BLUE-SIGN: ATTENDENCE MANAGEMENT SYSTEM

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS OF THE DEGREE OF

BACHELOR OF ENGINEERING

IN

INFORMATION TECHNOLOGY

BY

DIVYAJOTHI RAJA (202103011)

SAMUEL PALLIKONDA (2022032004)

KHIZAR SHAIKH (2022032006)

NELSON KOLAS (2022032007)

UNDER THE GUIDANCE OF

Prof. Stella J

(Department of Information Technology)

DEPARTMENT OF INFORMATION TECHNOLOGY

XAVIER INSTITUTE OF ENGINEERING

UNIVERSITY OF MUMBAI

2023 – 2024

XAVIER INSTITUTE OF ENGINEERING
Mahim, Mumbai 400016
Department of Information Technology
(NBA Accredited)
(Approved by AICTE, Govt. of Maharashtra and Affiliated to University of Mumbai)

CERTIFICATE

This to certify that

Divyajothi Raja	(202103011)
Samuel Pallikonda	(2022032004)
Khizar Shaikh	(2022032006)
Nelson Kolas	(2022032007)

Have satisfactorily carried out the MINI-PROJECT work titled "**BlueSign: Attendence Management System**" in partial fulfillment of the degree of Bachelor of Engineering as laid down by the University of Mumbai during the academic year 2022-2023.

Internal Examiner/Guide **External Examiner**

Date:

Place: MAHIM, MUMBAI

DECLARATION

I declare that this written submission represents my ideas in my own words and where other's Ideas or words have been included, I have adequately cited and referenced the original sources.

I also declare that I have adhered to all the principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission.

I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which thus have not been properly cited or from whom proper permission have not been taken when needed.

Divyajothi Raja (202103011) ----------

Samuel Pallikonda (2022032004) ----------

Khizar Shaikh (2022032006) ----------

Nelson Kolas (2022032007) ----------

Date:

ABSTRACT

BlueSign is a pioneering mobile application poised to revolutionize attendance management through cutting-edge Bluetooth technology. Offering a contemporary, streamlined, and intuitive approach to marking attendance, BlueSign transcends the constraints of conventional methods. With its innovative utilization of Bluetooth, BlueSign enables seamless wireless attendance tracking, providing a hassle-free experience for both administrators and users. By harnessing the power of modern technology, BlueSign redefines efficiency in attendance management, offering unparalleled convenience and accuracy. Through its user-friendly interface, BlueSign empowers organizations to optimize their attendance processes, saving valuable time and resources. With BlueSign, the era of cumbersome attendance tracking methods is a thing of the past, ushering in a new era of efficiency and productivity.

TABLE OF CONTENTS

LIST OF FIGURES

Acknowledgement

With a deep sense of gratitude, we would like to thank the guidance of our project guide Prof. Stella J for providing us time-to-time assistance with such an environment to achieve goals of our project and supporting us constantly.

We express our sincere gratitude to our Honorable Principal Dr. Y. D. Venkatesh for encouragement and facilities provided to us.

We would like to place on record our deep sense of gratitude to Prof. Jaychand Upadhyay, Head of Dept Of Information Technology, Xavier Institute of Engineering, Mahim, Mumbai, for her generous guidance help and useful suggestions.

We would also like to thank our entire Information Technology staff who have willingly cooperated with us in resolving our queries and providing us all the required facilities on time.

Divyajothi Raja (202103011) ----------

Samuel Pallikonda (2022032004) ----------

Khizar Shaikh (2022032006) ----------

Nelson Kolas (2022032007) ----------

TE (IT) SEM-VI (AY: 2023-24)

Subject: MAD & PWA Lab (Lab code:ITL604)

Course Code	Course Name	Teaching Scheme (Contact Hours)		Credits Assigned		
		Theory	Practical	Theory	Practical	Total
ITL604	MAD & PWA Lab	--	02	--	01	01

Course Code	Course Name	Examination Scheme							
		Theory					Term Work	Pract / Oral	Total
		Internal Assessment			End Sem Exam	Exam Duration (in Hrs)			
		Test1	Test 2	Avg.					
ITL604	MAD & PWA Lab	--	--	--	--	--	25	25	50

Lab Objectives:

1. To learn To learn the basics of Flutter framework.
2. Develop the App UI by incorporating widgets, layouts, gestures and animation.
3. Create a production ready Flutter App by including files and Firebase back end service.
4. Learn the Essential technologies, and Concepts of PWA's to get started as quickly and efficiently as possible.
5. Develop responsive web applications by combining AJAX development techniques with the jQuery JavaScript library.
6. Understand how service workers operate and also learn to Test and Deploy PWA.

Lab Outcomes:

LO	LO Statement	PO Mapped
1	**Design** and **Develop** cross platform mobile application development using Flutter framework	PO1, PO3, PO5, PO6, PO8, PO9, PO10, PO12
2	**Design** and **Develop** interactive Flutter App by using widgets, layouts, gestures and animation	PO1, PO3, PO5, PO6, PO8, PO9, PO10, PO12
3	**Analyze** and **Build** production ready Flutter App by incorporating back end services and deploying on Android / iOSBuild	PO1, PO3, PO5, PO6, PO8, PO9, PO10, PO12
4	**Analyze** various PWA frameworks and their requirements	PO1, PO3, PO5, PO6, PO8, PO9, PO10, PO12
5	**Design** and **Develop** a responsive User Interface by applying PWA Design techniques	PO1, PO3, PO5, PO6, PO8, PO9, PO10, PO12
6	**Develop** and **Analyse** PWA Features and deploy it over app hosting solutions	PO1, PO3, PO5, PO6, PO8, PO9, PO10, PO12

