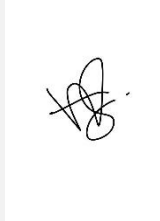






STIW2044: Mobile Programming  
Semester A222  
School of Computing, CAS, UUM

## FRONT COVER

### Lab Assignment 2

Name	MAZLIN NATASHA BINTI MOHAMAD NASIR
Matric No	288460
YouTube Presentation Link	<a href="https://youtu.be/OcweOphMtwU">https://youtu.be/OcweOphMtwU</a>
Phone Number	01128846070
GitHub Link	<a href="https://github.com/mazlinnatasha">https://github.com/mazlinnatasha</a>
Submission Date	21/5/2023
Acknowledgment	I hereby signed and 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 (book purchase)	mazlinnatasha02@gmail.com
Digital key (book purchase)	wqLPH54B8ARYW0Q

	STIW2044: Mobile Programming Semester A222 School of Computing, CAS, UUM
Lab Assignment 2	
Given date: <b>14/2/2023</b>	
Submission date: <b>21/2/2023</b>	
<b>15 Marks</b>	

### Application description

**BarterIt** is a mobile app that allows users to barter used items with each other. It's a great way to get rid of unwanted items, find new things to use, and save money. To use **BarterIt**, simply create an account and start browsing the items that are available. You can search for specific items, or browse by category. Once you find an item that you're interested in, you can contact the seller to make a trade. **BarterIt** is a great way to declutter your home, save money, and help the environment. By trading used items, you can keep them out of landfills and give them a new life with someone else.

### App Requirement

- Account creation: Users must create an account in order to use the app. This will require them to provide their name, email address, and password.
- Item listing: Users can list items that they are willing to trade. This will require them to provide a description of the item, a photo, and the value of the item.
- Search: Users can search for items that they are interested in trading for. This can be done by keyword, category, or location.
- Messaging: Users can message each other to negotiate trades.
- Rating and review system: Users can rate and review each other after a trade is completed. This helps to ensure that users have a positive experience with the app.

### Module 1:

The first part of the above app is implementation of login and registration. The app will require a splash screen, a user login screen, a user registration screen and empty main screen for the app. The app will require backend to handle login and registration using PHP and MySQL for the database.

## Instruction2 – Develop the application using Flutter SDK

Develop the application using the application described above.

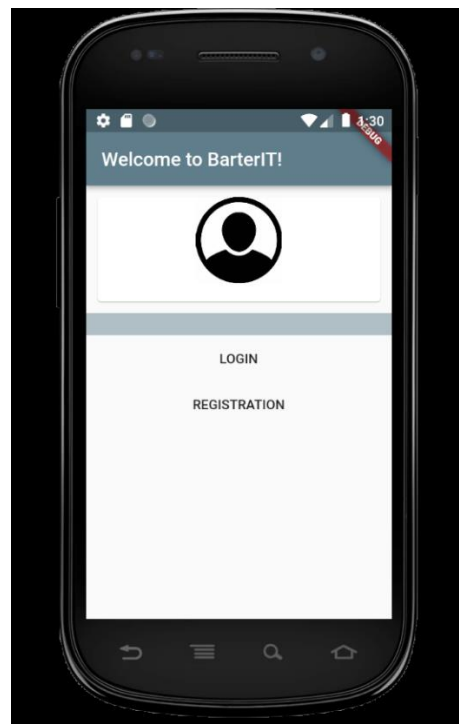
1. GitHub link to your project (Don't set it as private). Include **server-side** directory and **SQL export** statement file into the project directory.

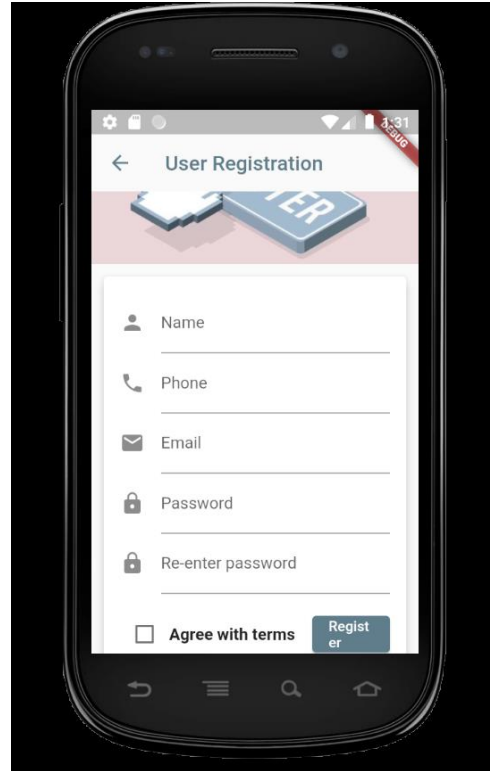
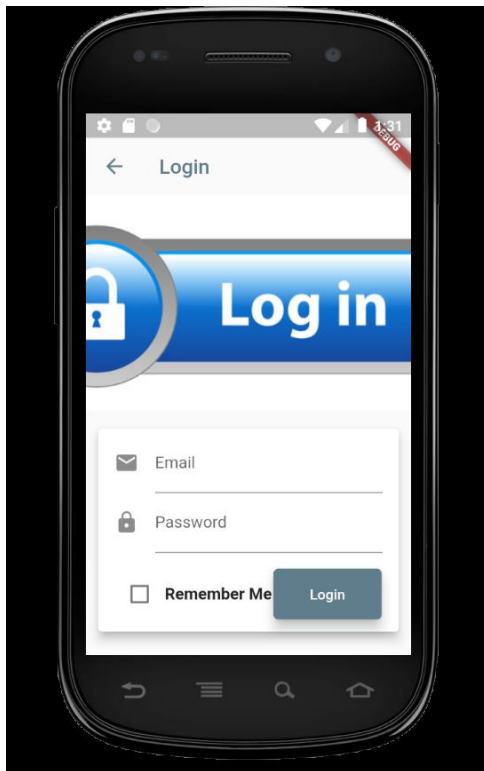
<https://github.com/mazlinnatasha/labAssignment2BarterIT>

