

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANA SANGAMA, BELAGAVI – 590 018



**An Internship Project Report
on**

Attendance Tracker UI

Submitted in partial fulfillment of the requirements for the VIII Semester of
degree of **Bachelor of Engineering in Information Science and Engineering** of
Visvesvaraya Technological University, Belagavi

by

Kayak V Gornale

1RN18IS060

Under the Guidance of

Mr. T S Bhagavath Singh

Associate Professor

Department of ISE



ESTD:2001

An Institute with a Difference

Department of Information Science and Engineering

RNS Institute of Technology

**Dr. Vishnuvaradhan Road, Rajarajeshwari Nagar post,
Channasandra, Bengaluru-560098**

2021-2022

RNS INSTITUTE OF TECHNOLOGY

Dr. Vishnuvaradhan Road, Rajarajeshwari Nagar post,
Channasandra, Bengaluru - 560098

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING



CERTIFICATE

Certified that the Internship work entitled *Attendance Tracker UI* has been successfully completed by **Kayak V Gornale (1RN18IS060)** a bonafide student of **RNS Institute of Technology, Bengaluru** in partial fulfillment of the requirements of 8th semester for the award of degree in **Bachelor of Engineering in Information Science and Engineering of Visvesvaraya Technological University, Belagavi** during academic year **2021-2022**. The internship report has been approved as it satisfies the academic requirements in respect of internship work for the said degree.

Mr. T S Bhagavath Singh
Internship Guide
Associate Professor
Department of ISE

Dr. Suresh L
Professor and HoD
Department of ISE
RNSIT

Dr. M K Venkatesha
Principal
RNSIT

Name of the Examiners

External Viva

Signature with Date

1. _____

1. _____

2. _____

2. _____

DECLARATION

I, **KAYAK V GORNALE [USN: 1RN18IS060]** student of VIII Semester BE, in Information Science and Engineering, RNS Institute of Technology hereby declare that the Internship work entitled *Attendance Tracker UI* has been carried out by us and submitted in partial fulfillment of the requirements for the *VIII Semester degree of Bachelor of Engineering in Information Science and Engineering of Visvesvaraya Technological University, Belagavi* during academic year 2021-2022.

Place : Bengaluru

Date :

KAYAK V GORNALE

(1RN18IS060)

Abstract

Classroom management is one of the key areas where a lot of time is wasted. This time could have otherwise been used effectively for a productive study time. Replacing these manual tasks of attendance tracking, management and feedback collection with some comprehensive apps can save a lot of precious teaching time.

Attendance Tracker is a system that deals with the maintenance of a student's attendance details. It generates attendance of the student on the basis of presence in class. Each student is given a username and password. The student can log in into his/her account and check their attendance.

ACKNOWLEDGMENT

At the very onset I would like to place our gratefulness to all those people who helped me in making the Internship a successful one.

Coming up, this internship to be a success was not easy. Apart from the sheer effort, the enlightenment of the very experienced teachers also plays a paramount role because it is they who guided me in the right direction.

First of all, I would like to thank the **Management of RNS Institute of Technology** for providing such a healthy environment for the successful completion of internship work.

In this regard, I express sincere gratitude to our beloved Principal **Dr. M K Venkatesha**, for providing us all the facilities.

We are extremely grateful to our own and beloved Professor and Head of Department of Information science and Engineering, **Dr. Suresh L**, for having accepted to patronize me in the right direction with all her wisdom.

We place our heartfelt thanks to **Mr. T S Bhagavath Singh**, Associate Professor, Department of Information Science and Engineering for having guided internship and all the staff members of the department of Information Science and Engineering for helping at all times.

I thank **Mr. Akshay D R, Co-Founder & CEO, Enmaz Engineering Services Pvt.Ltd** for providing the opportunity to be a part of the Internship program and having guided me to complete the same successfully.

I also thank our internship coordinator **Dr. R Rajkumar**, Associate Professor, Department of Information Science and Engineering. I would thank my friends for having supported me with all their strength and might. Last but not the least, I thank my parents for supporting and encouraging me throughout. I have made an honest effort in this assignment.

Kayak V Gornale

TABLE OF CONTENTS

Abstract	i
Acknowledgment	ii
Contents	iii
List of figures	v
List of Abbreviations	vi
1. Introduction	1
1.1 Background	1
1.2 Existing System	1
1.3 Proposed System	1
2. Literature Survey	2
2.1 Attendance Tracker	2
2.1.1 Introduction	2
2.1.2 Proposed System	3
3. System Design	4
3.1 Widget Tree	4
3.1.1 Login Screen Widget Tree	4
3.1.2 Student Attendance details Widget Tree	5
4. Implementation	6
4.1 Requirement specification	6
4.1.1 Hardware Requirements	6
4.1.2 Software Requirements	6
4.1.3 Flutter	6
4.1.3.1 Dart Platform	6
4.1.3.2 Flutter Engine	7
4.1.3.3 Foundation Library	7
4.1.3.4 Design Specific Widgets	7
4.2 Discussion of Code Segments	7
4.2.1 main.dart	7
4.2.2 Login Screen	8
4.2.3 Attendance details	15

5. Testing	23
5.1 Introduction	23
5.2 Levels of Testing	23
5.2.1 Unit Testing	23
5.2.2 Integration Testing	24
5.2.3 System testing	24
5.2.4 Validation testing	24
5.2.5 Output testing	24
5.2.6 User acceptance testing	24
6. Results	25
7. Conclusion and future work	30
7.1 Conclusion	30
7.2 Future work	30
References	31

List of Figures

Fig. No.	Figure Description	Page No.
3.1.1	Login Screen Widget Tree	4
3.1.2	Student Attendance details Widget Tree	5
6.1	Login Screen	25
6.2	Loading Screen	26
6.3	Student Attendance details Screen 1	27
6.4	Student Attendance details Screen 2	28
6.5	Subject wise Attendance Screen	29

List of Abbreviations

UI	User Interface
API	Application Programming Interface
CPU	Central Processing Unit
HDD	Hard Disk Drive
RAM	Random Access Memory
IDE	Integrated Development Environment

1. Introduction

1.1 Background

Attendance has been given utmost importance by schools, colleges and universities. But for a student it can get tedious to keep track of their attendance across all the subjects and to determine the subjects in which they have lesser attendance.