Rubrics for Mini Project						
Roll no:	Name of the Student	Problem Statement (15)	Creativity & Quality of Work Done (15)	Punctuality & Lab Ethics (10)	Performance & Presen-tation (10)	Total (50)
01	Divyajothi Raja					
02	Nelson Kolas					
35	Samuel Pallikonda					
47	Khizar Shaikh					

1. INTRODUCTION TO MAD OR PWA

1.1 MAD (MODERN ANDROID DEVELOPMENT):

MAD refers to a modern approach to developing Android applications, emphasizing best practices, tools, and libraries provided by Google to streamline the development process and improve the overall quality of Android apps.
Key components of MAD include Kotlin as the preferred programming language, Jetpack libraries for common tasks like navigation, UI design, and data management, Kotlin Coroutines for asynchronous programming, and Material Design guidelines for creating visually appealing and consistent user interfaces.
MAD also promotes a modular architecture, where apps are divided into independent modules to improve maintainability, testability, and scalability.

1. Kotlin: Kotlin is the preferred programming language for Android development in MAD. It offers concise syntax, null safety, and interoperability with existing Java codebases, making development more efficient and less error-prone.

2. Jetpack Libraries: Jetpack is a set of libraries, tools, and guidance to help developers build high-quality Android apps more quickly and easily. It includes components like LiveData, ViewModel, Room, Navigation, and WorkManager, which simplify common tasks such as UI design, data persistence, navigation, and background processing.

1.2 PWA (PROGRESSIVE WEB APPLICATION):

PWA is a type of web application that utilizes modern web technologies to deliver an app-like experience to users, regardless of the device or browser they are using.
Key features of PWAs include responsiveness (they adapt to different screen sizes), reliability (they work offline or on low-quality networks), and the ability to be installed on the user's device like native apps.
PWAs leverage technologies such as Service Workers for offline capabilities, Web App Manifest for installation, and HTTPS for security.
They offer benefits to both developers and users, including faster load times, improved engagement, and reduced development costs compared to native apps.

1. Responsive Design: PWAs are designed to be responsive, meaning they adapt to different screen sizes and orientations, providing a consistent user experience across desktops, tablets, and smartphones.

2. Service Workers: Service Workers are a key technology in PWAs that enable offline capabilities, push notifications, and background sync. They run separately from the main browser thread, allowing PWAs to cache content and resources, making them available even when the device is offline or on a slow network.

In summary, MAD focuses on modernizing Android app development using Google's recommended tools and practices, while PWA enables the creation of web applications with app-like features and experiences across various platforms and devices.

2. INTRODUCTION

BlueSign is an innovative mobile application developed using Flutter, designed to revolutionize attendance management for organizations of all sizes. By leveraging the power of Bluetooth technology, BlueSign offers a seamless and efficient solution for marking attendance wirelessly, eliminating the constraints of traditional methods. With its cross-platform compatibility, BlueSign ensures accessibility across iOS and Android devices, providing a user-friendly experience for both administrators and employees. The intuitive interface allows users to mark attendance with a simple tap, while administrators gain access to comprehensive analytics and reporting tools to optimize workforce management strategies. With a focus on data security and privacy, BlueSign prioritizes the protection of sensitive information, ensuring compliance with regulatory standards. BlueSign represents a paradigm shift in attendance tracking, promising to streamline operations and enhance productivity in today's digital age.

2.1 PROBLEM DEFINITION

Manual attendance tracking methods are inefficient and prone to errors, leading to inaccuracies and discrepancies in attendance records. This inefficiency is exacerbated by the lack of real-time visibility into attendance status, hindering timely intervention and decision-making processes. Additionally, traditional attendance records pose significant privacy and security concerns, as sensitive information is often stored in physical formats vulnerable to loss, theft, or unauthorized access. Moreover, the limited scalability of manual attendance tracking presents challenges for large institutions or organizations, as the manual processes struggle to accommodate the growing number of employees efficiently. These issues underscore the urgent need for a modern and secure attendance management solution that can provide real-time visibility, streamline operations, and ensure data privacy and scalability for organizations of all sizes.

2.2 AIMS AND OBJECTIVES

Aim:

The aim of this project is to develop and implement an innovative attendance management solution that addresses the shortcomings of manual tracking methods and provides a modern, efficient, and secure alternative for organizations.

Objectives:

1. To design a user-friendly mobile application utilizing Flutter framework that enables employees to mark attendance effortlessly and administrators to manage attendance records efficiently.
2. To integrate Bluetooth technology into the application to facilitate wireless attendance tracking, reducing the reliance on manual input and minimizing errors.
3. To provide real-time visibility into attendance status through a robust dashboard interface, empowering administrators to make informed decisions and intervene promptly when necessary.
4. To prioritize data security and privacy by implementing encryption protocols and ensuring compliance with regulatory standards for the protection of sensitive attendance records.
5. To enhance scalability by developing features that accommodate the diverse needs of organizations, particularly large institutions with a significant number of employees.
6. To conduct thorough testing and evaluation of the application to ensure reliability, accuracy, and seamless functionality across different devices and platforms.
7. To provide comprehensive documentation and training resources to facilitate the adoption and implementation of the attendance management solution by organizations.
8. To gather feedback from users and stakeholders to identify areas for improvement and iteratively enhance the application to meet evolving needs and preferences.

