# Development Documents for Our Prototype

## UI/Front-end

### Bootstrap

Takes you to sign up page

### Homepage

Graphical user interface, text

Description automatically generated

Takes you to sign up page

The homepage will act as a landing page for users that are not logged in or do not have an account. It will show basic site information and graphics explaining what the company is about and why somebody may choose them. I have used lorem ipsum auto generated text to act as placeholders for the appropriate writing (this is common theme among the rest of the site).

I have used the Z pattern design for this landing page to ensure users will look at what we want them to look at. This makes them more likely to read information and, in turn, sign up to our clients website. More on this here (<https://instapage.com/blog/z-pattern-layout>).

### Sign-up

Graphical user interface, application

Description automatically generated

Takes you to login page

Takes you to dashboard

This is the sign-up page that a user will be presented with when choosing to sign in or “get started” on the homepage. Here you can see that a user will need to fill in all the appropriate details and then select whether they want to sign up as a teacher or a student.

Upon completion, the appropriate account will be created and inserted into the database and the user will be taken to their dashboard depending on their account type chosen.

A user may also click “Already have an account?” to take them to the login page.

### Graphical user interface, text, application Description automatically generatedLogin

Takes you to dashboard

Takes you to sign-up page

From the log in page, you can log into a teacher or a student account. You may also navigate to the sign-up page by clicking “Don’t have an account?”.

A user may also check the “Stay logged in?” box to set a cookie that will remember the user for 2 months (easily variable to client’s needs). Failing to click this means they will have to log back in manually after the session ends.

We’ve also included a “Forgot your password” option. This, however, is not functional in this prototype as we felt it was not a necessary feature for the time being and can be added later with more time on the project.

### Student Dashboard

#### Unfiltered

Graphical user interface, application

Description automatically generated

Allows quiz/course searching

Allows filtering

Takes you to quiz

Takes you to course

Goes to account page

This is the student dashboard, it’s what you see once you have successfully logged in as a student. It acts as a central point for all tasks that a student is able to complete.

From here you can:

* View available courses and quizzes
* Enrol in a course
* Take a quiz
* Filter for courses or quizzes
* Search for courses or quizzes
* Navigate to your account page.

#### Courses Filter

Graphical user interface, application, Teams

Description automatically generated

#### Quiz Filter

Graphical user interface, application

Description automatically generated

#### Search Term

Graphical user interface, application

Description automatically generated

### View Course

Text

Description automatically generated

Back to dashboard

This is the course page. You can see the course title, subtitle, and content.

Again, this is a prototype and as such we’ve used lorem ipsum because actual content is expected to be provided by our clients themselves.

This page in future iterations may include images, hyperlinks, embedded YouTube videos etc.

### View Quiz

Graphical user interface, text, application, website

Description automatically generated

Takes to quiz submit page and shows score.

Back to dashboard

Above you can see the quiz page. This is the template used for any quiz being taken. The example above is a very basic maths quiz to demonstrate functionality.

You can see:

* The quiz name
* Each question number and title
* Three labelled multiple-choice options for each question
* A quiz submit button

The quiz will not be submitted unless all questions have an option selected.

### Quiz landing page

Text, timeline

Description automatically generated

As you can see, this page is very simple. It’s primary purpose to the user is to simply show them what they scored in the quiz and provide means of navigating back out of the page.

However, from a back-end point of view (more on this at [Back-end/DB](#_Back-end/DB)), this is where the quiz is marked, score is calculated, reward points calculated, and reward points and quiz results are inserted into the database for storage.

### Account

Graphical user interface

Description automatically generated

Logs out and goes back to homepage

The account page is a very important page for a student, from here you can:

* View how many reward points you have
* View the courses that you are enrolled on
* View quiz results for any quizzes that you’ve taken
* View your personal details
* Log out

This page is vital in a website such as this because it allows students to see their progress and how they’re getting along. This acts as a motivator for them to learn more (this is particularly true in the case of reward points).

### Teacher Dashboard

Graphical user interface, chart

Description automatically generated

This is the page you land on after logging in with a teacher account. This page allows a teacher to accomplish any of the tasks for their use case by navigating to the appropriate section with the large central buttons.

Here, a teacher can:

* Navigate to monitoring their students’ progress
* Navigate to quiz creation page
* Navigate to course creation page
* Log out

The features listed above are enough to cover all of a teacher’s requirements for this project.

### A picture containing graphical user interface Description automatically generatedStudent Monitor

This page is vital for teachers to be able to monitor their students’ progress and find and help those who may be lagging behind.

Here you can select a course from the dropdown list that is automatically populated with any courses that the currently logged in teacher has created. The quiz dropdown works in the same way.

Selecting a course or quiz will show you all students who have enrolled on that course as well as showing all quiz results for a selected quiz.

### Create Quiz

Graphical user interface, application, table

Description automatically generated

Select number of questions

Creates quiz and takes to quiz created page

This is the quiz creation page. From here you can select the number of questions you want on the quiz and the page will change accordingly. You can enter a question title and all three options as well as selecting the option that contains the correct answer (correct answers are stored as a list in the DB).