Attendance Tracker UI is an application developed for daily student attendance in schools, colleges and other institutes. It facilitates to access the attendance information of a particular student in a particular class. Each student is provided with a username and password. This allows the student to log in into their profile and have access to their subject wise attendance status.

1.2 Existing System

In the existing system, the students have to manually contact the teacher to get their attendance status. It will be tedious for the teacher to when a large number of students want to check their attendance. The human effort is more here.

1.3 Proposed System

To overcome the drawbacks mentioned above, the proposed system has been developed. This project has been developed so that a student can easily track their attendance. The interface is easy to understand user friendly.

2. Literature Survey

2.1 Attendance Tracker

Students' attendance taking and tracking are important in order to monitor students' performance in class. More often than not, students' performance is closely related to their attendance. Good attendance usually leads to good performance and vice versa. Therefore, any problems related to students' attendance should be identified as early as possible so that appropriate measures can be taken to address them. However, tracking students' attendance, especially if done manually, can be tedious and time consuming, especially for classes with large number of students. Not to mention issues related to attendance taking such as signatures forgery where other students are signing on behalf of their absence friends. To address this issue, a unique and secure attendance tracking system is proposed.

The system automates most of the steps involved in tracking students' attendance. To address the issue of signature forgery, secret code using MD5 hashing algorithm is implemented as part of the system so that each student will be given a unique code each day to be used for signing attendance. Implementation of the system shows that the time taken to track students' attendance using this system can be significantly reduced and the secret code is able to prevent signature forgery amongst students.

2.1.1 Introduction

The method, witness's huge notice and a wealth of assure in content-based image recovery as a rising technology. It also a horizontal way for a huge number of new techniques and systems, get various new citizens include. In this piece, we survey almost 300 new hypothetical and experimental charity in the existing decade related to image recovery and regular image clarification. We also discuss significant challenges involved in the difference of existing image recovery techniques to build systems that can be useful in the genuine world. In retrospect of what has been achieved so far, we also work out what the prospect may hold for image recovery study. Predictable methods of image revival require that metadata is connected with the image, usually known as keywords. Though some content-based image retrieval systems utilize together semantic and prehistoric attributes to relation search principle, history has proven that it is tricky to remove linguistic in sequence from a 2D picture. In this observe,

activity theory is used as a foundation to express how semantic in sequence can be retrieved from objects recognized in a picture. Via a picture segmentation method. By The Berkeley Digital Library Project, and merge it with, a high-level accepting of the pictures can be established Content-Based Image Retrieval has become one of the popular most research areas. Many diagram attribute representations contain been explored and many systems build. While, this research information found the foundation of satisfied based image recovery, the kindness of the future approaches is incomplete. Specially, these efforts have comparatively overlooked two different characteristics of systems the space between towering level concepts and low-level skin texture bias of human compassion of visual content. Which electively takes into account the above two uniqueness in CBIR. During the recovery process, the user's high-level query and insight partisanship are captured by dynamically updated weights based on the user's advice. The provisional results over more than 70,000 images show that the future approach greatly reduces the user's effort of composing a doubt and capture the users in sequence.

2.1.2 Proposed System

The whole session attendance is stored in register and at the end of the session the reports are carried out. We are not interested in generating report in the middle of the session or as per the requirement because it takes more time in calculation. At the end of session, the students who don't have 85% attendance gets a notice.

