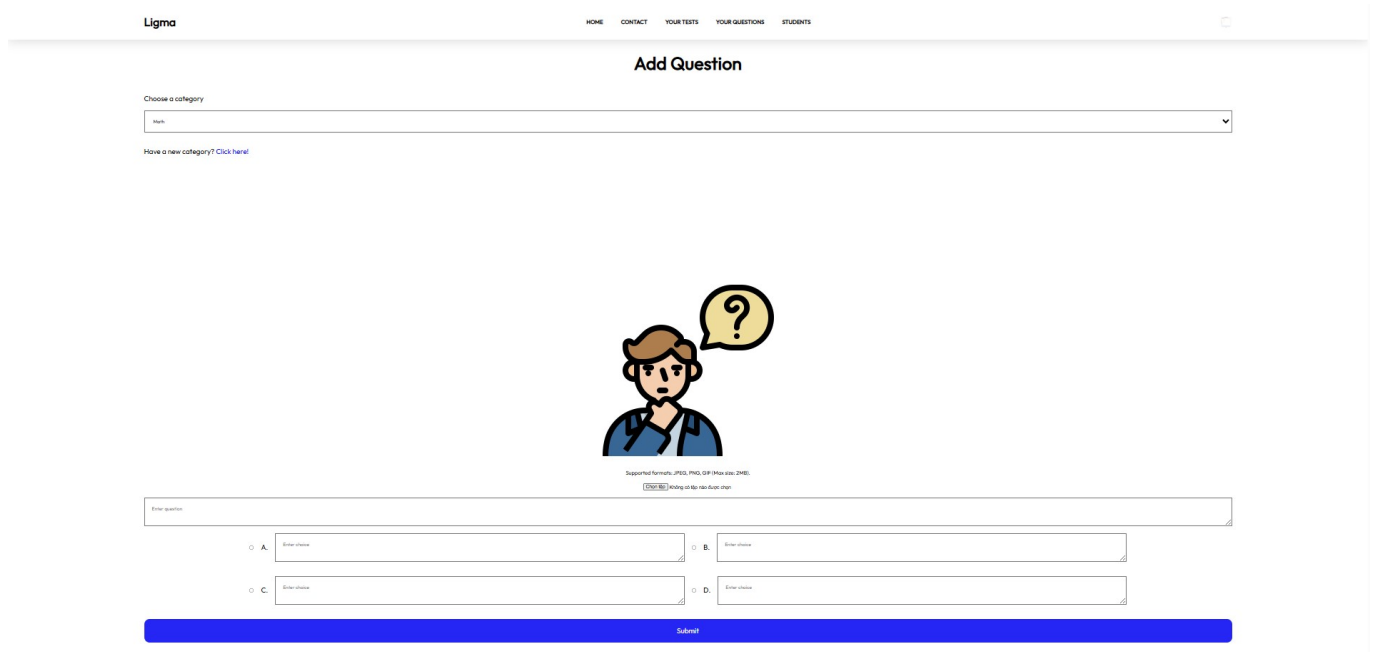


Figure 4: Add test questions area

The "Create Questions" option offers a straightforward process. Admins can create one question at a time. The interface includes a category field, allowing admins to select a predefined category. If the desired category is not listed,

they can create a new one by clicking "Have a new category? Click here!" and entering the category name. Once added, the new category will appear in the list for future selections.

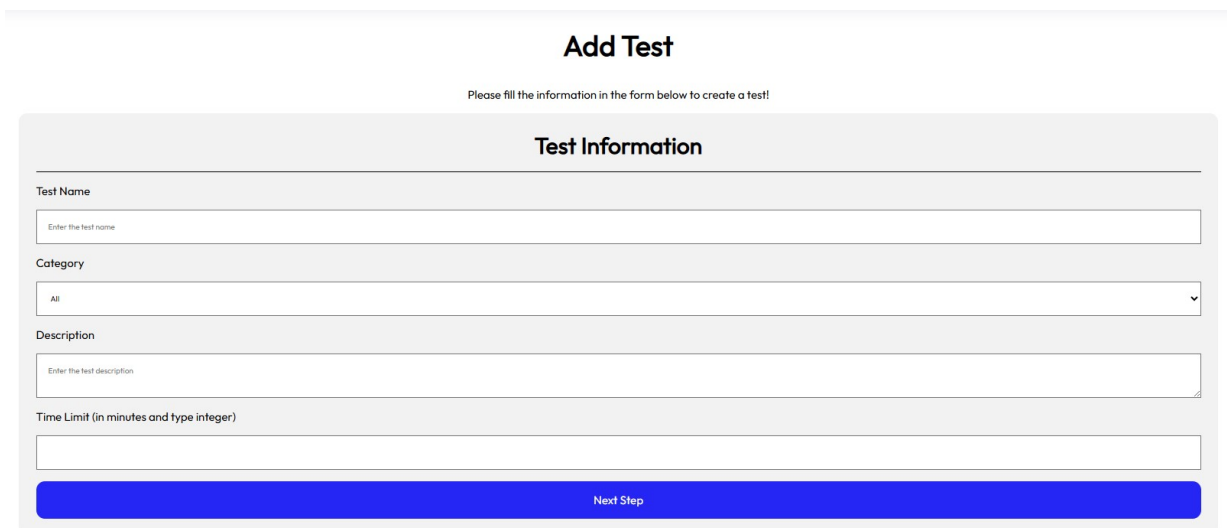


The screenshot shows the 'Add Question' interface. At the top, there's a navigation bar with 'Ligma' and links for HOME, CONTACT, YOUR TESTS, YOUR QUESTIONS, and STUDENTS. Below the navigation bar, the title 'Add Question' is centered. A dropdown menu for 'Choose a category' is shown with 'Math' selected. Below the dropdown, there's a link 'Have a new category? Click here!'. In the center, there's a cartoon character with a question mark in a speech bubble. Below the character, there's a text input field for 'Enter question'. Underneath the question field, there are four multiple-choice options labeled A, B, C, and D, each with a radio button and a text input field for the answer. At the bottom, there's a blue 'Submit' button.

Figure 5: Add questions

After selecting or adding a category, admins can proceed to fill in the question along with its four answer choices. They designate the correct answer by clicking on one of the four available options. When the admin clicks "Submit," the system processes the input and verifies that the question and all four answer choices have been provided. If any required fields are left empty, the system returns to the previous page and displays an appropriate error message.

To create a test, the admin begins by selecting the "Create Test" option, located in the same area as the "Create Questions" option. The process starts with the admin filling out the test's details, including the name, category, and time limit. Once this information is provided, the system filters the available questions based on the selected category, allowing the admin to choose which questions to include in the test. If the "All" category is selected, all questions in the database will be displayed for selection.



The screenshot shows the 'Add Test' interface. At the top, there's a navigation bar with 'Ligma' and links for HOME, CONTACT, YOUR TESTS, YOUR QUESTIONS, and STUDENTS. Below the navigation bar, the title 'Add Test' is centered. Below the title, there's a subtitle 'Please fill the information in the form below to create a test!'. The form is titled 'Test Information' and contains four input fields: 'Test Name', 'Category' (a dropdown menu with 'All' selected), 'Description', and 'Time Limit (in minutes and type integer)'. At the bottom, there's a blue 'Next Step' button.