2.3 SCOPE OF THE PROJECT

The project aims to develop and implement an innovative attendance management solution to address the limitations of manual tracking methods in organizations. It involves designing a user-friendly mobile application using the Flutter framework, allowing employees to mark attendance effortlessly and administrators to manage records efficiently. Integration of Bluetooth technology facilitates wireless attendance tracking, reducing reliance on manual input and minimizing errors. A robust dashboard interface provides real-time visibility into attendance status, empowering administrators to make informed decisions promptly. Prioritizing data security, encryption protocols will be implemented to safeguard sensitive attendance records and ensure compliance with regulatory standards. Scalability will be enhanced by developing features that cater to diverse organizational needs, particularly large institutions with numerous employees. Thorough testing and evaluation across different devices and platforms will ensure reliability and seamless functionality. Comprehensive documentation and training resources will facilitate adoption and implementation by organizations. Feedback from users and stakeholders will be gathered to identify areas for improvement, allowing for iterative enhancements to meet evolving needs and preferences.

3. SYSTEM DESCRIPTION

1. The attendance management solution comprises a mobile application developed using the Flutter framework, which serves as the primary interface for both employees and administrators. Through this application, employees can easily mark their attendance using their mobile devices, leveraging the intuitive user interface for seamless interaction. The application integrates Bluetooth technology to enable wireless attendance tracking, reducing the need for manual input and minimizing potential errors associated with traditional methods.
2. Administrators access a robust dashboard interface within the application, providing real-time visibility into attendance status across the organization. This dashboard empowers administrators to monitor attendance trends, identify patterns, and make informed decisions promptly. Additionally, the dashboard may include features such as attendance reports, employee profiles, and configurable settings to tailor the solution to the organization's specific requirements.
3. Data security and privacy are paramount in the system design, with encryption protocols implemented to safeguard sensitive attendance records. Compliance with regulatory standards ensures that organizational data is protected and managed in accordance with legal requirements.
4. Scalability is a key aspect of the system, with features designed to accommodate the diverse needs of organizations, including large institutions with a significant number of employees. The system architecture allows for easy customization and expansion to meet evolving requirements as the organization grows.
5. Thorough testing and evaluation are conducted to ensure the reliability, accuracy, and seamless functionality of the system across different devices and platforms. This includes compatibility testing, performance testing, and user acceptance testing to validate the system's effectiveness in real-world scenarios.
6. Comprehensive documentation and training resources are provided to facilitate the adoption and implementation of the attendance management solution by organizations. This includes user manuals, technical documentation, and training materials tailored to various user roles within the organization.