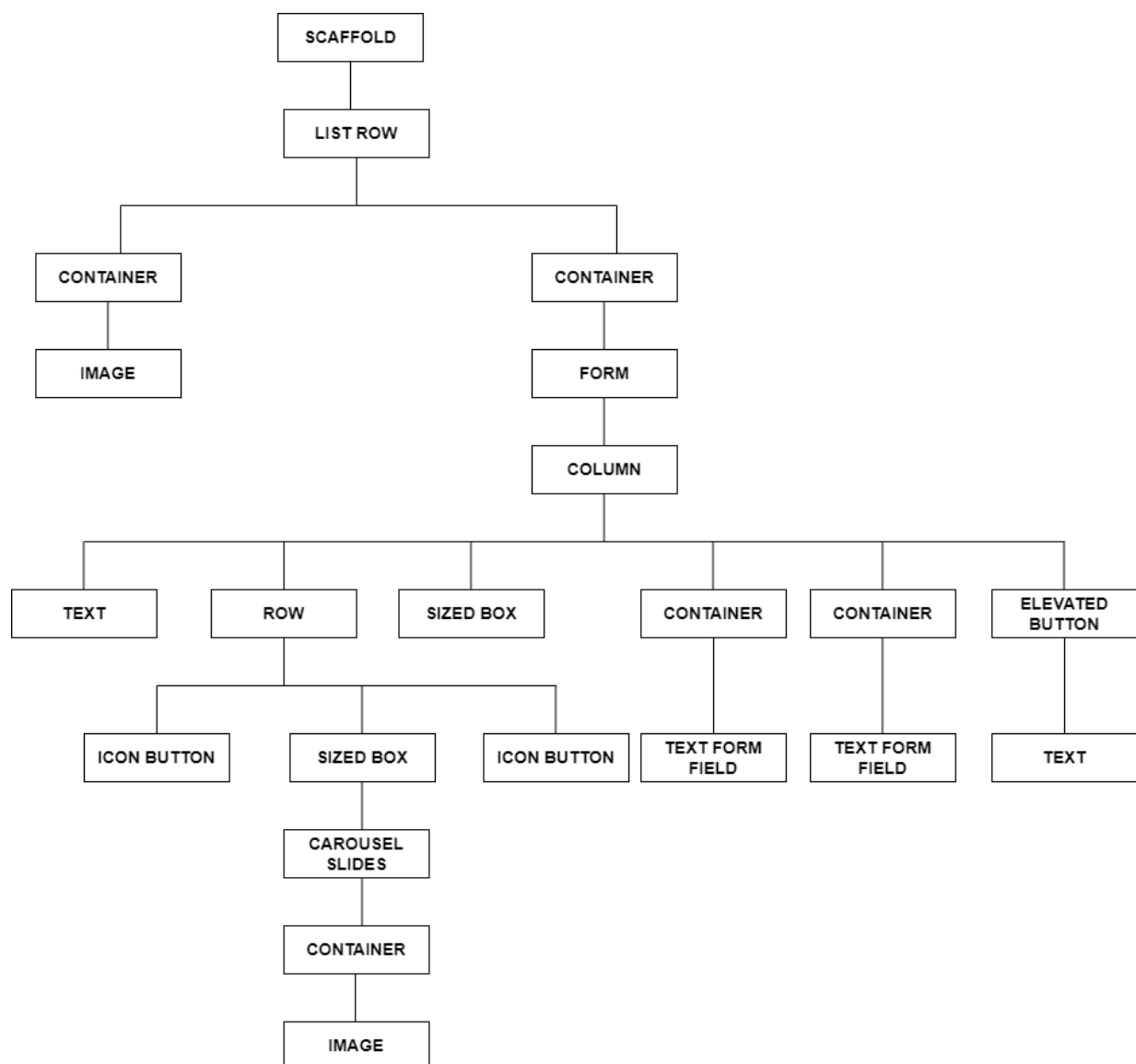
- User Friendly - The proposed system is user friendly because the retrieval and storing of data is fast and data is maintained efficiently. Moreover, the graphical user interface is provided in the proposed system, which provides user to deal with the system very easily.
- Reports are easily generated: reports can be easily generated in the proposed system so user can generate the report as per the requirement (monthly) or in the middle of the session. User can give the notice to the students so he/she become regular.
- Very less paper work: The proposed system requires very less paper work. All the data is feted into the computer immediately and reports can be generated through computers. Moreover, work becomes very easy because there is no need to keep data on papers.
- Computer operator control: Computer operator control will be there so no chance of errors. Moreover, storing and retrieving of information is easy. So, work can be done speedily and in time.

3. System Design

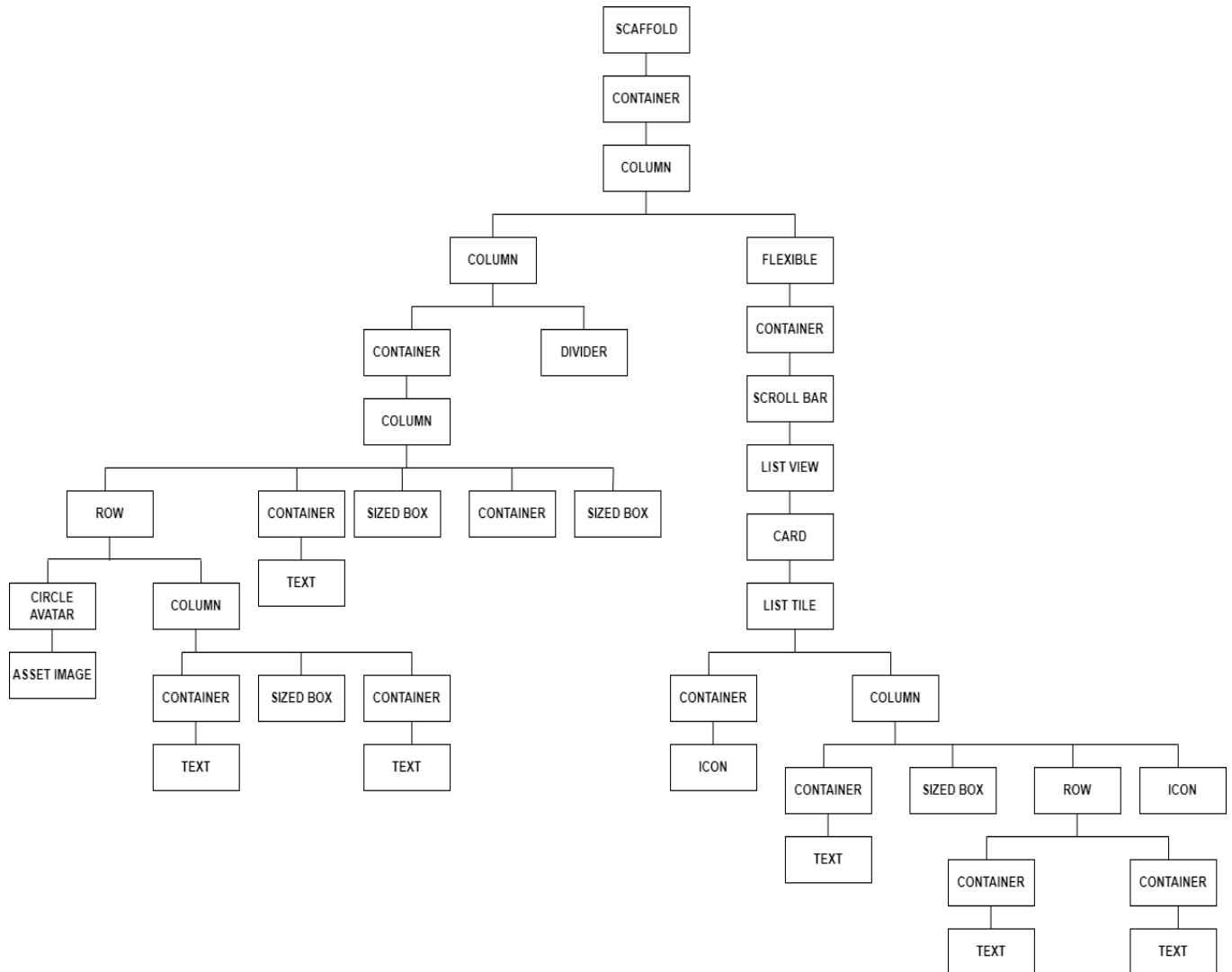
3.1 Widget Tree

The widget tree is how developers create their user interface; developers position widgets within each other to build simple and complex layouts. Since just about everything in the Flutter framework is a widget, and as they start nesting them, the code can become harder to follow. Widget trees help the developers to visualize a screen in a better way and have a clear picture of the screen that is being developed.