Figure 6: Add test step 1: Fill out test information

### 4.3.2 View questions, tests

The questions and tests are shared among admins. If an admin wishes to view existing questions or tests, they can simply navigate to the "Your Tests" or "Your Questions" tabs in the header. Both tabs are equipped with sorting and AJAX search functions, allowing admins to efficiently filter and navigate through the available information.

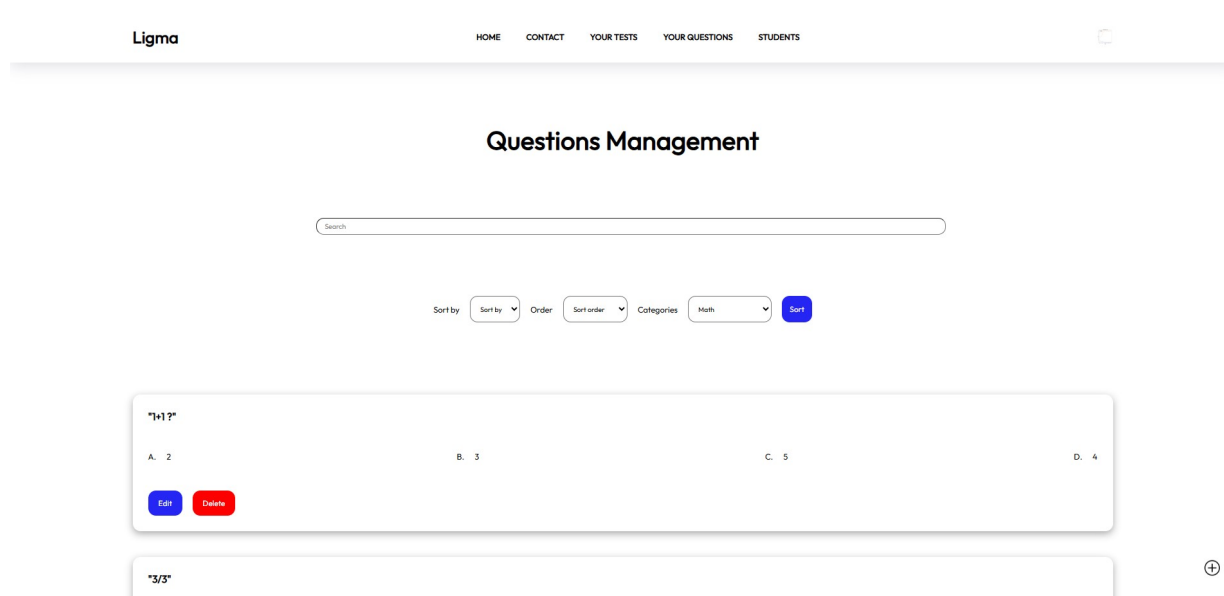


Figure 7: Question tabs

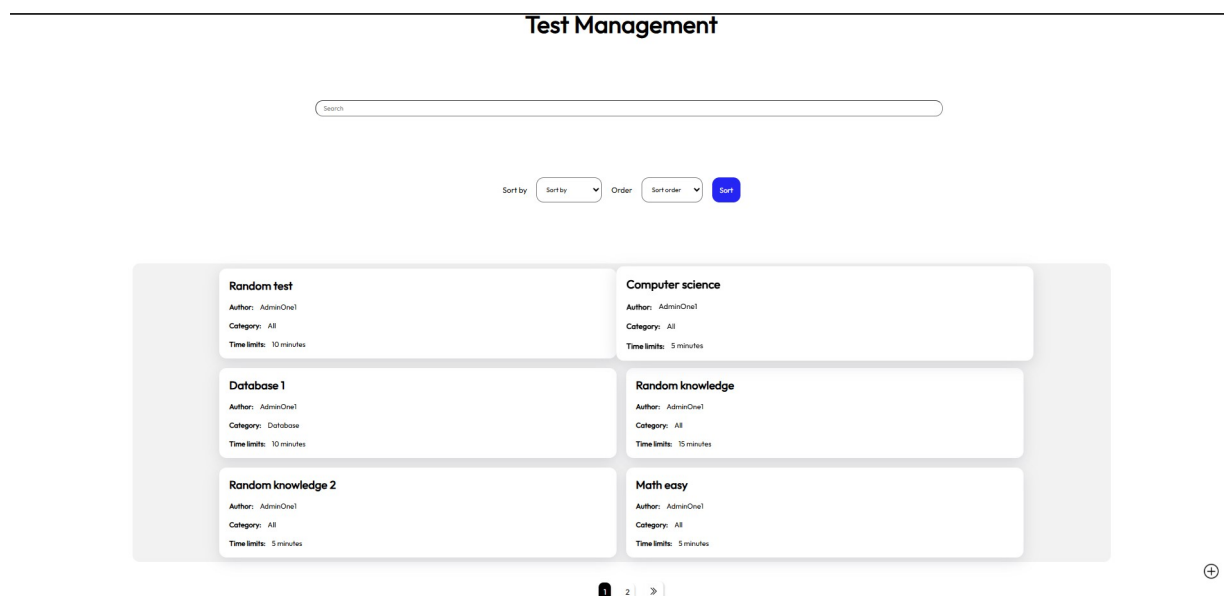


Figure 8: Test tabs

### 4.3.3 Update and delete questions

To update or delete questions, simply navigate to the "Questions" tab and choose one of the available options: "Edit" or "Delete." If the "Edit" option is selected, the editing process will closely resemble the process of adding a new question. This includes steps such as updating the question text, selecting or modifying the category, editing the four answer choices, and reassigning the correct answer if needed.

"Which type of data can be stored in the database?"

A. Image oriented data      B. Text, files containing data      C. Data in the form of audio or video      D. All of the above

Edit Delete

Figure 9: Question card with edit and delete options

#### 4.3.4 Update student status

To update a student's status, navigate to the "Student" tab. For each student, there will be a button indicating their current status. Although this might be initially confusing, the button labeled "Ban" signifies that the student is currently active, and clicking it will ban the student. Conversely, if the button displays "Active," it indicates that the student is currently banned, and clicking it will reinstate their active status. When a student is banned, any attempt to log in will result in denied access to all pages of the website.