3.1 DESIGN

Figure 3.1.1: Design 1

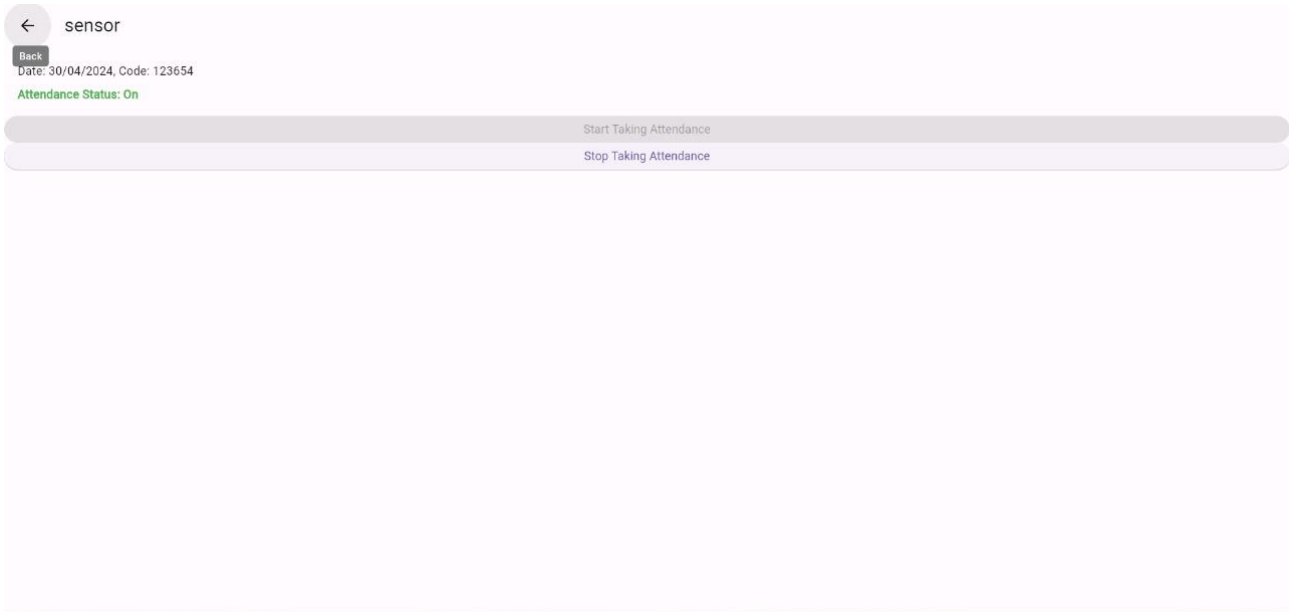

Figure 3.1.2: Design 2

3.2 HARDWARE, SOFTWARE AND CLOUD PLATFORMS USED

3.2.1 HARDWARE

- OS – Windows
- Ram : 8GB
- Bluetooth Technology

3.2.2 SOFTWARE

- Flutter SDK
- Android Studio
- VScode

3.2.3 DATABASE USED

- Mongo DB

3.3 IMPLEMENTATION METHODOLOGY

1. **Project Initiation**:
    - Define project objectives, scope, and deliverables.
    - Establish project team roles and responsibilities.
    - Conduct a kickoff meeting to align stakeholders and team members.
2. **Requirements Gathering**:
    - Engage with stakeholders to understand organizational needs and requirements.
    - Document functional and non-functional requirements for the attendance management solution.
    - Prioritize requirements based on importance and feasibility.
3. **Design Phase**:
    - Design the system architecture, including database schema, application components, and integration points.
    - Create wireframes and mockups to visualize the user interface.
    - Define data models, API specifications, and system workflows.
4. **Development**:
    - Develop the mobile application using the Flutter framework, adhering to design specifications and coding standards.
    - Implement Bluetooth integration for wireless attendance tracking.
    - Develop the dashboard interface for administrators, incorporating real-time data visualization and reporting features.
    - Implement encryption protocols for data security and privacy.
5. **Testing**:
    - Conduct unit testing to validate individual components and functionalities.
    - Perform integration testing to ensure seamless interaction between different modules of the system.
    - Conduct compatibility testing across various devices and platforms.
    - Perform performance testing to assess system scalability and responsiveness.
    - Engage end-users in user acceptance testing (UAT) to validate the system against real-world scenarios.
6. **Deployment**:
    - Prepare the system for deployment, including server setup, database configuration, and application deployment.
    - Roll out the attendance management solution to a staging environment for final testing and validation.
    - Plan and execute a deployment strategy, ensuring minimal disruption to ongoing operations.
    - Monitor system performance and address any issues that may arise during the deployment process.

3.4 CODE

Main

.dart

```
import
'package:flutter/material.dart';
import
'package:frontend/pages/CreateCla
ssroomPage.dart';
import
'package:frontend/pages/verificatio
n.dart';
import 'package:get/get.dart';
import
'package:frontend/authentication/a
uth.dart';
import
'package:frontend/pages/home.dart
';
import
'package:frontend/pages/login.dart'
;

void main() {
  Get.put(AuthService());

  runApp(MyApp());
}

class MyApp extends
StatelessWidget {
  final AuthService authService =
AuthService();

  MyApp({super.key});

  @override
  Widget build(BuildContext
context) {
    return GetMaterialApp(

debugShowCheckedModeBanner:
false,
      // Define initial route using
GetX
      initialRoute:
!authService.isLoggedIn ? '/home'
: '/login',
      // Define routes using GetX
      getPages: [
        GetPage(name: '/login', page:
() => const LoginPage()),
        GetPage(name: '/home', page:
() => const HomePage()),
        GetPage(name: '/verification',
page: () => const
VerificationPage()),
        GetPage(
          name: '/create_classroom',
          page: () => const
CreateClassroomPage()), // Add
this line
      ],
      // Initialize AuthService in the
builder
      builder: (context, child) {
        authService.initialize(); //
Initialize AuthService
        return child!;
      },
    );
  }
}
```

Auth.dart

```
import 'dart:async';
import 'dart:convert';
```

```
import 'package:http/http.dart' as
http;

class AuthService {
  bool _isLoggedIn = false;
   String? name;
   String? email;
  bool get isLoggedIn =>
_isLoggedIn;
  late bool debugLog;
  static const ip =
'http://192.168.34.183:3000';
  late String uuid;
  // Simulating async initialization
task
  Future<void> initialize() async {
    // Simulate asynchronous
initialization tasks, such as
fetching initial data
    await Future.delayed(const
Duration(seconds: 2));

    // Set isLoggedIn to true if
initialization is successful
    // _isLoggedIn = true;
  }

  // Future<Map<String,
dynamic>> login(String username,
String password) async {
  //   // Perform authentication logic
here
  //   // For demonstration purposes,
we'll simulate a successful login
with dummy JSON data
  //   // Replace this with your
actual login logic

  //   // Simulate authentication
with delay
  //   await Future.delayed(const
Duration(seconds: 1));

  //   // If authentication is
successful, return dummy JSON
data
  //   if (username == 'test' &&
password == '1234') {
  //     _isLoggedIn = true;
  //     return {
  //       'success': true,
  //       'message': 'Login
successful!',
  //       'user': {
  //         'id': 1,
  //         'username': username,
  //         // Add other user data here
if needed
  //       },
  //     };
  //   } else {
  //     // If authentication fails,
return error JSON data
  //     return {
  //       'success': false,
  //       'message': 'Invalid
username or password',
  //     };
  //   }
  // }

  Future<Map<String, dynamic>>
signup(
      String name, String email,
String password, String number)
async {
    try {
      const String url =
'$ip/auth/signup/signup';
      final Map<String, String>
headers = {'Content-Type':
'application/json'};
      final http.Response response =
await http.post(
        Uri.parse(url),
        headers: headers,
        body: jsonEncode({
          'name': name,
```

```
        'email': email,
        'password': password, // Assuming this is a constant value
        'phoneNo': number, // Assuming this is a constant value
       }),
     );

     if (response.statusCode == 200) {
       final Map<String, dynamic> restoken = jsonDecode(response.body);

       // print('Response: ${}');
       // uuid = restoken['authorization'];
       _isLoggedIn = true;
       return {'success': true, 'auth': restoken['authorization']};
     } else {
       final Map<String, dynamic> resBody = jsonDecode(response.body);
       final String errorMsg = resBody['data']['msg'];
       print('Request failed with status: $errorMsg');
       return {'success': false, 'message': errorMsg};
     }
   } catch (e) {
     print('Request error failed: $e');
     return {'success': false, 'message': 'Failed to connect to the server.'};
   }
  }

  Future<Map<String, dynamic>> login(String email, String password) async {
   try {
     const String url = '$ip/auth/signup/Login';
     final Map<String, String> headers = {'Content-Type': 'application/json'};
     final http.Response response = await http.post(
       Uri.parse(url), // Replace 'url' with your login API endpoint
       headers: headers,
       body: jsonEncode({
         'email': email,
         'password': password,
       }),
     );
     if (response.statusCode == 200) {
       final Map<String, dynamic> responseData = jsonDecode(response.body);
       _isLoggedIn = true;
       uuid = responseData['authorization'];
       name = responseData['name'];
       email = responseData['email'];
       return {'success': true};
     } else {
       final Map<String, dynamic> errorData = jsonDecode(response.body);
       final String errorMsg = errorData['msg'];
       return {'success': false, 'message': errorMsg};
     }
   } catch (e) {
     return {'success': false, 'message': 'Failed to connect to the server.'};
   }
  }
```

```
  Future<Map<String, dynamic>>
OtpVerify(String otp, String
token) async {
   try {
     const String url =
'$ip/auth/signup/otp';
     final Map<String, String>
headers = {'Content-Type':
'application/json'};
     final http.Response response =
await http.post(
       Uri.parse(url), // Replace 'url'
with your login API endpoint
       headers: headers,
       body: jsonEncode({
         'email': token,
         'password': otp,
       }),
     );
     print(uuid);

print(jsonDecode(response.body));
     if (response.statusCode ==
200) {
       // final Map<String,
dynamic> responseData =
jsonDecode(response.body);
       // _isLoggedIn = true;
       // uuid =
responseData['authorization'];

print(jsonDecode(response.body));
       return
jsonDecode(response.body);
     } else {
       final Map<String, dynamic>
errorData =
jsonDecode(response.body);
       final String errorMsg =
errorData['message'];
       return {'status': false,
'message': errorMsg};
     }
   } catch (e) {
     print('here');
     print(e);
     return {'success': false,
'message': 'Failed to connect to the
server.'};
   }
  }

  Future<void> logout() async {
   // Logic to log out user
   _isLoggedIn = false;
  }
}
```