### Quiz Created

Graphical user interface, text, timeline

Description automatically generated

To help provide a good user experience, this page provides visual feedback and confirmation that the quiz has been successfully created.

This is also where lots of background processes get done such as inserting the quiz into the database.

### Graphical user interface, application Description automatically generatedCreate Course

The course creation page allows a teacher to create any course they want. It allows input of a course name, course description, subtitles, and paragraphs. Currently the prototype supports only one subtitle and paragraph but this is a feature that could be implemented in the future relatively easily.

All fields must be completed before creating the course.

Upon clicking “Create course”, user is sent to the “Course Created” page.

### Graphical user interface, text, timeline Description automatically generatedCourse Created

To help provide a good user experience, this page (like the quiz created page) provides visual feedback and confirmation that the course has been successfully created.

This is also where background processes get done such as inserting the course into the database.

## Back-end/DB

### Database Creation

Diagram

Description automatically generated

The image above is the relational database we designed in the design phase of this project. However, you will see that there were some oversights, and the database has now been changed in a few key ways that will be highlighted later

**Table creation code**

Some screenshots below are also outdated and created before the database changes. Where the current database differs, I will highlight and show the updated version.

Text

Description automatically generatedText

Description automatically generatedText

Description automatically generatedText

Description automatically generatedText

Description automatically generated with medium confidenceText

Description automatically generated with low confidenceGraphical user interface, text

Description automatically generated with medium confidence

Now: quizContent TEXT NOT NULL

Reason: VARCHAR(255) not long enough and was cutting of end of content

Now: courseContent TEXT NOT NULL

Reason: VARCHAR(255) not long enough and was cutting of end of content

Table “enrols” (see above) has now been reconstructed to include a teacherId column

Table “quizzes” (see above) has now been reconstructed to include quizAnswers column and a teacherId column

Each table is linked to at least one other (this creates a relational database)

Each table has an Id from another table in order to relate to it

Each table has an Id set to auto\_increment and set as a primary key

Sector table (see above) is redundant in this prototype as it is not used. In future versions it can be used to filter courses/quizze depending on sector

## Solution Code

In this section we will be sharing screenshots of some of the main code files, both front and back end, and briefly explaining them and their purpose.

Some files include large line counts of 100+ and there are many files to be seen, particularly in the function library, therefore not all files will be covered in full.

To see all code produced for this solution, check for the Code folder in the submission folder.

### Libraries

The code under the “libraries” heading is all libraries of either PHP or Javascript functions that are used by the [main](#_Main) code files and contain no front-end html.

#### course\_listings.php (dashboard population)

Populate\_courses function which adds all available courses onto the student dashboard

Function to create the cards which house the data coming from the below populate functions

|  |
| --- |
| <?php      include\_once "C:/xampp/htdocs/Projects/GibJohn/libraries/db.php";      //function that creates a student dashboard card given the appropriate arguments      function create\_card($name, $description, $id, $type){          if ($type ==  'Quiz'){              $button\_name = 'Take quiz';          }          else{              $button\_name = 'Join Course';  Echos a html card which is a bootstrap element          }          echo'          <div class="card" style="display:inline-block; width:20vw; height:30vh; margin:10px;">          <h5 class="card-header">'.$type.'</h5>          <div class="card-body">              <h5 class="card-title">'.$name.'</h5>              <p class="card-text" style="height:10vh;">'.$description.'</p>              <form method="POST" action="'.$type.'\_template.php">              <input type="hidden" name="hiddenId" value="'.$id.'"/>              <button class="btn btn-primary" type="submit">'.$button\_name.'</button>              </form>          </div>          </div>';      }      //function that populates student dashbaord with courses. Calls above create\_card function      function populate\_courses()      {            $conn = get\_conn();          $result = mysqli\_query($conn, "            SELECT courseName, courseDescription, courseId FROM courses          ");          while ($row = $result->fetch\_assoc()) {                create\_card($row['courseName'], $row['courseDescription'], $row['courseId'], 'Course');          }        }      //function that populates student dashbaord with quizzes. Calls above create\_card function      function populate\_quizzes()      {            $conn = get\_conn();          $result = mysqli\_query($conn, "            SELECT quizName, quizDescription, quizId FROM quizzes          ");          while ($row = $result->fetch\_assoc()) {                create\_card($row['quizName'], $row['quizDescription'], $row['quizId'], 'Quiz');          }        } |

Loops through every query result and turns it into a string and makes a card from it

Populate\_quizzes function which adds all available quizzes onto the student dashboard

Above is a snippet of the code from course\_listings.php, as such, this is not the whole file. I’ve decided to show this part as it displays some key functions that are used frequently or are very important the functionality of the website.

Mainly the functions populate\_courses and populate\_quizzes work by querying the database for courses or quizzes respectively. They then pass their values into the create\_card function which will echo the card onto the php page where it is called.