Ligma

HOMECONTACTYOUR TESTSYOUR QUESTIONSSTUDENTS

Users Management

Search

Sort bySort byOrderSort orderSort

concacnhor

Email: viet.nguyenlodaunv1@hcmut.edu.vn

Ban

UserOne1

Email: user1@example.com

Ban

UserTwo2

Email: user2@example.com

Active

UserThree3

Email: user3@example.com

Ban

Figure 10: Student tab with search and sort

Assignment Report for Database system - Academic year 2024 - 2025

Page 15/28



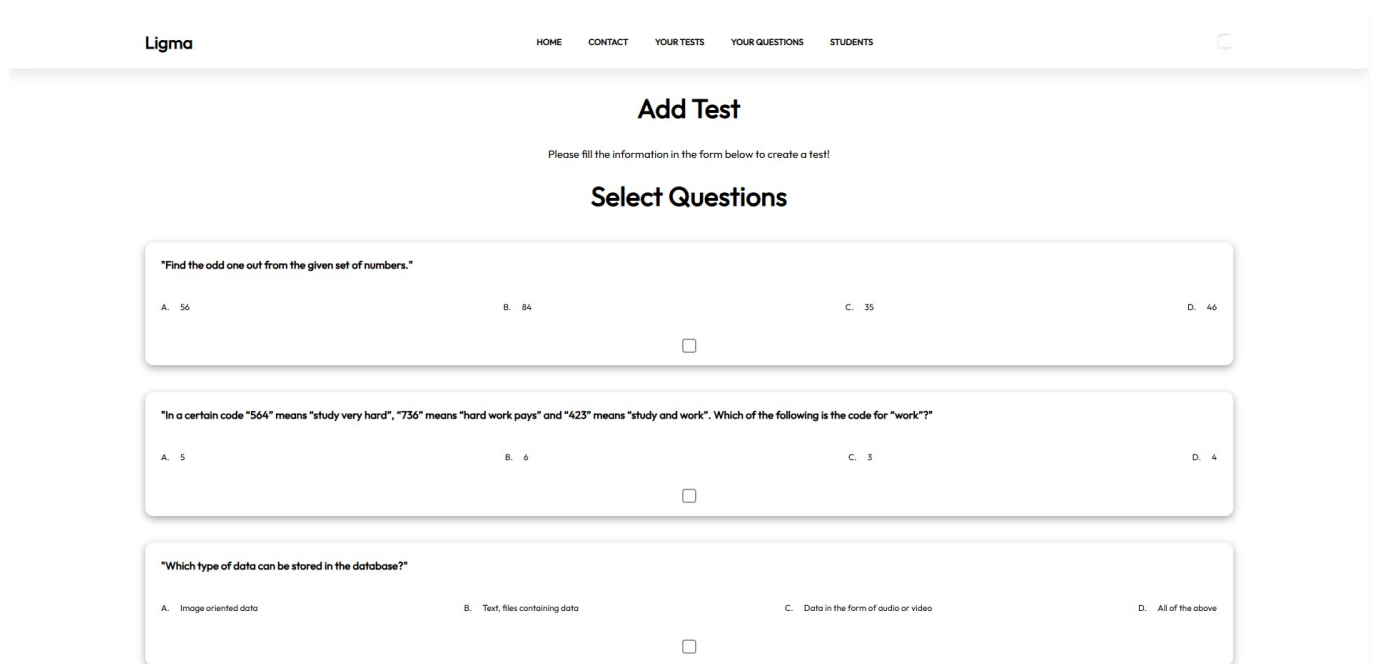
YOU HAVE BEEN BANNED

Figure 11: Student account with denied access

#### 4.4 Implement image upload feature for questions

Additionally, admins have the option to include an image with the question. If no image is provided, a default image is assigned instead. The uploaded image must meet specific criteria: it must be in .gif, .png, .jpg, or .jpeg format and must not exceed 2MB in size. If the image does not meet these requirements, the question is still inserted into the question bank but with the default image. In such cases, an error message is displayed to inform the admin about the issue.

#### 4.5 Admin panel to retrieve and select questions for a test



**Ligma** HOME CONTACT YOUR TESTS YOUR QUESTIONS STUDENTS

### Add Test

Please fill the information in the form below to create a test!

#### Select Questions

"Find the odd one out from the given set of numbers."

A. 56 B. 84 C. 35 D. 46

☐

"In a certain code "564" means "study very hard", "736" means "hard work pays" and "423" means "study and work". Which of the following is the code for "work"?"

A. 5 B. 6 C. 3 D. 4

☐

"Which type of data can be stored in the database?"

A. Image oriented data B. Text, files containing data C. Data in the form of audio or video D. All of the above

☐

Figure 12: Add test step 2: choose questions for test

## 5 Frontend UI Development

### 5.1 Homepage, login, and test-taking pages with CSS

The homepage provides a brief introduction to the website's services, including browsing available tests, taking tests, and viewing completed tests as a user. It also offers guidance for admins who are responsible for creating and managing questions and tests. The header includes several navigation links: Home (to return to the homepage), Explore (to browse quizzes and categories), Quiz (to view all quizzes created by the admin), and Contact (allowing users to share feedback or report issues related to the website's layout or services). CSS was used to design the header, footer, and overall layout of the website, including the styling, positioning, and order of text and image content.

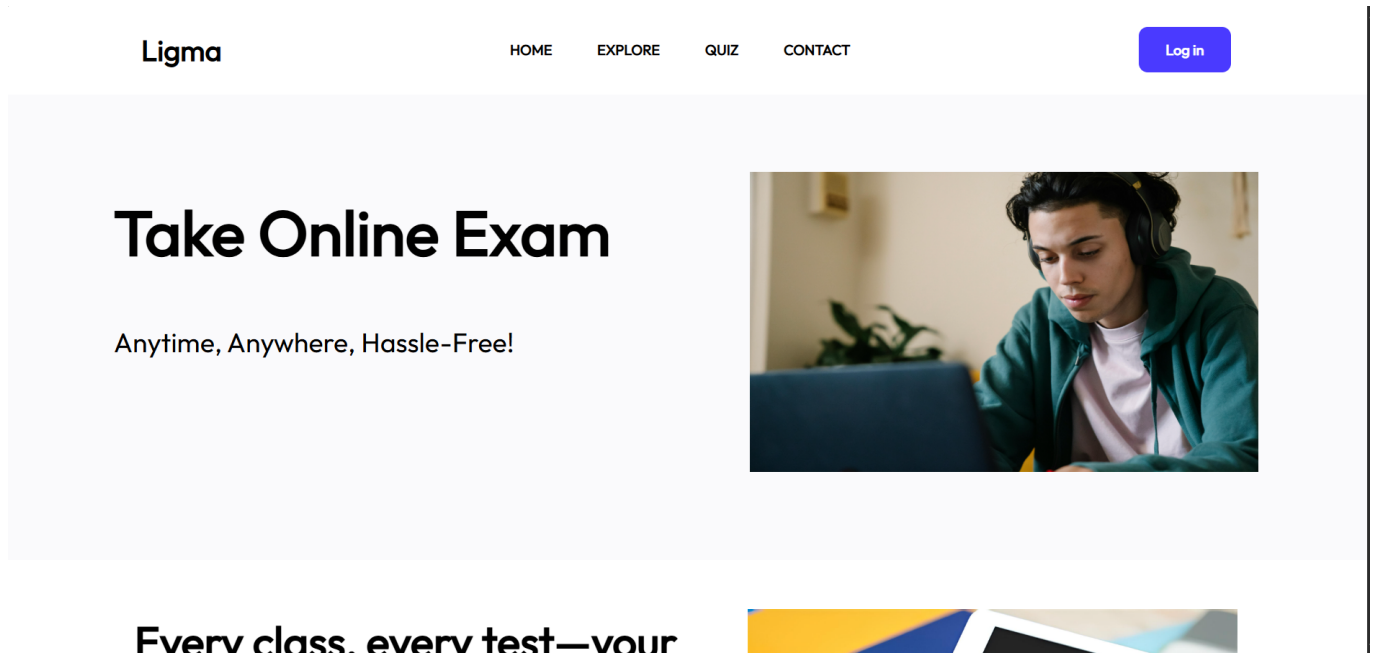


Figure 13: Multiple Choice Website Homepage Layout (User)

To access all features of the MCQ website, users must log in by clicking the blue button located in the top-right corner of the screen, which redirects them to the sign-in page.

