|  |  |
| --- | --- |
| **Logo_Baru-Rasmi_UUM_02** | STIW2044: Mobile Programming Semester A212School of Computing, CAS, UUM **FRONT COVER** |
| **MIDTERM EXAM** | |
| |  |  | | --- | --- | | **Name** | Juanrico Alvaro | | **Matric No** | 702301 | | **YouTube Presentation Link** | https://youtu.be/6mW5gURV-7I | | **Phone Number** | 0136798361 | | **GitHub Link** | https://github.com/victorico123/flutter-projrct-UUM | | **Submission Date** | 11/6/2022 | | **Acknowledgment** | I hereby acknowledge that the following works are from my effort in submitting this document. If found otherwise, severe action such as marks deduction or removal from the assignment can be taken against me. | | **Digital Signature** |  | | **Students Picture** |  | | **Email use during book purchase** | juanricoalvaro@gmail.com | | **Your digital key from the book purchase** | 5oRiMDDWKW4U5hz | | |

**A212 MOBILE PROGRAMMING (100 MARKS)**

**Instructions**

The following midterm assignment is a continuation of lab assignment 2.

You have been assigned to continue with the development "My Tutor" mobile application from lab assignment 2. Complete the following instructions by implementing the solution using the Flutter-PHP-MySQL development stacks. You may use the lab 2 assignment project to complete this midterm assignment.

1. User login module: The following module will require for implementing user login.
2. Once the user registration module is completed and the user successfully registers their account, navigate to the login screen and perform the user authentication. Implement backend service for user authentication. Fill in the following section with the required codes and scripts.
   * 1. User class definition

5 Marks

class User {

  int? id;

  String? name;

  String? email;

  String? phone;

  String? address;

  String? image;

  User({this.id, this.email, this.name, this.phone, this.address, this.image});

  User.fromJson(Map<String, dynamic> json) {

    id = json['id'];

    email = json['email'];

    name = json['name'];

    phone = json['phone'];

    address = json['address'];

    image = json['image'];

  }

  Map<String, dynamic> toJson() {

    final Map<String, dynamic> data = <String, dynamic>{};

    data['id'] = id;

    data['email'] = email;

    data['name'] = name;

    data['phone'] = phone;

    data['address'] = address;

    data['image'] = image;

    return data;

  }

}

* + 1. PHP backend script

10 Marks

<?php

if (!isset($\_POST)) {

    $response = array('status' => 'failed', 'data' => null);

    sendJsonResponse($response);

    die();

}

include\_once("dbconnect.php");

$email = addslashes($\_POST['email']);

$password = $\_POST['password'];

$sqlLogin = "SELECT \* FROM users WHERE email = ? AND password = ?";

$stmt = $conn->prepare($sqlLogin);

$stmt->bind\_param("ss", $email, $password);

if ($stmt->execute()) {

    $result = $stmt->get\_result();

    $user\_name = "";

    while ($row = $result->fetch\_assoc()) {

        $user['id'] = $row['id'];

        $user['email'] = $row['email'];

        $user['name'] = $row['name'];

        $user['phone'] = $row['phone'];

        $user['address'] = $row['address'];

        $user['image'] = $row['image'];

    }

    $response = array('status' => 'success', 'data' => $user);

    sendJsonResponse($response);

} else {

    $response = array('status' => 'failed', 'data' => null);

    sendJsonResponse($response);

}

function sendJsonResponse($sentArray)

{

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

    echo json\_encode($sentArray);

}

* + 1. The Flutter method that implements the HTTP object POST call to login and creates a user object.

10 Marks

void \_validateUser() {

    String email = \_emailController.text;

    String password = \_passwordController.text;

    http.post(Uri.parse("http://10.19.48.148/myTutorAPI/login.php"),

        body: {"email": email, "password": password}).then((response) {

      var data = jsonDecode(response.body);

      if (response.statusCode == 200 && data['status'] == 'success') {

        gb.globaUser = User.fromJson(data['data']);

        Fluttertoast.showToast(

            msg: "Success",

            toastLength: Toast.LENGTH\_SHORT,

            gravity: ToastGravity.BOTTOM,

            timeInSecForIosWeb: 1,

            fontSize: 16.0);

        Navigator.of(context)

            .push(MaterialPageRoute(builder: (context) => HomeScreen()));

      } else {

        Fluttertoast.showToast(

            msg: data['status'],

            toastLength: Toast.LENGTH\_SHORT,

            gravity: ToastGravity.BOTTOM,

            timeInSecForIosWeb: 1,

            fontSize: 16.0);

      }

    });

  }

1. Main screen module: Once the user successfully login, navigate to the main page.
   1. The main page should implement a bottom navigation menu with the following menu icon; Subjects, Tutors, Subscribe, Favourite, and Profile. Show only the screenshot for the main page.