#### Db.php (database retrieval/insertion)

|  |
| --- |
| //handy function to return the connection to my database easily.  function get\_conn(){  Function that creates a connection to the database. Used commonly among entire project      $hostname = "localhost";      $username = "root";      $password = "";      $dbname = "GibJohn";      $conn = mysqli\_connect($hostname, $username, $password, $dbname) or die ("Could not connect to database.");      return $conn;  Functions that return the student and teacher IDs respectively  }  //handy function to return the student id of the current user  function get\_id(){      $email = unserialize($\_SESSION["user"])[0];      $user\_id = current(get\_conn()->query("SELECT studentId FROM students WHERE email = '".$email."'")->fetch\_assoc());      return $user\_id;  }  //handy function to return the teacher id of the current user  function get\_id\_teacher(){      $email = unserialize($\_SESSION["user"])[0];      $user\_id = current(get\_conn()->query("SELECT teacherId FROM teachers WHERE email = '".$email."'")->fetch\_assoc());      return $user\_id;  Function to return reward points for user. Could be improved by calling get\_id() instead of querying with the email  }  //returns the number of reward points that the current user has  function get\_reward\_points(){      $email = unserialize($\_SESSION["user"])[0];      $reward\_points = mysqli\_query(get\_conn(), "SELECT rewardPoints FROM students WHERE email = '".$email."'");      while ($row = $reward\_points->fetch\_assoc()) {          echo $reward\_points\_new = $row['rewardPoints'];      }        return $reward\_points\_new;  }  //function to echo personal details onto the the screen. Used for account page  function my\_details(){      $conn = get\_conn();      $user\_id = get\_id();        $query = "SELECT \* FROM students WHERE studentId = '".$user\_id."'";      $result = mysqli\_query($conn, $query);  Function to print user details. Can be optimised by selecting only the required fields instead of all.      while ($row = $result->fetch\_assoc()) {          echo 'Email: '.$row['email'].'</br>';          echo 'First Name: '.$row['fname'].'</br>';          echo 'Second Name: '.$row['sname'].'</br>';      }    }  //Echos all of the courses that the current user is enroled on. Used for account page.  function get\_courses(){        $query = " SELECT courseName FROM enrols INNER JOIN courses ON enrols.courseId = courses.courseId WHERE studentId = '".get\_id()."'";      $result = mysqli\_query(get\_conn(), $query);  Get courses or quizzes by querying the tables using the current user ID      while ($row = $result->fetch\_assoc()){          echo $row['courseName'];          echo '<br/>';      }  }  //Echos all of the quizzes and results that the current user has. Used for account page.  function get\_quizzes(){      $query = "SELECT quizName, quizResult, dateOfCompletion FROM quizResults INNER JOIN quizzes ON quizResults.quizId = quizzes.quizId WHERE studentId = '".get\_id()."'";      $result = mysqli\_query(get\_conn(), $query);      while ($row = $result->fetch\_assoc()){          echo $row['quizName']. ' : '. $row['quizResult'] ;          echo '<br/>';      }  } |

Print each result neatly and on a new line by looping through each result

The code from db.php (shown above) is a library of functions that mostly houses easy ways of retrieving information from the database such as functions to retrieve a user’s ID or their personal details or courses taken etc.

The functions in this file are used commonly all across the site and so their functionality and efficiency is essential.

#### Lib.js (changing page elements through code)

Get each element by their id and manually change their class name.

|  |
| --- |
| //Changes the highlighted option depending which one is last clicked  function change\_class(id){      document.getElementById('addQuestion5').className = "btn btn-primary";      document.getElementById('addQuestion10').className = "btn btn-primary";      document.getElementById('addQuestion15').className = "btn btn-primary";      document.getElementById('addQuestion20').className = "btn btn-primary";      document.getElementById(id).className = "btn btn-outline-primary";  }  //changes how many questions are displayed on quiz creation page depending on selected option  function add\_question(loop\_num){    This function selects the span by its ID. Then inserts into it a defined amount of html depending on the input parameter        var mySpan = document.getElementById('mySpan');      mySpan.innerHTML = "";      count = 1;      for (let i = 0; i < loop\_num; i++) {          var question = '<input type="hidden" name="qAmount'+loop\_num+'"/><label for="q'+count+'">Q'+count+': Title</label><input type="text" placeholder="What is cat in spanish?" id="q'+count+'" name="q'+count+'" class="form-control" required/><br/><p>Please select the option with the correct answer</p><input type="radio" name="radioq'+count+'" id="radioq'+count+'option1" style="float:left; margin-right: 1vw; margin-left:1vw;" class="form-check-input" required/><input style="width:30%" type="text" placeholder="Option 1" id="q'+count+'option1" name="q'+count+'option1" class="form-control" required/><input type="radio" name="radioq'+count+'"  id="radioq'+count+'option2" style="float:left; margin-right: 1vw; margin-left:1vw;" class="form-check-input"/><input style="width:30%" type="text" placeholder="Option 2" id="q'+count+'option2" name="q'+count+'option2" class="form-control" required/><input type="radio" name="radioq'+count+'"  id="radioq'+count+'option3" style="float:left; margin-right: 1vw; margin-left:1vw;" class="form-check-input"/><input style="width:30%" type="text" placeholder="Option 3" id="q'+count+'option3" name="q'+count+'option3" class="form-control" required/><br/>';          mySpan.innerHTML += question;          count += 1;      }  }  //checks which is selected and loops through every option, submits form  function add\_values(){      if( document.getElementById("addQuestion5")){  This function determines which option is selected by checking which has a certain class. It then submits the form with a variable count          if (document.getElementById("addQuestion5").class == "btn btn-outline-primary"){              count = 5;          }      }      if( document.getElementById("addQuestion10")){          if (document.getElementById("addQuestion10").class == "btn btn-outline-primary"){              count = 10;          }      }      if( document.getElementById("addQuestion15")){          if (document.getElementById("addQuestion15").class == "btn btn-outline-primary"){              count =15;          }      }      if( document.getElementById("addQuestion20")){          if (document.getElementById("addQuestion20").class == "btn btn-outline-primary"){              count = 20;          }      }      for (let i = 1; i < count; i++) {          for (let x= 1; x < 4; x++){              if (document.getElementById('q'+i+'option'+x).value){                    document.getElementById('radioq'+i+'option'+x).value = document.getElementById('q'+i+'option'+x).value;                    var quizForm =  document.getElementById("quizForm");                  quizForm.action = 'create\_quiz\_action.php';                  quizForm.submit();              }              else{                  break;              }            }      }    } |