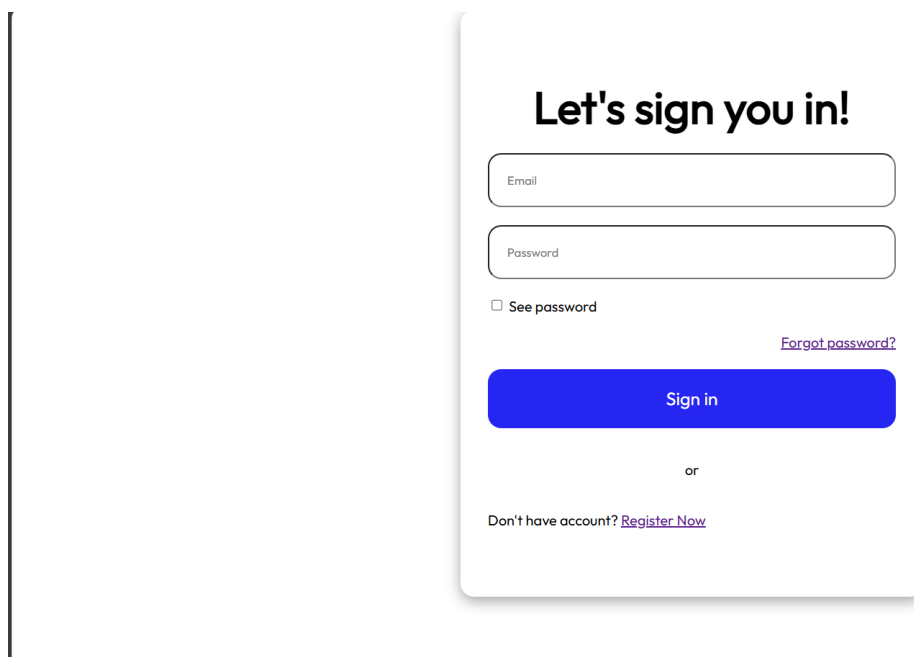
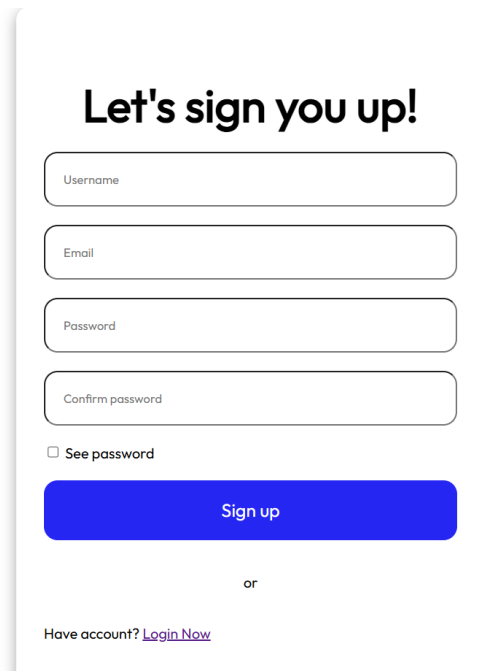


Figure 14: Sign In Section



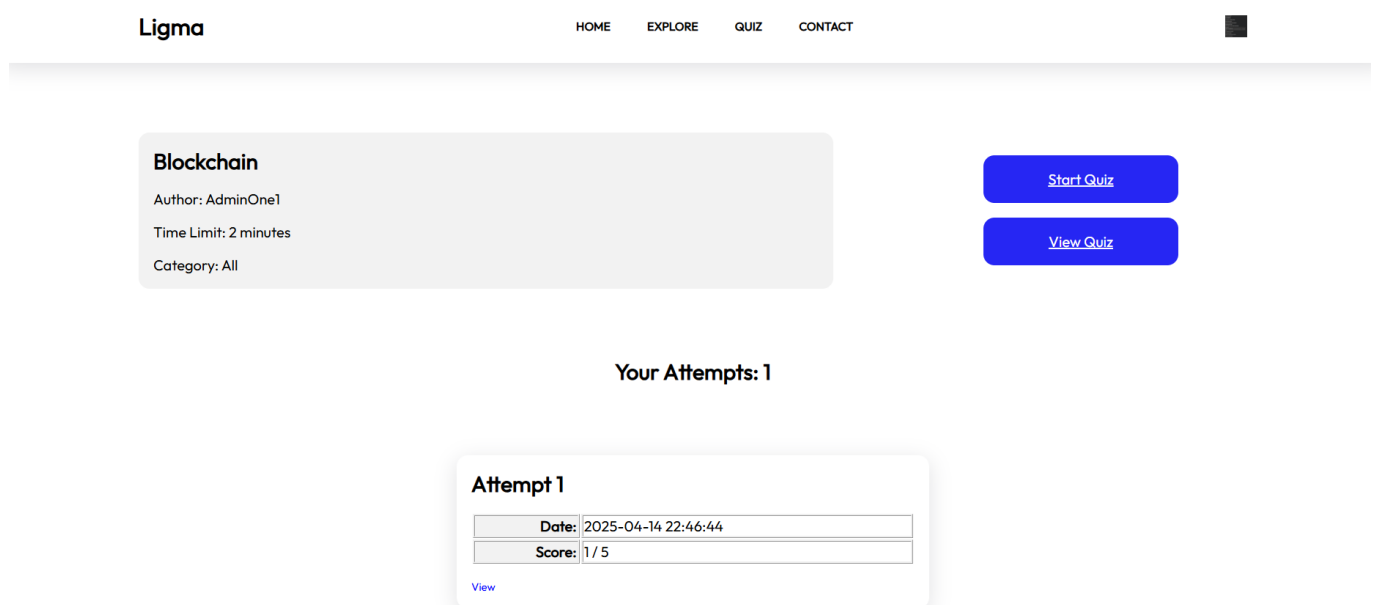
If a user is new to the website, they can click the **Register Now** link to access the sign-up section. In this section, the user is required to provide their **Username**, **Email**, and **Password** to create an account.



The sign-up form is titled "Let's sign you up!". It contains four input fields: "Username", "Email", "Password", and "Confirm password". Below the "Password" field is a checkbox labeled "See password". A blue "Sign up" button is positioned below the "Confirm password" field. Below the button is the word "or". At the bottom, there is a link "Have account? Login Now".

Figure 15: Sign Up Section

Users and guests both have the permission to view the display quizzes and categories provided by admin. However, guests can only view the quiz while users can not only view the quiz but also take the quizzes and view them after each attempts.



The quiz detail page for a user shows the quiz title "Blockchain" and its details: "Author: AdminOne1", "Time Limit: 2 minutes", and "Category: All". To the right of the details are two blue buttons: "Start Quiz" and "View Quiz". Below the details, it says "Your Attempts: 1". Underneath, there is a section for "Attempt 1" with a table showing the date and score.

Attempt 1	
Date:	2025-04-14 22:46:44
Score:	1 / 5

Below the table is a blue "View" link.