10 Marks

Graphical user interface, application

Description automatically generated

* 1. Download the following SQL file and import it into your database. Show both imported tables screenshots using the PhpMyAdmin table designer. SQL - <https://slumberjer.com/mytutor/mytutordb.sql>

2 Marks

Tbl\_tutors

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Tbl\_subjects

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

* 1. Download the following assets files and extract the folder into your application directory. The folder contains two subfolders that store tutors’ profile images and course images. Assets - <https://slumberjer.com/mytutor/assets.zip>. Print screen your server directory with all the assets.

2 Marks

A screenshot of a computer

Description automatically generated with medium confidence

Graphical user interface

Description automatically generated Graphical user interface, application, Teams

Description automatically generated

* 1. Once the user successfully login, the main page should display a list of subjects from the table tbl\_subjects from the database. Load the required data from the database and display it as a card list. The image for each subject is provided in (2. c).

1. Subject/Course class definition

5 Marks

class Course {

  String? subject\_id;

  String? subject\_name;

  String? subject\_description;

  String? subject\_price;

  String? tutor\_id;

  String? subject\_sessions;

  String? subject\_rating;

  Course(

      {this.subject\_id,

      this.subject\_name,

      this.subject\_description,

      this.subject\_price,

      this.tutor\_id,

      this.subject\_sessions,

      this.subject\_rating});

  Course.fromJson(Map<String, dynamic> json) {

    subject\_id = json['subject\_id'].toString();

    subject\_name = json['subject\_name'];

    subject\_description = json['subject\_description'];

    subject\_price = json['subject\_price'].toString();

    tutor\_id = json['tutor\_id'];

    subject\_sessions = json['subject\_sessions'].toString();

    subject\_rating = json['subject\_rating'].toString();

  }

  Map<String, dynamic> toJson() {

    final Map<String, dynamic> data = <String, dynamic>{};

    data['subject\_id'] = subject\_id;

    data['subject\_name'] = subject\_name;

    data['subject\_description'] = subject\_description;

    data['subject\_price'] = subject\_price;

    data['tutor\_id'] = tutor\_id;

    data['subject\_sessions'] = subject\_sessions;

    data['subject\_rating'] = subject\_rating;

    return data;

  }

}

1. PHP backend script with pagination that load subject/course list (5 items per page).

10 Marks

<?php

include("dbconnect.php");

session\_start();

if (!isset($\_POST)) {

    $response = array('status' => 'failed', 'data' => null);

    sendJsonResponse($response);

    die();

}

$limit = 5;

$page = (isset($\_POST['page']) && is\_numeric($\_POST['page'])) ? $\_POST['page'] : 1;

$paginationStart = ($page - 1) \* $limit;

$sqlSubject = "SELECT \* FROM tbl\_subjects LIMIT $paginationStart, $limit";

$stmt = $conn->prepare($sqlSubject);

$stmt->execute();

$result = $stmt->get\_result();

// Get total records

$sql = $conn->query("SELECT count(subject\_id) AS id FROM tbl\_subjects")->fetch\_assoc();

$allRecrods = $sql['id'];

// Calculate total pages

$totalPages = ceil($allRecrods / $limit);

// Prev + Next

$prev = $page - 1;

$next = $page + 1;

if ($allRecrods > 0) {

    $courses['courses'] = array();

    while ($row = $result->fetch\_assoc()) {

        $clist = array();

        $clist['subject\_id'] = $row['subject\_id'];

        $clist['subject\_name'] = $row['subject\_name'];

        $clist['subject\_description'] = $row['subject\_description'];