Lib.js is the only Javascript library we have created for this prototype. It houses a few relatively simple functions.

These functions are for changing the content of the html page upon button clicks which is most easily achieved in Javascript and so that’s that language we chose to use.

#### Login.php (handles logging in)

|  |
| --- |
| <?php  include\_once "C:/xampp/htdocs/Projects/GibJohn/libraries/db.php";  //function to check if entered login details are valid  function check\_login($email,$password){        $conn = get\_conn();      $email\_search = "SELECT \* FROM students WHERE email = '".$email."'";      $result = mysqli\_query($conn, $email\_search);      //email found in students table      if (mysqli\_num\_rows($result) > 0)      {          $result = mysqli\_query($conn, "SELECT pword FROM students WHERE email = '".$email."'");          while($row = mysqli\_fetch\_object($result)){              $pword = $row->pword;           }          if (password\_verify($password, $pword ))          {              return "student";          }          else{              return "Password Incorrect";          }        }      //email not found in students table      else      {          $email\_search = "SELECT \* FROM teachers WHERE email = '".$email."'";          $result = mysqli\_query($conn, $email\_search);          //email found in teachers table          if (mysqli\_num\_rows($result) > 0)          {              $result = mysqli\_query($conn, "SELECT pword FROM teachers WHERE email = '".$email."'");              while($row = mysqli\_fetch\_object($result)){                  $pword = $row->pword;              }              if (password\_verify($password, $pword ))              {                  return "teacher";              }              else{                  return "Password Incorrect";              }            }          //email not found in either table          else          {              return "Email not found!";          }      }  }  ?> |

The login.php file (see above) is responsible for logging you in after entering details on the login form.

Here is the process in simple, non-technical steps:

* It first searches to see if it can find the email already in the students table
* Failing the above, it will search the teacher table
* After finding an email in either table it will verify that the password entered is the same as the one associated with the email entered.
* If the email is not found, “Email not found” will be returned.
* If the email is found but password does not match, “Incorrect password” will be returned.
* If all details are corrected, you will be sent to either the teacher or student dashboard depending on which table your email was found in.

We search the students table first because there will be more students than teachers and therefore the likelihood of a user being a student is higher. This means the result will usually be found in the students table so the code is more efficient most of the time.

#### quiz\_results.php (calculating and storing score)

|  |
| --- |
| <?php      include\_once "C:/xampp/htdocs/Projects/GibJohn/libraries/db.php";      //calculates score depending on if POSTED answers are equal to correct answers      function calculate\_score($answers\_arr){  Takes answer array as a parameter and checks every answer against each selected radio option.          $score = 0;          $loop = true;          $i = 0;          $x = 1;          while ($loop){                if (isset($\_POST['radioq'.$x])){                  if ($\_POST['radioq'.$x] == $answers\_arr[$i]){$score += 1;}              }              else{                  $loop = false;              }              $i += 1;              $x += 1;          }          return $score;      }  Turns score into a percentage depending on total questions (parameter). Then inserts into DB      //stores the score into the quizResults table      function store\_score($score,$total,$quizId){          $percentage = ($score / $total) \* 100;          $percentage\_str = strval($percentage).'%';          mysqli\_query(get\_conn(), "INSERT INTO quizResults (studentId, quizId, quizResult, dateOfCompletion) VALUES ('".get\_id()."','".$quizId."','".$percentage\_str."','".date("Y-m-d")."') ");          add\_reward\_points($percentage);      }  ?> |

Quiz\_results.php does what it says on the tin. This file houses functions that handle the results of a quiz. Here, the score is calculated, turned into a percentage, and stored in the database.