Figure 16: Quiz Detail Page (User)

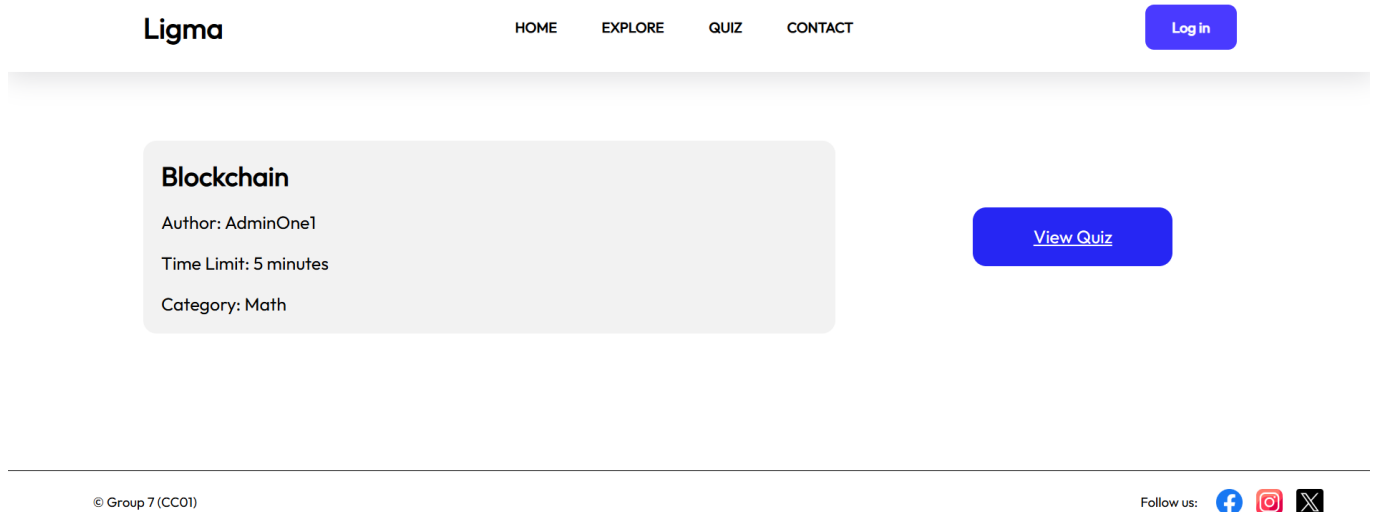


Figure 17: Quiz Detail Page (Guest)

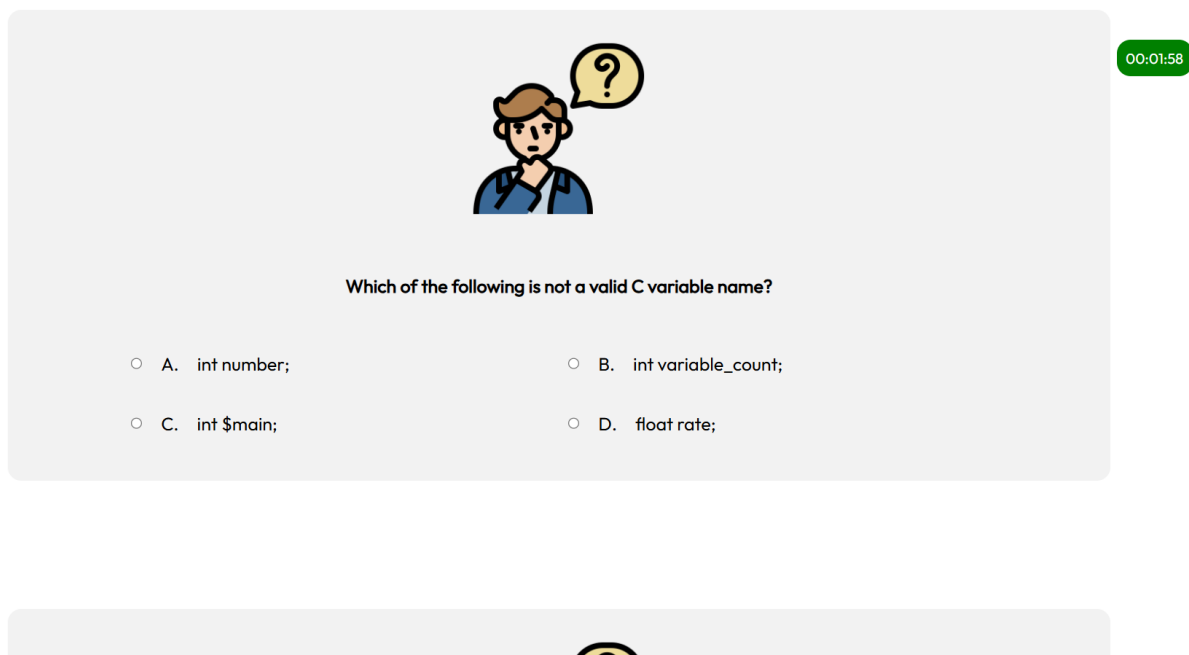


Figure 18: Quiz Detail Page (Guest)

By using CSS in combination with JavaScript, the homepage, authentication pages, and test-taking page are designed to be interactive, visually appealing, and responsive across different devices. This combination allows for dynamic content updates, smooth transitions, and user-friendly interactions throughout the website.

## 5.2 Category selection page before starting a test

On the MCQ website, users have access to a wide variety of quizzes created by admins to test their knowledge or enhance their learning. Users can navigate to the **Explore** section to browse quizzes by category, or visit the Quiz page to view all available quizzes. The **Quiz** page also includes a search bar and sorting options, making it easier for users to find specific quizzes based on their interests or needs.

### Categories

Database

All

Math

Algorithms

### All

[View all](#)

Singthum

Time allowed:  
30

Random

Time allowed:  
10

Random  
test

Time allowed:  
10

Computer  
science

Time allowed:  
5

Random  
knowledge

Time allowed:  
15

Random  
knowledge  
2

Time allowed:  
5

Math  
easy

Time allowed:  
5

Figure 19: Explore Page

### 5.3 Question display with answer selection functionality

The test-taking page is designed to display each question along with an image, the question text, and four answer choices. Each question includes a "select one" functionality, allowing users to choose only one answer from the given options.

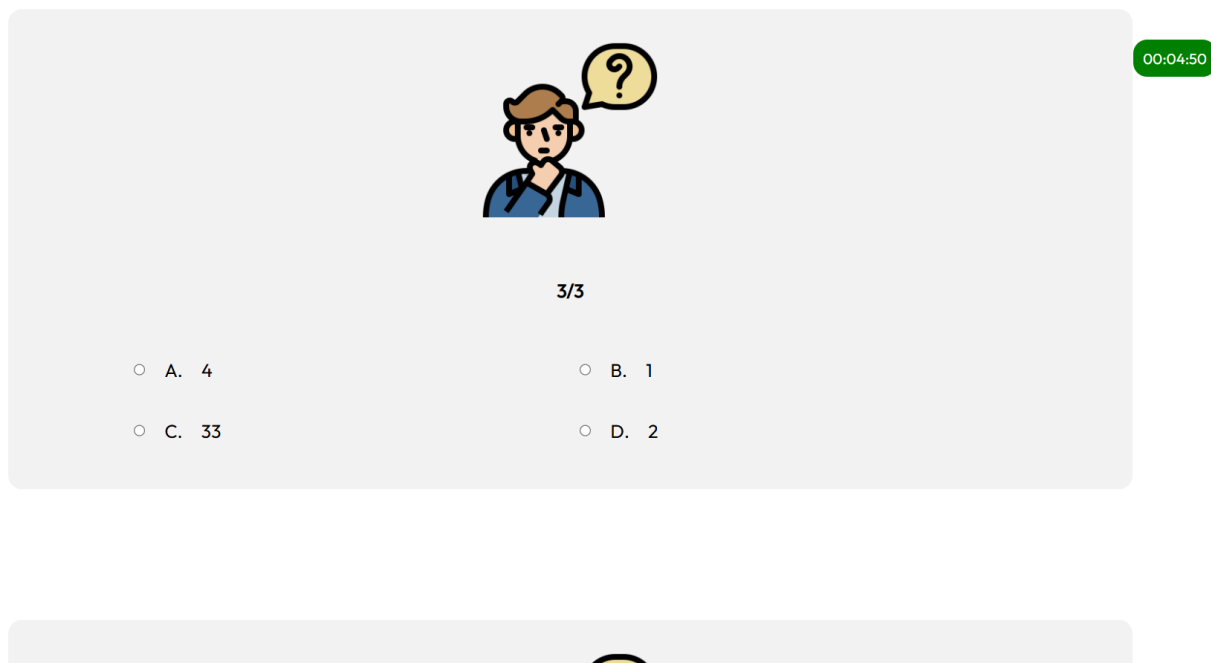


Figure 20: A section with question display with answer selections (Test taking page)