3.1.1 Login Screen Widget Tree



3.1.2 Student Attendance details Widget Tree



4.Implementation

4.1 Requirement Specifications

4.1.1 Hardware Requirements

- **CPU:** Pentium processor and above
- **RAM:** 4 GB
- **HDD:** 40 GB

4.1.2 Software Requirements

- **Operating System:** Windows 8 and above
- **Editor:** Visual Studio Code
- **IDE:** VS Code
- **Front-end Language:** Dart
- **Minimum Android Version :** Froyo (API Level 8)
-

4.1.3 Flutter

Flutter is an open-source UI, software development kit created by Google. It is used to develop cross platform applications for Android, iOS, Linux, Mac, Windows, Google Fuchsia, Web Platform and the web from a single codebase. The major components of Flutter include:

- Dart Platform
- Flutter Engine
- Foundation Library
- Design-specific widgets
- Flutter Development Tools (DevTools)

4.1.3.1 Dart Platform

Flutter apps are written in the Dart language and make use of many of the language's more advanced features.

On Windows, macOS, and Linux Flutter runs in the Dart virtual machine, which features a just-in-time execution engine. While writing and debugging an app, Flutter uses Just In Time

compilation, allowing for "hot reload", with which modifications to source files can be injected into a running application. Flutter extends this with support for stateful, hot reload, where in most cases changes to source code are reflected immediately in the running app without requiring a restart or any loss of state

4.1.3.2 Flutter Engine

Flutter's engine, written primarily in C++, provides low-level rendering support using Google's Skia graphics library. Additionally, it interfaces with platform Specific SDKs such as those provided by Android and iOS. The Flutter Engine is a portable runtime for hosting Flutter applications. It implements Flutter's core libraries, including animation and graphics, file and network I/O, accessibility support, plugin architecture, and a Dart runtime and compile toolchain.

4.1.3.3 Foundation Library

The Foundation library, written in Dart, provides basic classes and functions that are used to construct applications using Flutter, such as APIs to communicate with the engine.

4.1.3.4 Design Specific Widgets

The Flutter framework contains two sets of widgets that conform to specific design languages: Material Design widgets implement Google's design language of the same name, and *Cupertino* widgets implement Apple's iOS Human Interface Guidelines.

4.2 Discussion of Code Segments

4.2.1 main.dart

```
import 'package:flutter/material.dart';

import 'package:rnsit_college_app/screens/login.dart';

void main() {

  runApp(const MyApp());

}

class MyApp extends StatelessWidget {

  const MyApp({Key? key}) : super(key: key);
```



```
@override

Widget build(BuildContext context) {

  return const MaterialApp(

    title: "RNS Institute of Technology"

    themeMode: ThemeMode.light,

    debugShowCheckedModeBanner: false,

    home: LaunchScreen());

}

}

class LaunchScreen extends StatefulWidget {

  const LaunchScreen({ Key? key }) : super(key: key);

  @override

  _LaunchScreenState createState() => _LaunchScreenState()

  class _LaunchScreenState extends State<LaunchScreen> {

    @override

    Widget build(BuildContext context) {

      return Login(); }

  }
```