        $clist['subject\_sessions'] = $row['subject\_sessions'];

        $clist['subject\_rating'] = $row['subject\_rating'];

        $clist['subject\_price'] = $row['subject\_price'];

        $clist['tutor\_id'] = $row['tutor\_id'];

        array\_push($courses['courses'], $clist);

    }

    $response = array('status' => 'success', 'page' => "$page", 'totalPages' => "$totalPages", 'data' => $courses);

    sendJsonResponse($response);

} else {

    $response = array('status' => 'failed', 'page' => "$page", 'totalPages' => "$totalPages", 'data' => null);

    sendJsonResponse($response);

}

function sendJsonResponse($sentArray)

{

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

    echo json\_encode($sentArray);

}

?>

1. The Flutter method that implements the HTTP object POST call to load the subject data into the list object.

10 Marks

I Use both tutor and subject data.

void loadCourse() {

    \_numPages ?? 1;

    http.post(Uri.parse("http://10.19.48.148/myTutorAPI/load\_subject.php"),

        body: {

          'page': \_currentPage.toString(),

        }).timeout(

      const Duration(seconds: 5),

      onTimeout: () {

        return http.Response(

            'Error', 408); // Request Timeout response status code

      },

    ).then((response) {

      var jsondata = jsonDecode(response.body);

      if (response.statusCode == 200 && jsondata['status'] == 'success') {

        var extractdata = jsondata['data'];

        \_numPages = int.parse(jsondata['totalPages']);

        if (extractdata['courses'] != null) {

          CourseList = <Course>[];

          extractdata['courses'].forEach((v) {

            CourseList.add(Course.fromJson(v));

          });

          titlecenter = CourseList.length.toString() + " Courses Available";

        } else {

          titlecenter = "No Course Available";

          CourseList.clear();

        }

        setState(() {});

      } else {

        titlecenter = "No Course Available 2";

        CourseList.clear();

        setState(() {});

      }

    });

  }

  void loadTutor() {

    http.post(Uri.parse("http://10.19.48.148/myTutorAPI/load\_tutor.php"),

        body: {

          'page': \_currentPage.toString(),

          'limit': 'all',

        }).timeout(

      const Duration(seconds: 5),

      onTimeout: () {

        return http.Response(

            'Error', 408); // Request Timeout response status code

      },

    ).then((response) {

      var jsondata = jsonDecode(response.body);

      if (response.statusCode == 200 && jsondata['status'] == 'success') {

        var extractdata = jsondata['data'];

        if (extractdata['tutors'] != null) {

          TutorList = <Tutor>[];

          extractdata['tutors'].forEach((v) {

            TutorList.add(Tutor.fromJson(v));

          });

        } else {

          TutorList.clear();

        }

        setState(() {});

      } else {

        TutorList.clear();

        setState(() {});

      }

    });

  }

1. Screenshot for the main page that shows a list of data.

2 Marks

Graphical user interface, text, application

Description automatically generatedGraphical user interface, text, application, chat or text message

Description automatically generatedA screenshot of a phone

Description automatically generated with low confidence

1. Tutor List module. This module is accessible from the Tutor menu icon from the bottom navigation bar. Create a new class screen for listing available tutors from the tutor’s table.
   1. Tutor class definition

5 Marks

class Tutor {

  String? tutor\_id;

  String? tutor\_email;

  String? tutor\_phone;

  String? tutor\_name;

  String? tutor\_description;

  String? tutor\_datereg;

  Tutor(

      {this.tutor\_id,

      this.tutor\_email,

      this.tutor\_phone,

      this.tutor\_name,

      this.tutor\_description,

      this.tutor\_datereg});

  Tutor.fromJson(Map<String, dynamic> json) {

    tutor\_id = json['tutor\_id'].toString();

    tutor\_email = json['tutor\_email'].toString();

    tutor\_phone = json['tutor\_phone'].toString();

    tutor\_name = json['tutor\_name'].toString();

    tutor\_description = json['tutor\_description'].toString();

    tutor\_datereg = json['tutor\_datereg'].toString();

  }

  Map<String, dynamic> toJson() {

    final Map<String, dynamic> data = <String, dynamic>{};

    data['tutor\_id'] = tutor\_id;

    data['tutor\_email'] = tutor\_email;

    data['tutor\_phone'] = tutor\_phone;

    data['tutor\_name'] = tutor\_name;

    data['tutor\_description'] = tutor\_description;

    data['tutor\_datereg'] = tutor\_datereg;

    return data;

  }

}

* 1. PHP backend script with pagination to load tutor list (5 items per page).