#### Signup.php (handles signing up)

|  |
| --- |
| <?php  //signs up a new user with all of their details  function sign\_up($conn, $email, $password, $fname, $sname, $account\_type)  {  Hashes the inputted password      $password = password\_hash($password, PASSWORD\_BCRYPT);      $token = md5(rand().time());      //if they select they want to sign up as student  Handles student sign up      if ($account\_type == "student")      {          //does email already exist in DB?          $email\_search = "SELECT \* FROM students WHERE email = '".$email."'";          $result = mysqli\_query($conn, $email\_search);          //check if email exists          if (mysqli\_num\_rows($result) > 0)          {              return "Email Taken";          }          //insert new account into database          elseif (mysqli\_num\_rows($result) == 0)          {              $sql = "              INSERT INTO students(              email,              fname,              sname,              pword,              rewardPoints,              token              ) VALUES (                '{$email}',              '{$fname}',              '{$sname}',              '{$password}',              0,              '{$token}'              )              ";              //create mysql query              $sqlQuery = mysqli\_query($conn, $sql);              if (!$sqlQuery)              {                  die ("MySQL query failed!" . mysqli\_error($conn));                  return "Query Failed!";              }  Handles teacher sign up              return true;          }      }      //if they select they want to sign up as teacher      else      {          //does email already exist in DB?          $email\_search = "SELECT \* FROM teachers WHERE email = '".$email."'";          $result = mysqli\_query($conn, $email\_search);          //check if email exists          if (mysqli\_num\_rows($result) > 0)          {              return "Email Taken";          }          //insert new account into database          elseif (mysqli\_num\_rows($result) == 0)          {              $sql = "              INSERT INTO teachers(              email,              fname,              sname,              pword,              token              ) VALUES (                '{$email}',              '{$fname}',              '{$sname}',              '{$password}',              '{$token}'              )              ";              //create mysql query              $sqlQuery = mysqli\_query($conn, $sql);              if (!$sqlQuery)              {                  die ("MySQL query failed!" . mysqli\_error($conn));                  return "Query Failed!";              }              return true;          }      }    }  ?> |

Signup.php (see above) includes only one fairly large function that signs up a user upon completing a sign up form.

It will create a password hash and check for duplicate emails and then process to sign the user up as either a teacher or a student.

It signs them up by inserting user inputs into the correct table on the database along with creating a unique id, password hash, and token.

### Main

All code under this main header will be php pages containing html to display the page itself (with the exception of a few php action files).

This forms the front end of the solution and provides the UI. It’s our hope that these pages provide a satisfying user experience.

To keep this document a reasonable length we’ve decided to strip the repetitive html tags that are not unique in any way document to document. Essentially this means only showing the body tag contents. Again, full code files can be found as PDFs in the Code folder.

Navbar that is almost identical on all pages for intuitive design/consistency

#### Sign Up

Form with inputs for email, fname, sname, password

|  |
| --- |
| <body>      <!--NAV BAR-->      <nav class="sticky-top navbar">      <img src="images/book.png" style="margin-left:48.25%; max-width:55px; border-radius:55px; border: 3.5px solid white; box-shadow:none; max-height: 4vw; " onclick="Location: href='index.php'" />      <a style="float:right; margin-right: 10px;" href="index.php" class="btn btn-danger">Back</a>      </nav>      <!--Sign up form -->      <div style="width:33%; margin:auto;" class="min\_height">      <h1 style="padding-top:20px;">Sign up for an account!</h1>      <form style="padding-top:10px;" action="sign-up-action.php" method="post">          <!--Email-->          <label for="email" class="form-label">Email address</label>          <input type="email" name="email" class="form-control" required/> <br/>          <!--fname-->          <label for="fname" class="form-label">First name</label>          <input type="text" name="fname" class="form-control" required/> <br/>          <!--sname-->          <label for="sname" class="form-label">Second Name</label>          <input type="text" name="sname" class="form-control" required/> <br/>          <!--Password-->          <label for="password" class="form-label">Password</label>          <input type="password" name="password" class="form-control" required /> <br/>  Important option to choose account type          <!-- Student/Teacher radio-->          <p>I am a:</p>          <p>          Student<input type="radio" value="student" name="account\_type" style="margin-left: 10px;" required/>          Teacher<input type="radio" value="teacher" name="account\_type" style="margin-left: 10px;"/>          </p>          <!--Already have an account-->          <a style="font-size:14px;" href="login.php">Already have an account?</a>          <!--Error-->          <p style="margin:0px;padding:0px;padding-bottom:10px;color:red" ><?php if (isset($\_SESSION['sign\_up\_err'])) {echo $\_SESSION['sign\_up\_err']; unset($\_SESSION['sign\_up\_err']); }?></p>          <!--Submit Form-->          <p><button type="submit" class="btn btn-primary">Register</button></p>        </form>      </div>      <!--FOOTER-->      <footer>      <div class="row section " style="margin:0px;">          <div class="col-md-12 footer" style="margin:0;" >              <p style="color:white;"></br>Find us at:</br>                  Instagram: @GibJohn</br>                  Facebook: @GibJohn</br>                  Email: gibjohn@gmail.com</br>                  Phone: 07473820938              </p>          </div>      </div>      </footer>  </body> |