4.2.2 Login Screen

```
import 'dart:ui';

import 'package:flutter/material.dart';

import 'package:rnsit_college_app/screens/loading.dart';

import 'package:carousel_slider/carousel_slider.dart';

import 'package:rnsit_college_app/screens/student_home_page.dart';
```

```
import 'package:rnsit_college_app/service/authorization.dart';

import 'package:rnsit_college_app/values/string_constants.dart';

import 'package:rnsit_college_app/values/theme.dart';

import 'package:rnsit_college_app/widgets/hyperlink.dart';

import 'package:rnsit_college_app/widgets/password_text_field_form.dart';

import 'package:rnsit_college_app/widgets/user_name_text_field_form.dart';

class LoginWidget extends StatefulWidget {

  const LoginWidget({Key? key}) : super(key: key)

  @override

  _LoginWidgetState createState() => _LoginWidgetState();

}

class _LoginWidgetState extends State<LoginWidget> {

  @override

  Widget build(BuildContext context) {

    return Container(

      margin: EdgeInsets.fromLTRB(MediaQuery.of(context).size.width * .1, 0,

        MediaQuery.of(context).size.width * .1, 0),

      decoration: BoxDecoration(

        color: Colors.white,

        border: Border.all(color: kThemeColor, width: 3.5),

        borderRadius: BorderRadius.circular(10)),

      child: const LoginForm();

    )

  }
```

```
}

class LoginForm extends StatefulWidget {

  const LoginForm({ Key? key }) : super(key: key)

  @override

  _LoginFormState createState() => _LoginFormState();

}

class _LoginFormState extends State<LoginForm> {

  final _formKey = GlobalKey<FormState>();

  List<Map> loginType = [

    {"type": "Student", "logo": kLogosStudentLogo, "labelText": "USN"},

    {"type": "Teacher", "logo": kLogosTeacherLogo, "labelText": "Email"}

  ];

  int index = 0;

  final controller = CarouselController();

  final userNameController = TextEditingController();

  final passwordController = TextEditingController();

  @override

  Widget build(BuildContext context) {

    return Form(

      key: _formKey,

      child: Column(

        children: [

          const SizedBox(height: 20),
```

```
FormHeader(loginType: loginType, index: index),

const SizedBox(height: 20),

Row(

  mainAxisAlignment: MainAxisAlignment.center,

  children: [

    IconButton(

      onPressed: () {

        controller.previousPage();

      },

      icon: Icon(Icons.arrow_left_outlined)),

    SizedBox(

      width: 70,

      height: 70,

      child: CarouselSlider.builder(

        carouselController: controller,

        itemCount: 2,

        itemBuilder: (context, index, realIndex) {

          final imageUrl = loginType[index]["logo"];

          this.index = index;

          return buildLoginImage(imageUrl, index);

        },

        options: CarouselOptions(

          onPageChanged: (index, reason) {

            changeLabel(index);

          },
```

```
        viewportFraction: 1, )), ),
    IconButton(
      onPressed: () {
        controller.nextPage();
      },
      icon: Icon(Icons.arrow_right_outlined)),
    ],),
    const SizedBox(height: 20),
    UserNameTextFieldForm(
      TextInputAction.next,
      userNameController,
      loginType[index]["labelText"],
      "Enter the " + loginType[index]["labelText"],
      "Please enter your " + loginType[index]["labelText"]),
    const SizedBox(height: 20),
    PasswordTextFieldForm(TextInputAction.next, passwordController),
    const SizedBox(height: 20),
    ElevatedButton(
      style: ButtonStyle(
        backgroundColor: MaterialStateProperty.all(kThemeColor)),
      onPressed: () async {
        if (_formKey.currentState!.validate()) {
          Scaffold.of(context).showSnackBar(
            const SnackBar(content: Text("Login Successful")));
          Navigator.of(context).push(
```

```
        MaterialPageRoute(builder: (context) => Loading()));

var authorized = await Authorization().checkAuthorization(

    loginType[index]["type"],

    userNameController.text,

    passwordController.text);

Navigator.pop(context);

if (authorized is! bool) {

Navigator.pushReplacement(

    context,

    MaterialPageRoute(

        builder: (context) =>

            StudentHomePage(authorized)));

    } } },

    child: Text("Submit")),

const SizedBox(height: 20),

Row(

    crossAxisAlignment: CrossAxisAlignment.center,

    mainAxisAlignment: MainAxisAlignment.spaceAround,

    children: [

        HyperlinkText("Signup", true, Loading()),

        HyperlinkText("Forgot Password", true, Loading()),

    ],

const SizedBox(height: 20),

],

));
```

```
}
```

```
Widget buildLoginImage(var urlImage, int index) => Container(
```

```
    color: kContentThemeColor,
```

```
    child: urlImage,
```

```
    width: 70,
```

```
    height: 70,
```

```
);
```

```
void changeLabel(int index) {
```

```
    setState(() {
```

```
        this.index = index;
```

```
    });
```

```
}
```

```
}
```

```
class FormHeader extends StatelessWidget {
```

```
    const FormHeader({
```

```
        Key? key,
```

```
        required this.loginType,
```

```
        required this.index,
```

```
    }) : super(key: key);
```

```
    final List<Map> loginType;
```

```
    final int index;
```

```
    @override
```

```
    Widget build(BuildContext context) {
```

```
return Text(  
  loginType[index]["type"] + " Login",  
  style: const TextStyle(  
    color: kThemeColor,  
    fontSize: 24,  
    fontWeight: FontWeight.bold,  
    decoration: TextDecoration.underline),  
);}}
```