### 5.4 Timer countdown on test page

As shown in Figure 20, a countdown timer appears on each quiz, indicating the time remaining to complete it. Users must submit their answers before the timer reaches zero, or the system will automatically submit the quiz and display

the result once time is up. When there are only 60 seconds remaining, the countdown changes to red, alerting the user to finish the quiz as quickly as possible.

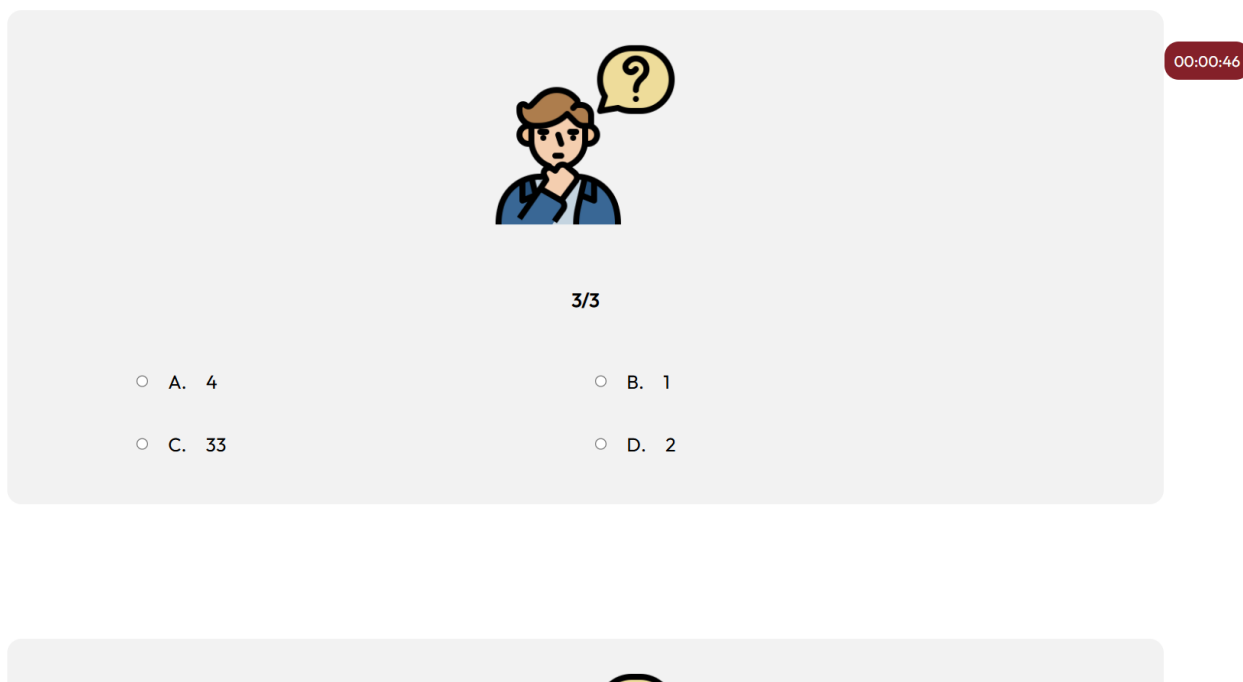


Figure 21: Timer turns red when under 60 seconds left

## 5.5 Image display with each question

As shown in Figures 18, 20, and 21, if a question does not have an image provided by the admin, a default image is displayed in its place. Conversely, when an image is uploaded for a question, it automatically replaces the default image.

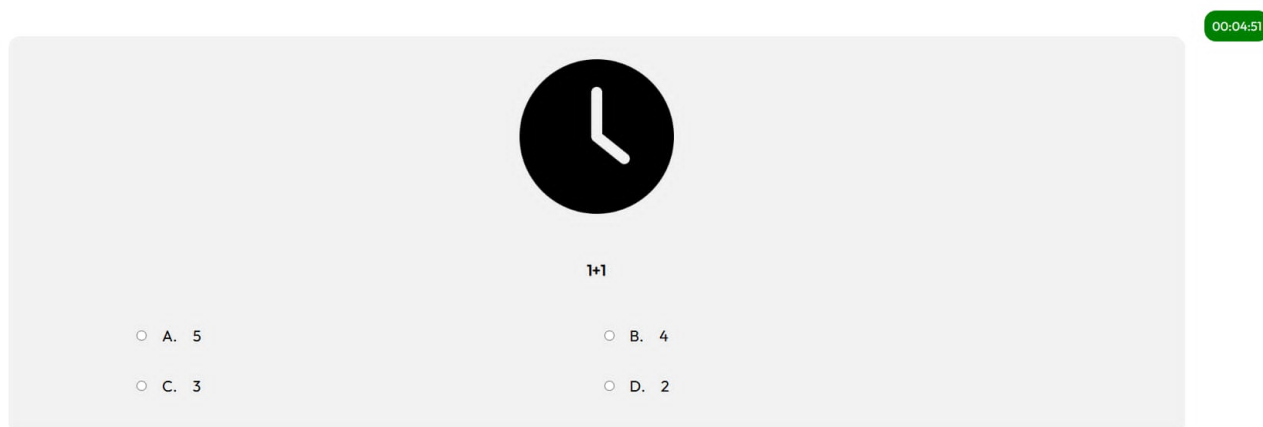


Figure 22: Question with the uploaded image

## 6 Test Functionality Implementation

### 6.1 Functional test-taking feature (answer selection and submission)

The testing function is a critical feature of the website, and its implementation is designed to be straightforward. For each test, if a user is logged in, the "Start Quiz" button is displayed, allowing them to take the test. However, for guests who are not logged in, only the "View Quiz" option is available. The "View Quiz" feature permits guests to view the test content but does not allow them to submit answers.