10 Marks

<?php

include("dbconnect.php");

session\_start();

if (!isset($\_POST)) {

    $response = array('status' => 'failed', 'data' => null);

    sendJsonResponse($response);

    die();

}

// Get total records

$sql = $conn->query("SELECT count(tutor\_id) AS id FROM tbl\_tutors")->fetch\_assoc();

$allRecrods = $sql['id'];

$limit = (isset($\_POST['limit'])) ? $allRecrods : 5;

$page = (isset($\_POST['page']) && is\_numeric($\_POST['page'])) ? $\_POST['page'] : 1;

$paginationStart = ($page - 1) \* $limit;

$sqlTutor = "SELECT \* FROM tbl\_tutors LIMIT $paginationStart, $limit";

$stmt = $conn->prepare($sqlTutor);

$stmt->execute();

$result = $stmt->get\_result();

// Calculate total pages

$totalPages = ceil($allRecrods / $limit);

// Prev + Next

$prev = $page - 1;

$next = $page + 1;

if ($allRecrods > 0) {

    $tutors['tutors'] = array();

    while ($row = $result->fetch\_assoc()) {

        $tlist = array();

        $tlist['tutor\_id'] = $row['tutor\_id'];

        $tlist['tutor\_email'] = $row['tutor\_email'];

        $tlist['tutor\_phone'] = $row['tutor\_phone'];

        $tlist['tutor\_name'] = $row['tutor\_name'];

        $tlist['tutor\_description'] = $row['tutor\_description'];

        $tlist['tutor\_datereg'] = $row['tutor\_datereg'];

        array\_push($tutors['tutors'], $tlist);

    }

    $response = array('status' => 'success', 'page' => "$page", 'totalPages' => "$totalPages", 'data' => $tutors);

    sendJsonResponse($response);

} else {

    $response = array('status' => 'failed', 'page' => "$page", 'totalPages' => "$totalPages", 'data' => null);

    sendJsonResponse($response);

}

function sendJsonResponse($sentArray)

{

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

    echo json\_encode($sentArray);

}

?>

* 1. The Flutter method that implements the HTTP object POST call to load the subject data into the list object

10 Marks

void loadTutor() {

    \_numPages ?? 1;

    http.post(Uri.parse("http://10.19.48.148/myTutorAPI/load\_tutor.php"),

        body: {

          'page': \_currentPage.toString(),

        }).timeout(

      const Duration(seconds: 5),

      onTimeout: () {

        return http.Response(

            'Error', 408); // Request Timeout response status code

      },

    ).then((response) {

      var jsondata = jsonDecode(response.body);

      if (response.statusCode == 200 && jsondata['status'] == 'success') {

        var extractdata = jsondata['data'];

        \_numPages = int.parse(jsondata['totalPages']);

        if (extractdata['tutors'] != null) {

          TutorList = <Tutor>[];

          extractdata['tutors'].forEach((v) {

            TutorList.add(Tutor.fromJson(v));

          });

          // titlecenter = TutorList.length.toString() + " Tutors Available";

        } else {

          titlecenter = "No Tutor Available";

          TutorList.clear();

        }

        setState(() {});

      } else {

        titlecenter = "No Tutor Available 2";

        TutorList.clear();

        setState(() {});

      }

    });

  }

* 1. Screenshot for the main page that shows a list of data.

2 Marks

Graphical user interface, application, chat or text message

Description automatically generated Graphical user interface, text, application, chat or text message

Description automatically generated Graphical user interface, text, application, chat or text message

Description automatically generated

1. Create a presentation video that describes all the above tasks. The video must demonstrate the following:

**Put your video link here:** https://youtu.be/6mW5gURV-7I

* 1. Login screen

5 Marks

* 1. Subjects list screen (main screen)

5 Marks

* 1. Tutors list screen (tutor screen)

2 Marks