4.2.3 Attendance details

```
import 'package:flutter/material.dart';  
import 'package:rnsit_college_app/values/theme.dart';  
import 'package:syncfusion_flutter_calendar/calendar.dart';  
import 'package:table_calendar/table_calendar.dart';  
  
class AttendanceDetails extends StatefulWidget {  
  Map studentData;  
  Map subjectData;  
  
  AttendanceDetails(  
    {required this.studentData, required this.subjectData, Key? key})  
    : super(key: key);  
  
  @override  
  _AttendanceDetailsState createState() => _AttendanceDetailsState();  
}  
  
getColor(int attendancePercentage) {  
  if (attendancePercentage >= 90) {  
    return Colors.green;
```



```
} else if (attendancePercentage >= 85) {  
    return Colors.yellow;  
}  
else {  
    return Colors.red;  
}  
}}  
  
class _AttendanceDetailsState extends State<AttendanceDetails> {  
  
    @override  
  
    Widget build(BuildContext context) {  
  
        Map subjectData = widget.subjectData;  
  
        return Column(  
  
            mainAxisAlignment: MainAxisAlignment.spaceEvenly,  
  
            children: [  
  
                Header(widget: widget, subjectData: subjectData),  
  
                AttendanceOverView(subjectData: subjectData),  
  
                Calendar(subjectData: subjectData)  
  
            ],);  
    }  
}  
  
class Calendar extends StatefulWidget {  
  
    Map subjectData;  
  
    Calendar({ Key? key, required this.subjectData }) : super(key: key);  
  
    @override  
  
    _CalendarState createState() => _CalendarState();  
  
}  
  
class _CalendarState extends State<Calendar> {  
  
    var calendarView = CalendarView.month;  
  
    @override
```

```
Widget build(BuildContext context) {  
  return Container(  
    height: 300,  
    padding: EdgeInsets.only(left: 25, right: 25),  
    child: SfCalendar(  
      backgroundColor: kContentThemeColor,  
      cellBorderColor: Colors.transparent,  
      headerHeight: 50,  
      viewHeaderHeight: 50,  
      viewHeaderStyle: ViewHeaderStyle(  
        dayTextStyle: TextStyle(fontSize: 17, color:kThemeColor)),  
      headerStyle: CalendarHeaderStyle(  
        textAlign: TextAlign.center,  
        backgroundColor: kThemeColor,  
      textStyle: TextStyle(color: kContentThemeColor, fontSize: 20)),  
      showDatePickerButton: true,  
      viewNavigationMode: ViewNavigationMode.snap,  
      showNavigationArrow: true,  
      view: calendarView,  
      initialDisplayDate: DateTime.now(),  
      monthViewSettings: MonthViewSettings(  
        monthCellStyle: MonthCellStyle(  
          backgroundColor: kContentThemeColor,  
          todayBackgroundColor: kContentThemeColor),  
        ),),);}}
```

```
class AttendanceOverView extends StatefulWidget {  
  Map subjectData;  
  
  AttendanceOverView({Key? key, required this.subjectData}) : super(key: key);  
  
  @override  
  _AttendanceOverViewState createState() => _AttendanceOverViewState();  
}  
  
class _AttendanceOverViewState extends State<AttendanceOverView> {  
  @override  
  
  Widget build(BuildContext context) {  
    Map subjectData = widget.subjectData;  
  
    return Container(  
      alignment: Alignment.center,  
  
      padding: EdgeInsets.only(left: 15, right: 15),  
  
      child: Row(  
        mainAxisAlignment: MainAxisAlignment.spaceEvenly,  
        children: [  
          Container(  
            decoration: BoxDecoration(  
              boxShadow: [  
                BoxShadow(  
                  blurRadius: 3, spreadRadius: .1, offset: Offset(0, 2)),  
                  color: kContentThemeColor,  
                  borderRadius: BorderRadius.circular(15)),  
                padding: EdgeInsets.all(10),  
                alignment: Alignment.center,
```

```
        child: RichText(  
          text: TextSpan(  
            children: [  
              TextSpan(  
                text: "Absent:\n",  
                style: TextStyle(fontSize: 17),  
              ),  
              TextSpan(text:  
subjectData["classesAbsent"].toString(),  
                style: TextStyle(fontSize: 30)),  
              TextSpan(text: " Days", style: TextStyle(fontSize: 8)),  
            ],  
            style: TextStyle(color: Colors.red),  
          )),  
        ),  
        SizedBox(  
          width: 20,  
        ),  
        Container(  
          decoration: BoxDecoration(  
            boxShadow: [BoxShadow(  
              blurRadius: 3, spreadRadius: .1, offset: Offset(0, 2)],  
              color: kContentThemeColor,  
              borderRadius: BorderRadius.circular(15)),  
            padding: EdgeInsets.all(10),
```

```
alignment: Alignment.center,

child: RichText(

  text: TextSpan(

    children: [

      TextSpan(

        text: "Present:\n",

        style: TextStyle(fontSize: 17),

      ),

      TextSpan(

        text: subjectData["classesAttended"].toString(),

        style: TextStyle(fontSize: 30)),

      TextSpan(text: " Days", style: TextStyle(fontSize: 8)),

    ],

    style: TextStyle(color: Colors.green),

  )),

  SizedBoxwidth: 20,

),

Container(

  decoration: BoxDecoration(

    boxShadow: [

      BoxShadow(

        blurRadius: 3, spreadRadius: .1, offset: Offset(0, 2)),

        color: kContentThemeColor,

        borderRadius: BorderRadius.circular(15)),

    padding: EdgeInsets.all(10),
```

```
        alignment: Alignment.center,

        child: RichText(

          text: TextSpan(

            children: [

              TextSpan(

                text: "Percent:\n",

                style: TextStyle(fontSize: 17),

              ),

              TextSpan(

                text: subjectData["attendancePercentage"].toString(),

                style: TextStyle(fontSize: 30)),

              TextSpan(text: " %", style: TextStyle(fontSize: 8)),

            ],

            style: TextStyle(

              color: getColor(subjectData["attendancePercentage"])),

            ), ) ],),

      );}}

class Header extends StatelessWidget {

  const Header({

    Key? key,

    required this.widget,

    required this.subjectData,

  }) : super(key: key);

  final AttendanceDetails widget;

  final Map subjectData;
```

```
@override

Widget build(BuildContext context) {

  return Column(

    children: [

      Container(

        padding: EdgeInsets.only(left: 25, right: 25),

        alignment: Alignment.center,

        child: Text(

          widget.subjectData["subjectName"],

          style: TextStyle(fontSize: 25),

          textAlign: TextAlign.center,

        )),

      SizedBox(

        height: 10,

      ),

      Container(

        padding: EdgeInsets.only(left: 25, right: 25),

        alignment: Alignment.center,

        child: Text(

          subjectData["subjectCode"],

          style: TextStyle(fontSize: 18),

        ))

    ],

  );

}
```

5. Testing

5.1 Introduction

Testing is a process of executing a program with the interest of finding an error. A good test is one that has high probability of finding the yet undiscovered error. Testing should systematically uncover different classes of errors in a minimum amount of time with a minimum number of efforts. Two classes of inputs are provided to test the process

- A software configuration that includes a software requirement specification, a design specification and source code.
- A software configuration that includes a test plan and procedure, any testing tool and test cases and their expected results.

5.2 Levels of Testing

5.2.1 Unit Testing

Unit testing is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output.

Unit testing is commonly automated, but may still be performed manually. The objective in unit testing is to isolate a unit and validate its correctness. A manual approach to unit testing may employ a step-by-step instructional document. The unit testing is the process of testing the part of the program to verify whether the program is working correctly or not. In this part the main intention is to check each and every input which we are inserting to our file. Here the validation concepts are used to check whether the program is taking the inputs in the correct format or not.

Unit testing may reduce uncertainty in the units themselves and can be used in a bottom-up testing style approach. By testing the parts of a program first and then testing the sum of its parts, integration testing becomes much easier. Unit test cases embody characteristics that are critical to the success of the unit.

5.2.2 Integration Testing

Integration testing is also taken as integration and testing this is the major testing process where the units are combined and tested. Its main objective is to verify whether the major parts of the program is working fine or not. This testing can be done by choosing the options in the program and by giving suitable inputs.