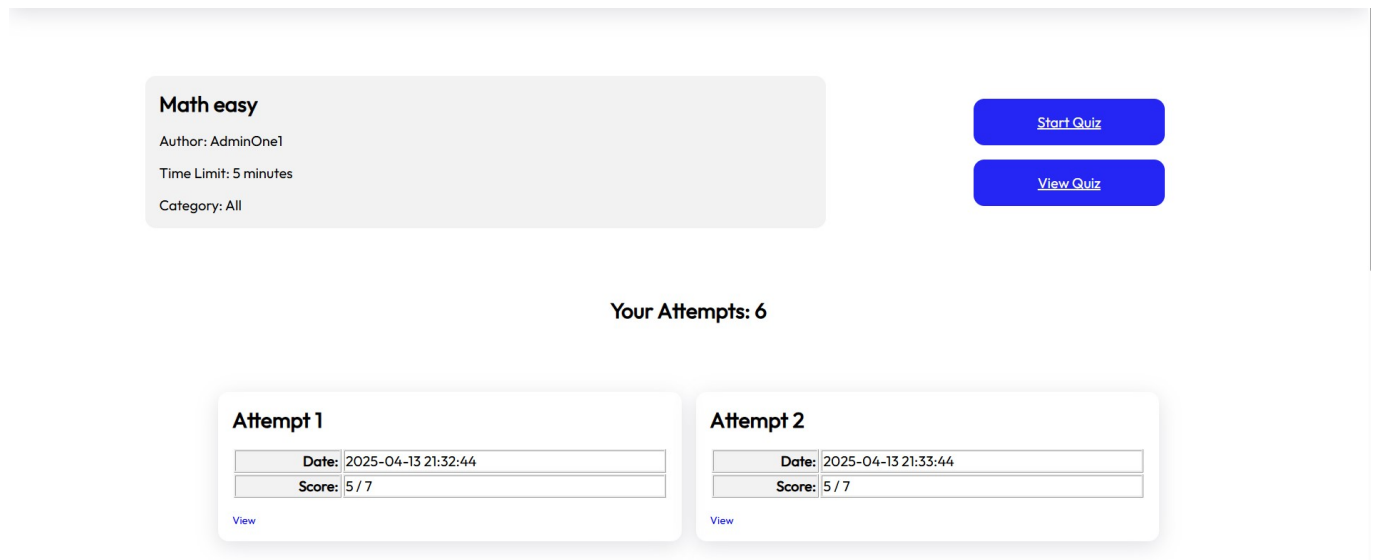


Figure 23: Test information for students

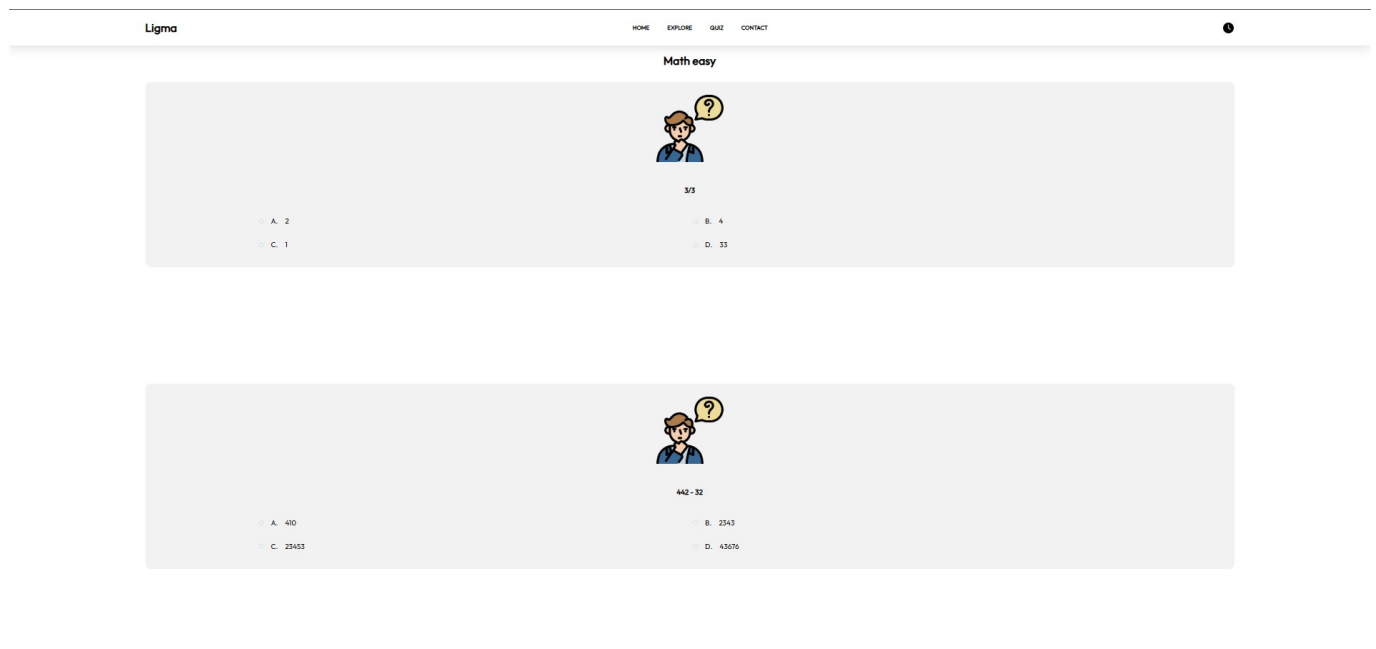


Figure 24: Test view for guests and users

The test tab for students is designed similarly to the test tab for admins, with the addition of a direct link to the test itself. When users take a test, they are directed to the test-taking page, which displays the questions along with their answer choices. A countdown timer is also shown on the page. The test is automatically submitted when the timer runs out, or when the user clicks the "Submit" button. Any questions left unanswered are marked as incorrect.

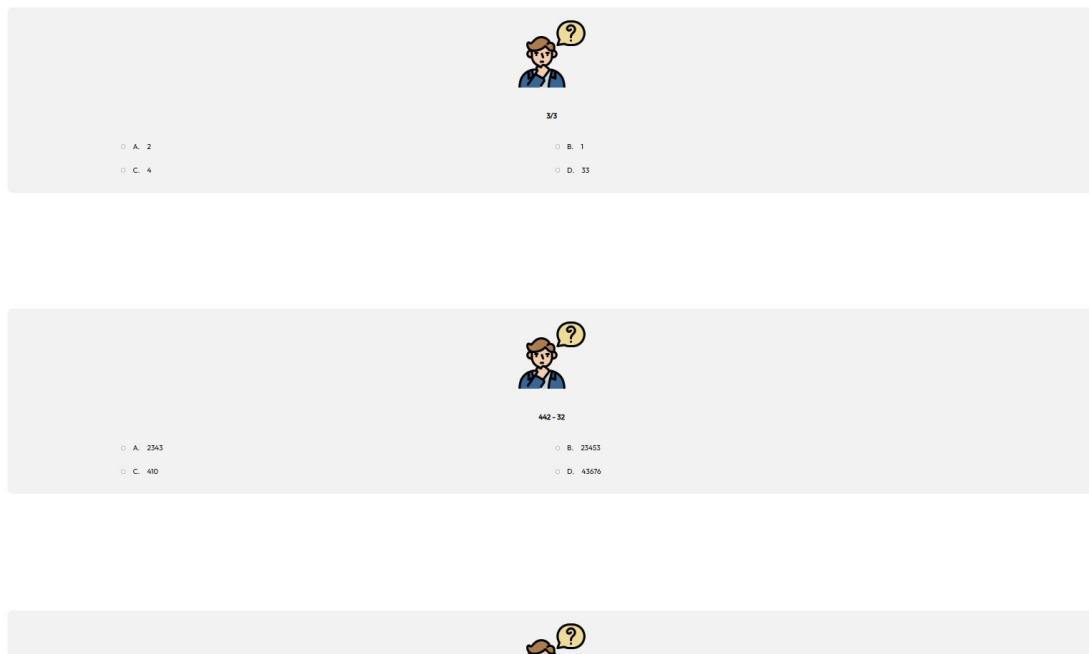


Figure 25: Test taking process for student

## 6.2 Store user test attempts in the database and allow users to review past tests.

Once the test is submitted, the system compares the user's responses to the correct answers to calculate the score. Additionally, below the test information, the system displays the number of attempts and a summary of each attempt for users who have taken the test before. For each attempt overview, there is a link to review the user's attempt. The review does not display the user's specific answers; instead, it indicates which questions were answered correctly or incorrectly. Correct answers are highlighted in green, while incorrect answers are marked in red.