Server File

```
const express = require('express');
const router = express.Router();
const db =
require('../database/db.js')
const token =
require('../utility/jwt.js')

router.use('/signup', (req, res, next)
=> {
   req.body.createdAt = new
Date().toLocaleString('en-US', {
timeZone: 'Asia/Kolkata' })
   req.body.profileUpdate = false
   if (!!!req.body.phoneNo) {
      req.body.phoneNo = 0;
   }
   next()
})

// Define routes for /auth/signup
router.post('/signup', async (req,
res) => {
   console.log(req);
   console.log('enter');
   try {
      console.log(req.body);
      console.log('resquest');
      let result = await
db.insert(req.body);
      if (result.status) {
```

```
            let tokenRes =
token.generateToken(req.body.em
ail)
            res.status(200)
                //
.setHeader('authorization', 'Bearer '
+ tokenRes.data)
                .json({
                    status: true,
                    data: result,
                    authorization:
tokenRes.data
                })
        } else {
            console.log("errorr");
            res.status(500).json({ data:
result })
        }
        // Handle signup logic here
        // res.status(200).json({
        //     message: 'Signup
successful',
        //     res: result
        // });
    } catch (error) {
        console.log('Error in signup:',
error);
        res.status(500).json({
            message: 'Error in signup
process',
            error: error.message
        });
    }
});



router.post('/login', async (req, res)
=> {
    // req.body
    console.log('request');
    try {
        // console.log(req.body);
        let result = await
db.login(req.body.email,
req.body.password);
        // console.log(result);
        if (result.status) {
            let tokenRes =
token.generateToken(req.body.em
ail)
            // console.log(tokenRes);
            res.status(200).json({
                msg: "login",
                authorization:
tokenRes.data,
                name : result.name,
                email:result.email
            })
        }
        else {
            res.status(500).json({
                msg: result.msg
            })
        }
        // res.status(200).json({ data:
result })
    } catch (error) {
        console.log('error in login ' +
error);
        res.status(500).json({
            msg: "Server is Down. \/n
try after some time."
        })
    }
})

router.post('/otp', async (req, res)
=> {
    // req.body
    const { email, password } =
req.body
    // console.log(req.body);
    // let verify = await
token.verifyToken(jwt)
    const jwtToken = await
token.verifyToken(email);

    console.log("jwtToken");
    console.log(jwtToken.email);
```

```
console.log(email,jwtToken.passw
ord);

   if (jwtToken.status) {
      const result = await
db.verifyOTP(jwtToken.email,
password)
      console.log("after db")
      console.log(result)
      if (result.status) {
         res.json({ status: true,
message: result.msg })
      } else {
         res.json({ status: false,
message: result.msg })
         // res.json({ data:
jwtToken.email, status:
jwtToken.status, message:
jwtToken.msg })
      }
   } else {
      res.status(500).json({
         status: false, message:
jwtToken.msg
      })
   }
   // console.log('request');
   // try {
   //    // console.log(req.body);
   //    let result = await
db.login(req.body.email,
req.body.password);
   //    // console.log(result);
   //    if (result.status) {
   //       let tokenRes =
token.generateToken(req.query.em
ail)
   //       console.log(tokenRes);
   //       res.status(200).json({
   //          msg: "login",
   //          authorization:
tokenRes.data
   //       })
   //    }
   //    else {
   //       res.status(500).json({
   //          msg: result.msg
   //       })
   //    }
   //    // res.status(200).json({
   data: result })
   // } catch (error) {
   //    console.log('error in login '
   + error);
   //    res.status(500).json({
   //       msg: "Server is Down.
   \n try after some time."
   //    })
   // }
   })



   router.post('/ai', (req, res) => {
      res.send('good')
   })
   router.get('/verifyJWT', async (req,
   res) => {
      const authHeader =
   req.headers['authorization'];
      let jwt = authHeader
      let verify = await
   token.verifyToken(jwt)
      res.json({ data: verify.data,
   status: verify.status, msg:
   verify.msg })
   })
   module.exports = router;
```