Footer that is identical on all pages for intuitive design/consistency

Some php code embedded in an empty <p> to check if an error has occurred. And show it

#### Log in

Form containing email and password inputs

|  |
| --- |
| <!--NAV BAR-->      <nav class="sticky-top navbar">      <img src="images/book.png" style="margin-left:48.25%; max-width:55px; border-radius:55px; border: 3.5px solid white; box-shadow:none; max-height: 4vw; " onclick="Location: href='index.php'" />      <a style="float:right; margin-right: 10px;" href="index.php" class="btn btn-danger">Back</a>      </nav>      <!--Log in form-->      <div style="width:33%; margin:auto;" class="min\_height">      <h1 style="padding-top:20px;">Please log in</h1>      <form style="padding-top:10px;" action="login-action.php" method="post">          <!--EMAIL-->          <label for="email" class="form-label">Email address</label>          <input type="email" name="email" class="form-control" required/> <br/>          <!--Password-->          <label for="password" class="form-label">Password</label>          <input type="password" name="password" class="form-control" required/> <br/>          <!--Remember me-->          <div style="height:30px; line-height: 30px;">          <label for="remember\_me" class="form-label" style="padding-right: 10px;" >Stay logged in?</label>          <input  type="checkbox" name="remember\_me" value="Yes" />          </div>          <!--Forgot Password-->          <a style="font-size:14px; padding-right:15px;" href="">Forgot your password?</a>          <!--Don't have an account-->          <a style="font-size:14px;" href="sign-up.php">Don't have an account?</a>          <!--Error message-->          <p style="margin:0px;padding:0px;padding-bottom:10px;color:red" ><?php if (isset($\_SESSION['login\_err'])){ echo $\_SESSION['login\_err']; unset($\_SESSION['login\_err']); }?></p>          <button type="submit" class="btn btn-primary">Login</button>        </form>      </div>      <!--FOOTER-->      <footer>          <div class="row section " style="margin:0px;">              <div class="col-md-12 footer" style="margin:0;" >              <p style="color:white;"></br>Find us at:</br>              Instagram: @GibJohn</br>              Facebook: @GibJohn</br>              Email: gibjohn@gmail.com</br>              Phone: 07473820938              </p>              </div>          </div>      </footer> |

Takes you to sign up for an account

Error message that works the same way as in the sign up page

Remember me checkbox to set a 2 month long cookie if checked

#### Homepage

|  |
| --- |
| <?php  session\_start();  Checking for either a cookie or a session that can verify we are already logged in  //assume not logged in  $logged\_in = false;  //check for log in cookie  if (isset($\_COOKIE["remember\_me"]))  {        $\_SESSION["user"] = $\_COOKIE["remember\_me"];      //check if stored user is student or teacher      if (unserialize($\_SESSION["user"])[2] == "student")      {           header("Location: student\_dashboard.php");      }      else      {            header("Location: teacher\_dashboard.php");      }  }  //if no cookie check for session user  elseif  (isset($\_SESSION["user"]))  {      if (unserialize($\_SESSION["user"])[2] == "student")  Unserializing the session details and checking if the account type (field[2]) is equal to “student”      {            header("Location: student\_dashboard.php");      }      else      {           header("Location: teacher\_dashboard.php");      }  }  ?>      <!--NAV BAR-->      <nav class="sticky-top navbar">      <img src="images/book.png" style="margin-left:10px; max-width:55px; border-radius:55px; border: 3.5px solid white; box-shadow:none; max-height: 4vw; " onclick="Location: href='index.php'" />      <button class="btn btn-info" onclick="location = href='sign-up.php'" style="float:right;margin-right:10px; box-shadow:none; color:black; max-height: 4vw;" >Log in/Sign up</button>      </nav>    Bootstrap grid sytem      <div class="container-fluid min-height" style="margin:0px;">          <div class="row" style="margin-top:10px;">          <!--ROW 1!!-->              <div class="col-md-6"><img src="images/elearningimg.jpg" style="height: 30vw;"/></div>              <div class="col-md-6">  Bootstrap cardC with lorem ipsum text  ]                <div class="help" style="max-height: 30vw; margin:0;overflow: visible;">                      <h5 class="card-title">What do we do?</h5>                      <p class="card-text">  Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec iaculis diam eu enim fermentum rhoncus. Morbi euismod nulla commodo, consequat ligula ut, hendrerit dui. Nam sapien nunc, scelerisque quis mi a, tempor porta enim. Integer molestie ac nibh condimentum cursus. Etiam faucibus euismod tincidunt. Cras sagittis lacus eget condimentum fringilla. Orci varius natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Etiam suscipit nulla non tellus eleifend interdum. Phasellus bibendum metus ut arcu sodales, a gravida magna scelerisque. Vivamus et ultrices justo. Etiam ac metus nec diam ullamcorper bibendum nec nec tortor. Sed consectetur interdum quam ac accumsan. Proin quis efficitur risus.  Maecenas euismod arcu sollicitudin erat venenatis pretium. Suspendisse sit amet lobortis dui. Sed hendrerit luctus lorem, et porttitor nisi consequat eget. Ut vestibulum puruserdum.</p>                      <a href="sign-up.php" class="btn btn-outline-primary">Get Started</a>                  </div>              </div>  Row 2 is the black dividing strip that can be seen on the page          </div>        <!--ROW 2!!-->      </br>      <div class="row section ">          <div class="col-md-12" style="height:10vw; opacity: 0.85; background-color: black;"></div>      </div>      <!--ROW 3!!-->      <div class="row" style="margin-top:10px;">            <div class="col-md-6">              <div class="help" style="max-height: 30vw; margin:0;overflow: visible;">    Basically, the same as row 1                  <h5 class="card-title" >Start your future now</h5>                  <p class="card-text">  Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec iaculisnsequat ligula ut, hendrerit dui. Nam sapien nunc, scelerisque quis mi a, tempor porta enim. Integer molestie ac nibh condimentum cursus. Etiam faucibus euismod tincidunt. Cras sagittis lacus eget condimentum fringilla. Orci varius natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Etiam suscipit nulla non tellus eleifend interdum. Phasellus bibendum metus ut arcu sodales, a gravida magna scelerisque. Vivamus et ultrices justo. Etiam ac metus nec diam ullamcorper bibendum nec nec tortor. Sed consectetur interdum quam ac accumsan. Proin quis efficitur risus.  Maecenas euismod arcu sollicitudin erat venenatis pretium. Suspendisse sit amet lobortis dui. Sed hendrerit luctus lorem, et porttitor nisi consequat eget. Ut vestibulum purus id justo varit luctus lorem, et porttitor nisi consequat eget. Ut vestibulum purus id justo varit luctus lorem, et porttitor nisi consequat eget. Ut vestibulum purus id justo varius sagittis. Vivamus vulputate laoreet dictum. Maecenas in vestibulum nulla, id imperdiet erat. Nullam in ipsum egestas, porta diam a, scelerisque felis. Fusce eu pretium odio, quis facilisis felis. Aenean eleifend lobortis libero, ut pharetra massa volutpat sit amet. Sed luctus vitae lectus et interdum.</p>                  <a href="sign-up.php" class="btn btn-outline-primary ">Get Started</a>              </div>          </div>          <div class="col-md-6"><img src="images/elearningimg.jpg" style="height: 30vw;"/></div>      </div>  Link to elearning graphic       <!--FOOTER-->      <footer>          <div class="row section " style="margin:0px;">              <div class="col-md-12 footer" style="margin:0;" >              <p style="color:white;"></br>Find us at:</br>              Instagram: @GibJohn</br>              Facebook: @GibJohn</br>              Email: gibjohn@gmail.com</br>              Phone: 07473820938              </p>              </div>          </div>      </footer> |