5.2.3 System Testing

System testing is defined as testing of a complete and fully integrated software product. This testing falls in black-box testing wherein knowledge of the inner design of the code is not a prerequisite and is done by the testing team. System testing is done after integration testing is complete. System testing should test functional and non-functional requirements of the software.

5.2.4 Validation Testing

In this, requirements established as part of software requirements analysis are validated against the software that has been constructed. Validation testing provides final assurance that software meets all functional, behavioral and performance requirements. Validation can be defined in many ways but a simple definition is that validation succeeds when software Function in a manner that can be reasonably by the customer.

1. Validation test criteria
2. Configuration review
3. Alpha and Beta testing (conducted by end user)

5.2.5 Output Testing

After preparing test data, the system under study is tested using the test data. While testing the system using test data, errors are again uncovered and corrected by using above testing and corrections are also noted for future use.

5.2.6 User Acceptance Testing

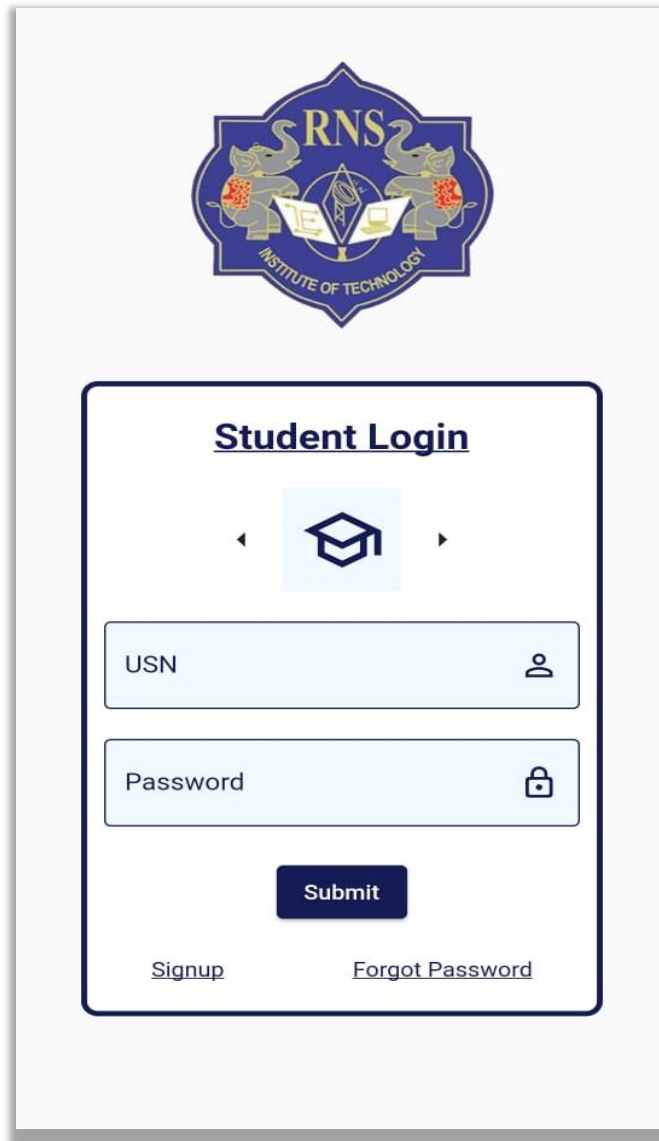
User acceptance testing is a type of testing performed by the end user or the client to verify/accept the software application to the production environment.

User Acceptance Testing is done in the final phase of testing.

6.Results

6.1 Login Screen

Fig 6.1 has 2 fields to enter User ID and Password where the user is required to enter the correct credentials to access their attendance details.



The image shows a web-based login interface for RNS Institute of Technology. At the top center is the RNS logo, which includes the letters 'RNS' and 'INSTITUTE OF TECHNOLOGY'. Below the logo is a white box with a dark blue border containing the following elements:

- The text **Student Login** in dark blue.
- A blue square icon containing a white graduation cap, flanked by left and right arrow symbols.
- A light blue input field labeled **USN** with a user icon on the right.
- A light blue input field labeled **Password** with a lock icon on the right.
- A dark blue button labeled **Submit**.
- Two links at the bottom: [Signup](#) and [Forgot Password](#).

Fig. 6.1 Login Screen

6.2 Loading Screen

Fig 6.2 shows When the user enters their user ID and Password this screen is displayed indicating that their information is loading.