JWT

```
const jwt = require('jsonwebtoken');
require('dotenv').config();

// Function to generate JWT token
function generateToken(email) {
   const payload = {
      email: email,
```

```
    // Set expiration time to 10
minutes (600 seconds)
    exp: Math.floor(Date.now() /
1000) + (100 * 60)
  };
  const token = jwt.sign(payload,
process.env.KEY)
  // Sign the JWT token with a secret
key
  return { data: token, status: true };
}
function verifyToken(token) {
  try {
    // Verify the token and decode its
payload
    const decoded = jwt.verify(token,
process.env.KEY);
    // Check if the token is expired
    if (decoded.exp < Date.now() /
1000) {
      return { status: false, msg:
'Token Expired' };
    }
    // Extract the email from the
decoded payload
    const email = jwt.decode(token);

    // console.log('jwt ');
    // console.log(email);
    // Return decoded data along with
email
    return {email: email.email,
status: true , data :email};
  } catch (error) {
    return { status: false, msg:
'Invalid Token' };
  }
}

module.exports = {
  generateToken,
  verifyToken
}
```

3.5 FINAL PROTOTYPE

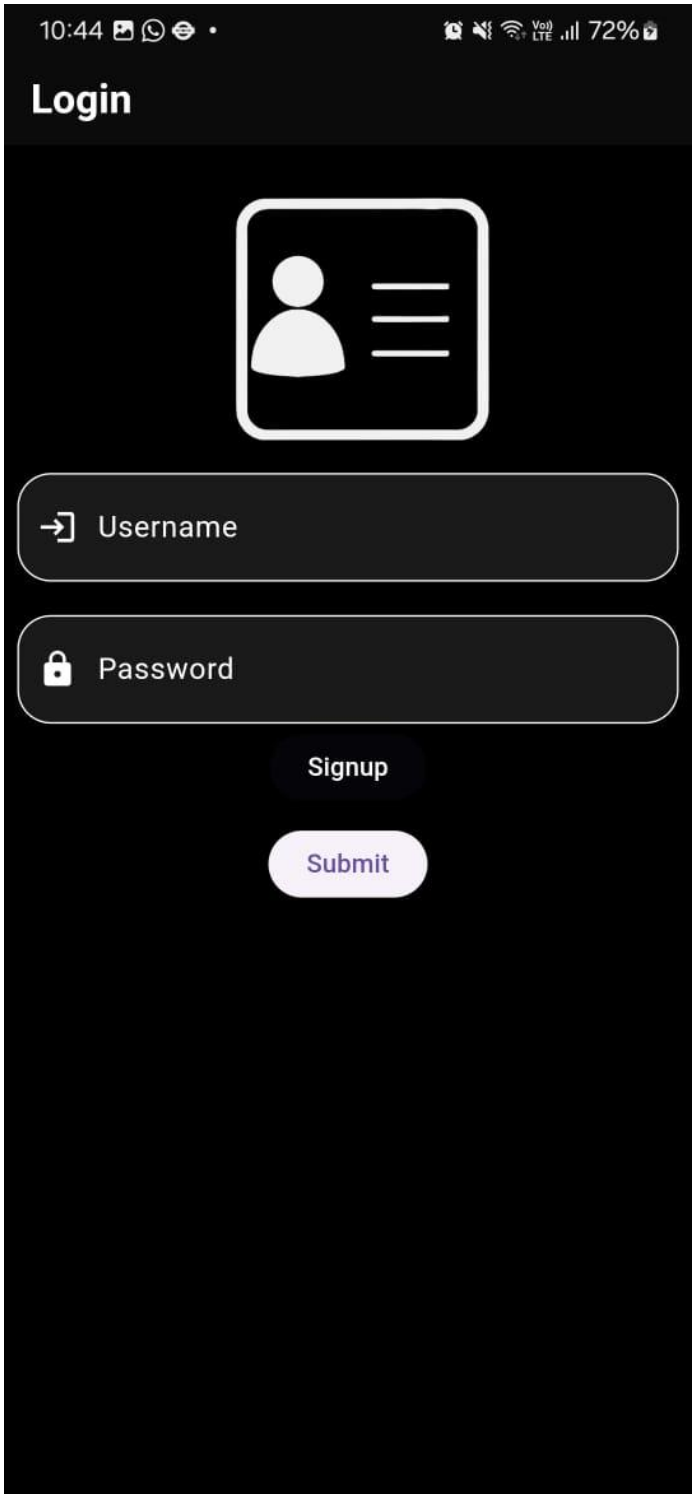

Figure 3.5.1: Mobile Login

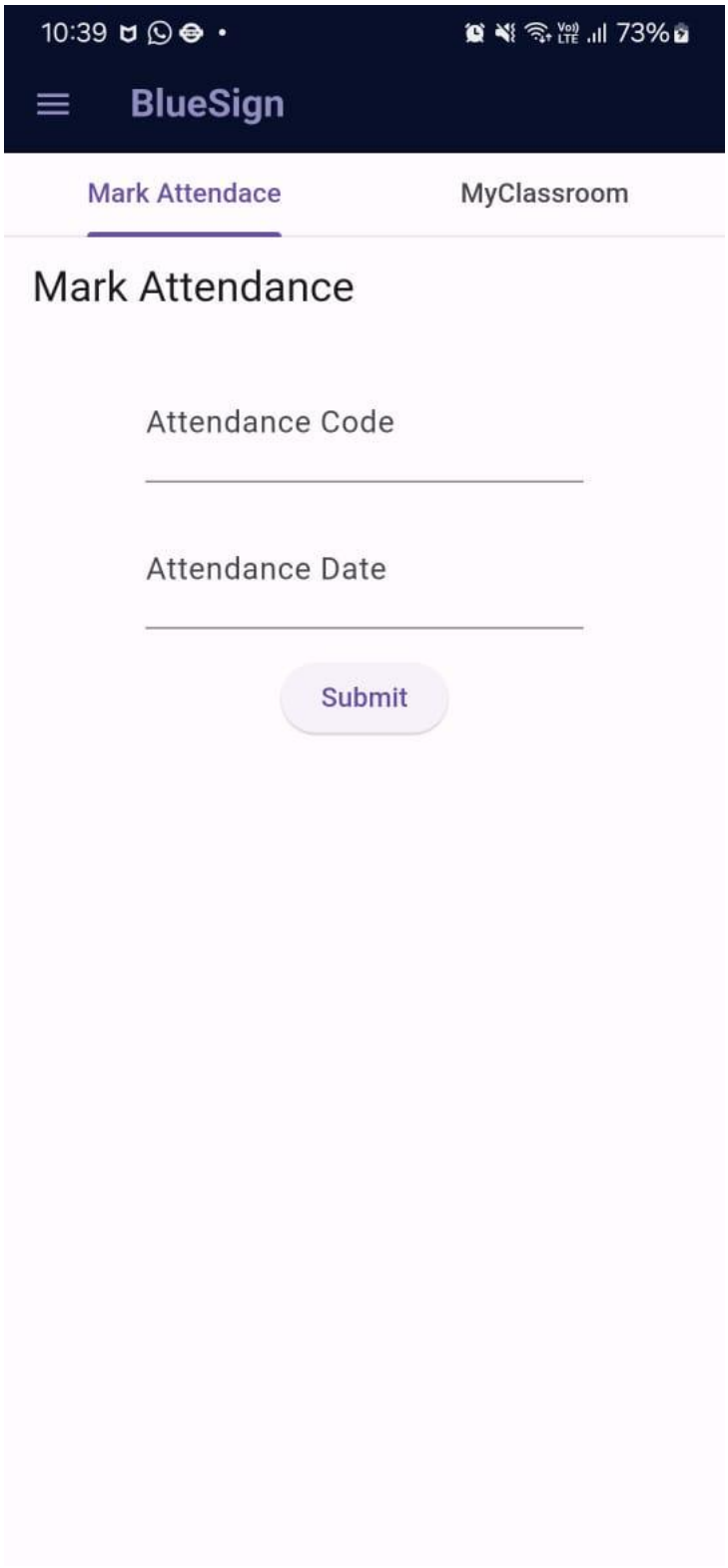

Figure 3.5.2: Creating Attendence

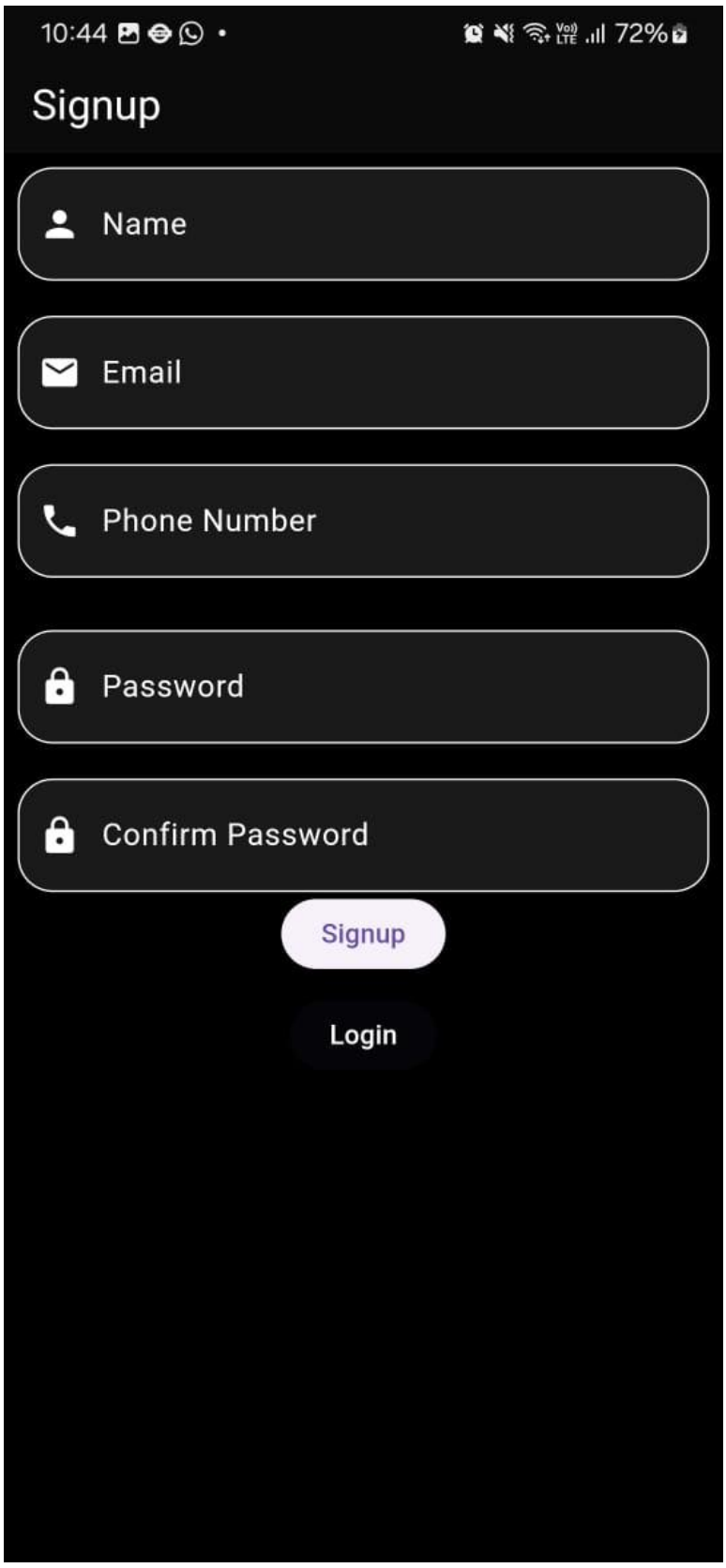

Figure 3.5.2: Mobile Sign-Up

3.6 CONCLUSION AND FUTURE SCOPE OF THE PROJECT

3.6.1 CONCLUSION:

In conclusion, the development and implementation of the innovative attendance management solution represent a significant milestone in addressing the shortcomings of manual tracking methods prevalent in organizations. Through the utilization of modern technologies such as the Flutter framework and Bluetooth integration, we have created a user-friendly mobile application that empowers employees to mark attendance effortlessly while providing administrators with efficient tools to manage attendance records in real-time.

The robust dashboard interface offers administrators unprecedented visibility into attendance status, enabling informed decision-making and proactive intervention when necessary. Moreover, our steadfast commitment to data security and privacy ensures that sensitive attendance records are safeguarded through rigorous encryption protocols and compliance with regulatory standards.

By prioritizing scalability and customization, we have developed features that cater to the diverse needs of organizations, including large institutions with extensive employee bases. Thorough testing and evaluation have ensured the reliability, accuracy, and seamless functionality of the solution across different devices and platforms.