(1 Marks)

2. User Interface screenshots for splash, registration, login screens, main screen and database design snapshot.

(2 Marks)





3. YouTube video link (Don't set it as private). Recorded video presentation that shows the login and registration are functionals.

(2 Marks)

<https://youtu.be/Ocwe0phMtwU>

Instructions 2 – Once instruction 1 is completed answer all the following questions. The following questions will only count and check if the above full source code and video are available. Your answers should only come from your code. Only shows the required code segment and any additional codes added to the answer will consider as the wrong answer.

1. Show code segment for parent widget from registration screen that responsible to check the states for all input text validation?

(1 Marks)

```
Expanded(  
  child: ElevatedButton(  
    onPressed: onRegisterDialog,  
    child: const Text("Register")) // ElevatedButton // Expanded  
  ),  
),
```

2. Show widget use from the registration screen to set image asset width?

(1 Marks)

```
43   SizedBox(  
44     height: screenHeight * 0.35,  
45     width: screenWidth,  
46     child: Image.asset(  
47       "assets/register.jpeg",  
48       fit: BoxFit.cover,  
49     )), // Image.asset // SizedBox
```

3. Show your PHP script used to execute SQL statement for the login operation.

(1 Mark)

```
login_user.php  
1 <?php  
2 if (isset($_POST)){  
3   $response = array('status' => 'failed', 'data' => null);  
4   sendJsonResponse($response);  
5   die();  
6 }  
7 $email = $_POST['email'];  
8 $password = sha1($_POST['password']);  
9  
10 include_once("dbconnect.php");  
11  
12 $sqllogin = "SELECT * FROM tbl_users WHERE user_email = '$email' AND  
13 user_password = '$password'";  
14 $result = $conn->query($sqllogin);  
15  
16 if ($result->num_rows > 0) {  
17   while ($row = $result->fetch_assoc()) {  
18     $userlist = array();  
19     $userlist['id'] = $row['user_id'];  
20     $userlist['name'] = $row['user_name'];  
21     $userlist['email'] = $row['user_email'];  
22     $userlist['phone'] = $row['user_phone'];  
23  
24     $response = array('status' => 'success', 'data' => $userlist);  
25     sendJsonResponse($response);  
26   }  
27 }else{  
28   $response = array('status' => 'failed', 'data' => null);  
29   sendJsonResponse($response);  
30 }  
31  
32 function sendJsonResponse($sentArray)  
33 {  
34   header('Content-Type: application/json');  
35   echo json_encode($sentArray);  
36 }  
37 ?>
```

4. Show code segment from HTTP object where the registration is passing its data to the backend script.

```
http.post(Uri.parse("${MyConfig().SERVER}/barterit/php/register_user.php"),
    body: {
        "name": name,
        "email": email,
        "phone": phone,
        "password": passa,
    }).then((response) {
```

5. Show code segment where the preference object is created to store preference data.

(2 Mark)

```

    }
    Future<void> loadPref() async {
        SharedPreferences prefs = await SharedPreferences.getInstance();
        String email = (prefs.getString('email')) ?? '';
        String password = (prefs.getString('pass')) ?? '';
        _isChecked = (prefs.getBool('checkbox')) ?? false;
        if (_isChecked) {
            setState(() {
                _emailEditingController.text = email;
                _passEditingController.text = password;
            });
        }
    }
}

```

6. Show your registration debug print output of the returned API JSON data if the login had success.

(1 Mark)

[illegible]

7. Show script where backend operation where it checks for the availability of post array.

(2 Mark)

```
if (!isset($_POST)){
    $response = array('status' => 'failed', 'data' => null);
    sendJsonResponse($response);
    die();
}
```