#### Student Dashboard

Radio options to filter page. Onclick = JavaScript function (see above)

|  |
| --- |
| <script lang="javascript">            //stops unwanted refresh          if ( window.history.replaceState ) {           window.history.replaceState( null, null, window.location.href );      }  Allows form to be submitted when a radio (filter) is selected      //simply submits form      function formsubmit(){          document.getElementById("myform").submit();      }  </script>  <body>        <!--NAV BAR-->      <nav class="sticky-top navbar">      <img src="images/book.png" style="margin-left:48%; max-width:55px; border-radius:55px; border: 3.5px solid white; box-shadow:none; max-height: 4vw; " onclick="Location: href='index.php'" />      <button class="btn btn-primary" onclick="location = href='account.php'" style="float:right;margin-right:10px; box-shadow:none; color:black; max-height: 4vw;" >Account</button>      </nav>      <!--CONTAINER-->      <div class="container">          <!--SEARCH BAR-->          <div class="row" style="margin-top:7vw;">              <div style="margin:0; padding:0;" class="col-md-2">              <form id="myform" method="POST" action="#">                  <!--FILTER OPTIONS-->                  <center>See:</center>                  <label for="filter">All</label>                  <input class="form-check-input" type="radio" name="filter" value="1" onclick="formsubmit()" <?php if(!isset($\_POST['filter']))  echo "checked='checked'"; ?> />                  <label for="filter">Courses</label>                  <input class="form-check-input" type="radio" name="filter" value="2" onclick="formsubmit()" <?php if(isset($\_POST['filter'])) { if ($\_POST['filter'] == '2') echo "checked='checked'"; }?> />                  <label for="filter">Quizzes</label>                  <input class="form-check-input" type="radio" name="filter" value="3" onclick="formsubmit()" <?php if(isset($\_POST['filter'])) { if ($\_POST['filter'] == '3') echo "checked='checked'"; }?> />              </form>              </div>  Functioning search bar              <!--Searchbar -->              <div class="col-md-8">              <form action="#" method="post" class="d-flex">              <input class="form-control me-sm-2" type="text" name="criteria" placeholder="Search">              <button class="btn btn-outline-primary my-2 my-sm-0" type="submit">Search</button>              </form>              <div class="col-md-2"></div>          </div>          </div>      </div>      <!--COURSE LISTINGS-->        <div style="margin-left: 7vw; width: 90vw;" class="min\_height">        <?php            if (isset($\_POST['filter'])){S              $choice = $\_POST['filter'];  Code to determine which filter option is selected          }          else{              $choice = '1';          }          if (isset($\_POST['criteria'])){              $choice = "search";          }          include\_once "C:/xampp/htdocs/Projects/GibJohn/libraries/course\_listings.php";          if ($choice == '1')          {              populate\_courses();  Depending on filter choice, populates page with appropriate function from course\_listings.php              populate\_quizzes();          }          elseif ($choice == '2')          {              populate\_courses();          }          elseif ($choice == '3')          {              populate\_quizzes();          }          else{              searchbar($\_POST['criteria']);          }  Runs the searchbar function using the criteria entered into the searchbar input      ?>      </div>     <!--FOOTER-->   <footer>      <div class="row section " style="margin:0px;">          <div class="col-md-12 footer" style="margin:0;" >              <p style="color:white;"></br>Find us at:</br>                  Instagram: @GibJohn</br>                  Facebook: @GibJohn</br>                  Email: gibjohn@gmail.com</br>                  Phone: 07473820938              </p>          </div>      </div>  </footer>  </body> |