Comprehensive documentation and training resources facilitate the adoption and implementation of the attendance management solution by organizations, empowering users to maximize the benefits of the system effectively. Furthermore, our commitment to gathering feedback and iteratively enhancing the application ensures its continuous improvement to meet evolving needs and preferences.

In essence, the attendance management solution represents a paradigm shift in how organizations track and manage attendance, offering tangible benefits in terms of time savings, accuracy, and administrative efficiency. As we move forward, we remain dedicated to supporting organizations in their journey towards modernizing their attendance tracking processes and embracing the transformative power of technology.

3.6.2 FUTURE SCOPE OF THE PROJECT:

The attendance management solution, developed with a focus on modernizing traditional tracking methods, lays a strong foundation for future integration with Artificial Intelligence (AI) and Human Resource Management (HRM) systems. This integration aims to enhance efficiency, accuracy, and strategic decision-making within organizations. Here's a detailed description of how AI and HRM can be integrated into the existing project:

1. **AI-Powered Attendance Tracking**:
    - Implement AI algorithms to automate attendance tracking processes further. This could involve facial recognition technology to identify employees as they enter the

premises or advanced biometric authentication methods for seamless and secure attendance marking.

- Utilize machine learning algorithms to analyze historical attendance data and predict future attendance patterns. This predictive analysis can help administrators anticipate attendance trends, plan resources more effectively, and identify potential issues before they arise.

2. **Employee Performance Analytics**:
    - Integrate AI-driven analytics into the attendance management solution to assess employee performance based on attendance records, productivity metrics, and other relevant factors.
    - Use predictive analytics to identify correlations between attendance patterns and performance outcomes, enabling HR managers to make data-driven decisions regarding employee evaluations, promotions, and training initiatives.
3. **Personalized HR Insights**:
    - Leverage AI-powered algorithms to generate personalized insights for employees and managers based on attendance data, performance metrics, and organizational objectives.
    - Provide employees with personalized recommendations for improving attendance, time management, and productivity, enhancing their overall engagement and performance within the organization.