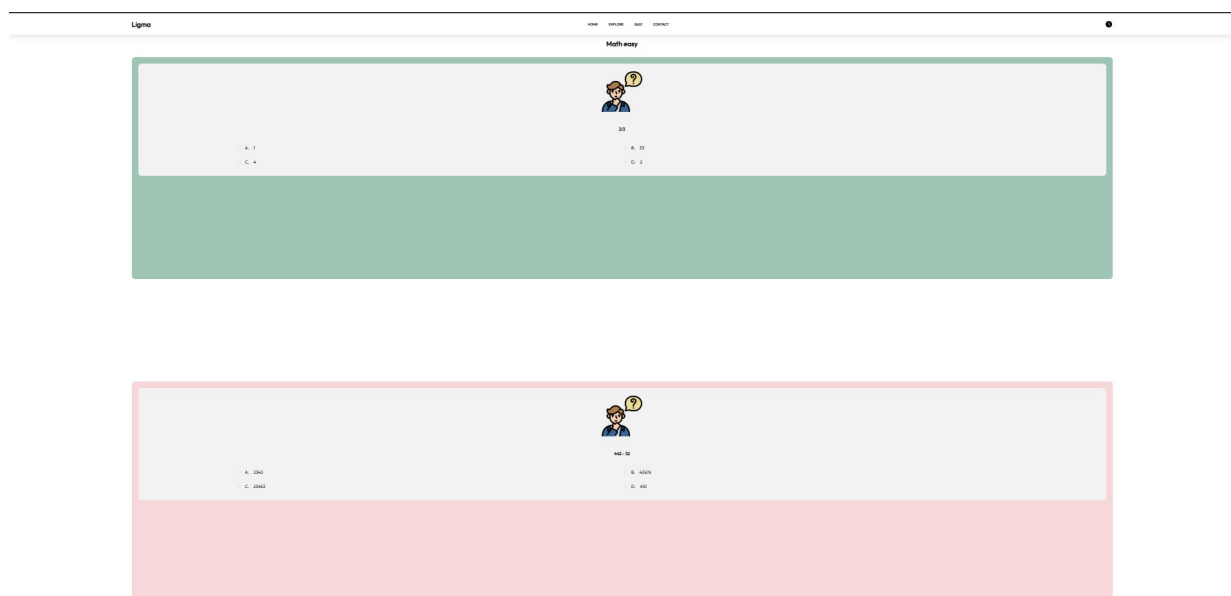


Figure 26: Attempt review for student





## 7 User manual

### 7.1 Guest

#### To log in:

1. Select log in on the upper right of the screen
2. Field out the input
3. Click submit

#### To sign up:

1. Select log in on the upper right of the screen
2. Select register now at the bottom
3. Field out the input
4. Click submit

#### To view test:

1. Select quiz tab
2. Search/sort and select any quiz tab you want to view
3. Click on view quiz

#### To sent contact:

1. Select contact tab in header
2. Field out the input include email, title and message
3. Click submit

### 7.2 Student

#### To take test:

1. Select quiz tab in header
2. Search/sort and select any quiz tab you want to view
3. Click on start quiz
4. Click submit or let the clock run out when finish

#### To view past attempt:

1. Select quiz tab in header
2. Search/sort and select any quiz tab you want to view
3. Click on view attempt link (if there is one) at the bottom of the attempt card

#### To change profile picture:

1. Click on profile picture at top right
2. Click on select picture
3. Choose any picture you want and click save changes

#### To change password:

1. Click on profile picture at top right

2. Click on reset password
3. Choose a new password and type it 2 times in 2 field
4. Click save changes

**To log out:**

1. Click on profile picture at top right
2. Click on log out

## 7.3 Admin

**To create questions:**

1. Click on your tests or your questions in header
2. Click on the + in bottom right of the screen
3. Choose add questions
4. Field out the input and submit

**To create tests:**

1. Click on your tests or your questions in header
2. Click on the + in bottom right of the screen
3. Choose add tests
4. Field out the input and choose next steps
5. Choose questions that you want to appear in the tests and click submit

**To manage question:**

1. Click on your questions in header
2. Search/sort and choose questions you want to manage
3. Choose edit or delete questions

**To manage students:**

1. Click on students in header
2. Search/sort and choose the students you want to manage
3. Choose ban to ban the student or activate to allow student to access the website again

## 8 How to set up the website

1. Install and setup XAMPP [through this links](#) or or go to the next url: <https://drive.google.com/file/d/1ttqLfJJmGcsZQckl9YW4lUSS3gX2scE/view?usp=sharing>
2. Download the zip and extract it in htdocs folder or fork and clone the project through [the github link](#) to the htdocs folder
3. Open <http://localhost/phpmyadmin/> and create a database name mcq
4. Navigate to the sql folder in the project, locate the mcq.sql file
5. In <http://localhost/phpmyadmin/>, click in the mcq database and click import tab to import the mcq.sql file
6. Navigate links to pages folder to go to index.php file
7. Use the following account with email and password to use the website:
  - user1@example.com, \$2y\$10\$f1M@.YHash3dPa\$\$w0rd1
  - user2@example.com, \$2y\$10\$f1M@.YHash3dPa\$\$w0rd2
  - user3@example.com, \$2y\$10\$f1M@.YHash3dPa\$\$w0rd3
  - user4@example.com, \$2y\$10\$f1M@.YHash3dPa\$\$w0rd4
  - user5@example.com, \$2y\$10\$f1M@.YHash3dPa\$\$w0rd5
  - user6@example.com, \$2y\$10\$f1M@.YHash3dPa\$\$w0rd6
  - admin1@example.com, \$2y\$10\$f1M@.YHash3dPa\$\$w0rdAdmin1
  - admin2@example.com, \$2y\$10\$f1M@.YHash3dPa\$\$w0rdAdmin1

## 9 Conclusion

Completing the assignment has been an enriching experience, allowing my team to deepen my understanding of web development. I successfully implemented beginner-level CRUD features, which have strengthened my practical knowledge and skills in creating, reading, updating, and deleting data within web applications. Some of the features includes:

1. Sign in / Sign up
2. Sessions
3. Hashing password
4. Sanitize input
5. Dynamic display test and questions
6. AJAX based search
7. Sorting
8. Edit profile, question details
9. Take test, review attempts
10. Add test, questions
11. Post contact information
12. Display test and questions detail dynamically

Due to time constraints and the workforce of only 2 persons, there were certain tasks that our team was unable to complete before the deadline. Some of these include:

1. Testing Report
2. Final working web application deployed on a server.
3. Cookies



## 10 References

1. W3 Schools PHP tutorial. <https://www.w3schools.com/php/>.
2. Nixon, R. (2014). Learning PHP, mysql, JavaScript, CSS & HTML5: A step-by-step guide to creating dynamic websites. O'Reilly.
3. Elmasri, R., & Navathe, S. (2020). Fundamentals of Database Systems. Pearson.