#### Student Monitor

|  |
| --- |
| <?php        if(!isset($\_SESSION)) {          session\_start();      }      include\_once "C:/xampp/htdocs/Projects/GibJohn/libraries/db.php";      include\_once "C:/xampp/htdocs/Projects/GibJohn/libraries/course\_listings.php";      /\*storing the selected option in session so that      courses table doesnt refresh when you change quiz table and vice versa \*/      if (isset($\_POST['selectcourse'])){          $courseId = $\_POST['selectcourse'];          $\_SESSION['selectcourse'] = $\_POST['selectcourse'];      }      elseif (isset($\_SESSION['selectcourse'])){          $courseId = $\_SESSION['selectcourse'];      }        else{          $courseId = '';      }          if (isset($\_POST['selectquiz'])){          $quizId = $\_POST['selectquiz'];          $\_SESSION['selectquiz'] = $\_POST['selectquiz'];      }      elseif (isset($\_SESSION['selectquiz'])){          $quizId = $\_SESSION['selectquiz'];      }      else{          $quizId = '';      }  ?>  <body>        <!--NAV BAR-->      <nav class="sticky-top navbar">      <img src="images/book.png" style="margin-left:48.25%; max-width:55px; border-radius:55px; border: 3.5px solid white; box-shadow:none; max-height: 4vw; " onclick="Location: href='index.php'" />      <a style="float:right; margin-right: 10px;" href="teacher\_dashboard.php" class="btn btn-danger">Back</a>      </nav>        <div class="contain\_me min\_height">      <!--LEFT SIDE-->      <div class="left-side">      <h4>My courses</h4>      <form method="post" action="#">      <select name="selectcourse" onchange="this.form.submit()" class="form-select" id="myselect" style="width:25vw;" >      <option  name='courseOption' value='none' >None</option>";      <?php teacher\_course\_options(); ?>      </select>      </form>      <?php echo selected\_name($courseId, 'course').      '<table class="table" style="width:80%;" >              <tr>                  <th>Name</th>                  <th>Progress</th>                  <th>Enrolment Date</th>              </tr>';                $result = mysqli\_query(get\_conn(), "SELECT concat(fname, ' ', sname) as Name, currentProgress, dateOfEnrol FROM enrols INNER JOIN students ON enrols.studentId = students.studentId WHERE courseId = '".$courseId."'");                while ($row = mysqli\_fetch\_array($result)) {                      echo '                      <tr>                      <td>'.$row['Name'].'</td>                      <td>'.$row['currentProgress'].'</td>                      <td>'.$row['dateOfEnrol'].'</td>                      </tr>';                }              echo '</table>'; ?>        </div>          <!--RIGHT SIDE-->      <div class="right-side">      <h4>Quiz Results</h4>      <form method="post" action="#">      <select name="selectquiz" onchange='this.form.submit()' class="form-select"  style="width:25vw;" >      <option  name='quizOption' value='none' >None</option>";      <?php teacher\_quiz\_options(); ?>      </select>      </form>      <?php echo selected\_name($quizId, 'quiz').      '<table class="table" style="width:80%;" >              <tr>                    <th>Student Name</th>                  <th>Result</th>              </tr>';                $result = mysqli\_query(get\_conn(), "SELECT quizName, quizResult FROM quizResults INNER JOIN quizzes ON quizResults.quizId = quizzes.quizId WHERE quizzes.quizId = '".$quizId."'");              $result2 = mysqli\_query(get\_conn(), "SELECT concat(fname, ' ', sname) as Name FROM students INNER JOIN quizResults ON students.studentId = quizResults.studentId WHERE quizResults.quizId = '".$quizId."'");              while ($row = mysqli\_fetch\_array($result)) {                  while ($rows = mysqli\_fetch\_array($result2)){                      echo '                      <tr>                      <td>'.$rows['Name'].'</td>                      <td>'.$row['quizResult'].'</td>                      </tr>';                    }                }              echo '</table>'; ?>      </div>  </div> |