    - Empower HR managers with actionable insights and recommendations for optimizing workforce management strategies, fostering a more productive and engaged workforce.
4. **Automated HR Processes**:
    - Integrate AI-driven chatbots or virtual assistants into the HRM system to automate routine administrative tasks such as leave management, attendance tracking, and employee inquiries.
    - Implement natural language processing (NLP) capabilities to enable employees to interact with the HRM system conversationally, simplifying the process of accessing information and completing HR-related tasks.
5. **Predictive HR Analytics**:
    - Develop predictive HR analytics models using AI and machine learning techniques to forecast future workforce trends, attrition rates, and talent acquisition needs based on historical attendance and performance data.
    - Enable HR managers to proactively address potential HR challenges, such as employee turnover or talent shortages, by identifying early warning signs and implementing targeted interventions.

3.7 CONSTRAINTS FOR REAL TIME DEPLOYMENT

Deploying the task management system prototype in real-time may encounter several constraints. Firstly, infrastructure limitations could arise, such as insufficient server resources or bandwidth, impacting scalability and reliability under varying load conditions. Additionally, ensuring robust security measures to protect user data and privacy may pose challenges, including implementing encryption protocols and complying with data protection regulations. Integration with existing tools and services may also present constraints, particularly regarding compatibility and data migration. Furthermore, optimizing system performance to ensure responsiveness and reliability, especially during peak usage periods, may face constraints related to resource allocation and database query optimization. Finally, encouraging user adoption and overcoming potential resistance to change could be a constraint, requiring effective communication and training strategies to promote acceptance and utilization of the new system.

3.8 REFERENCES

1. Patel, Ravi. "10 Best Practices for Building a Attendence Management System." Blog post, TaskManagement.com, May 2021. [Online]. Available: https://www.taskmanagement.com/blog/10-best-practices-building-task-management-system

2. "Building a Attendence Management System with React.js and Node.js." YouTube video tutorial series, CodeWithJosh, 2022. [Online]. Available: https://www.youtube.com/playlist?list=PLGNSioXgrIEfBNd6R21ZuYJRhg8pAcEma

3. Smith, John. "Effective Task Mansagement Strategies." Journal of Productivity Research, vol. 25, no. 2, 2020, pp. 45-62.