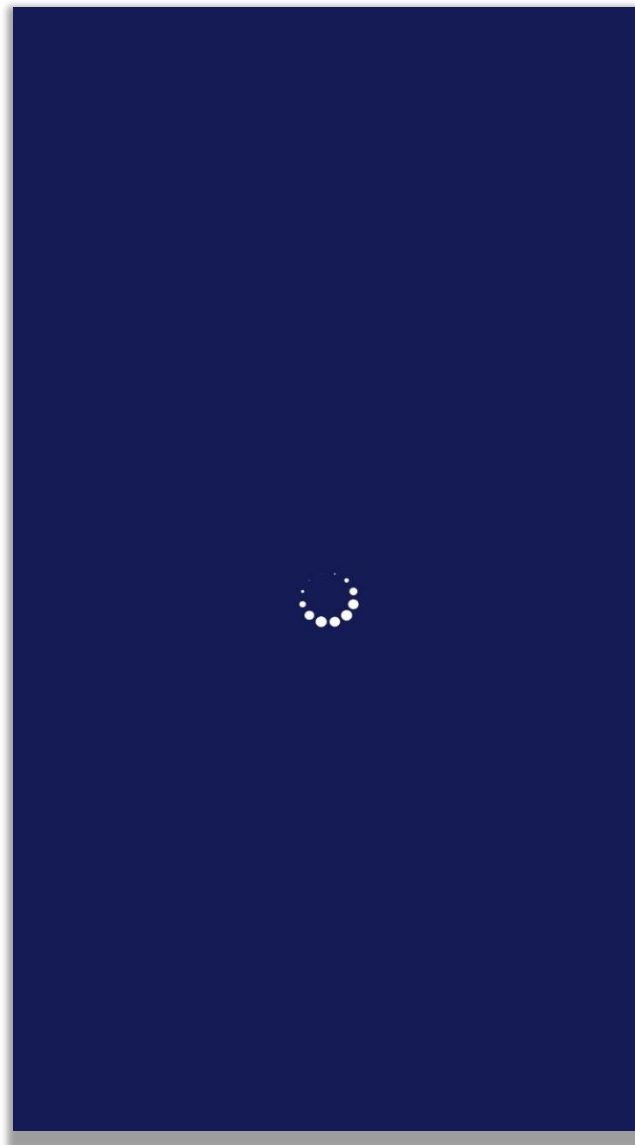


Fig. 6.2 Loading Screen

6.3 Student Attendance details screen 1

Fig 6.3 gives an overview of the attendance details of the student across all subjects.

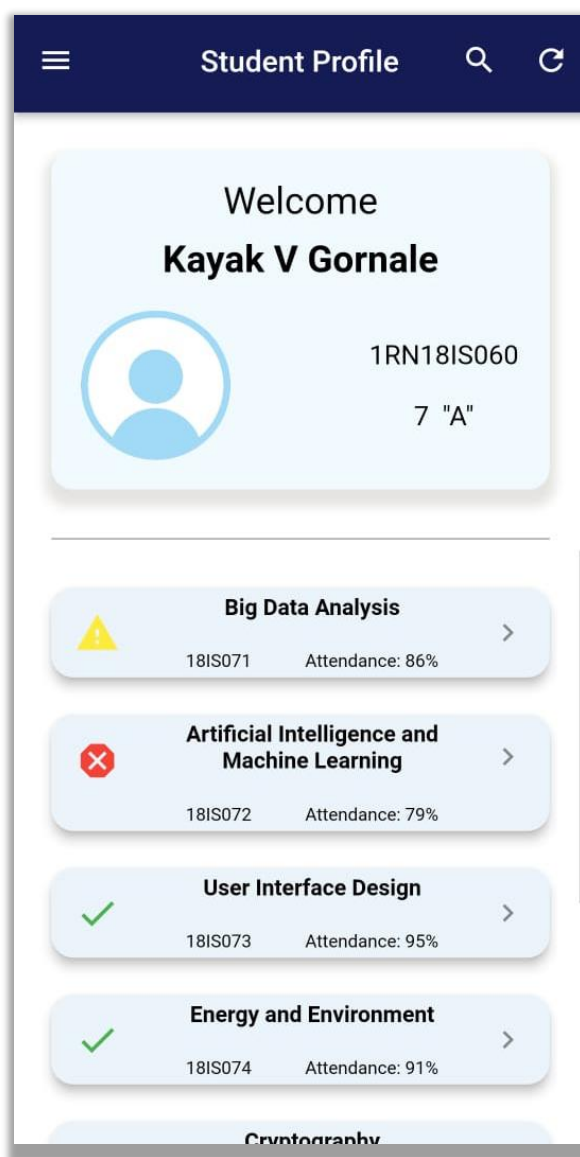


Fig. 6.3 Student Attendance details Screen 1

6.4 Student Attendance details screen 2

Fig 6.4 gives an overview of the attendance details of the student across all subjects.

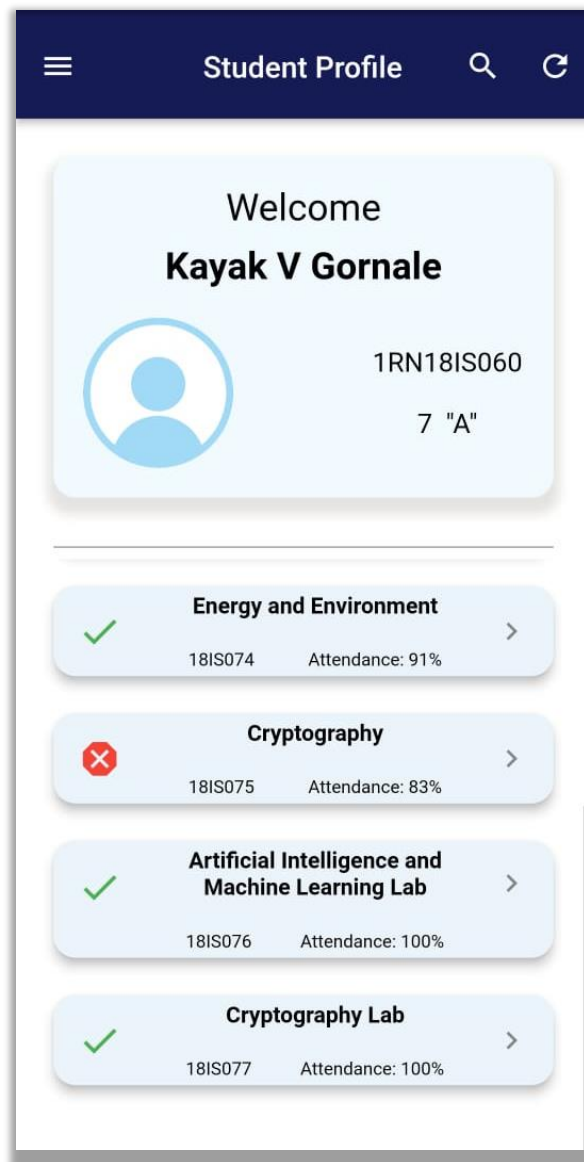


Fig. 6.4 Student Attendance details Screen 2

6.5 Subject wise attendance screen

Fig 6.5 gives the attendance details of the student in a particular subject. The number of days the student was present, the number of days the student was absent and the attendance percentage.

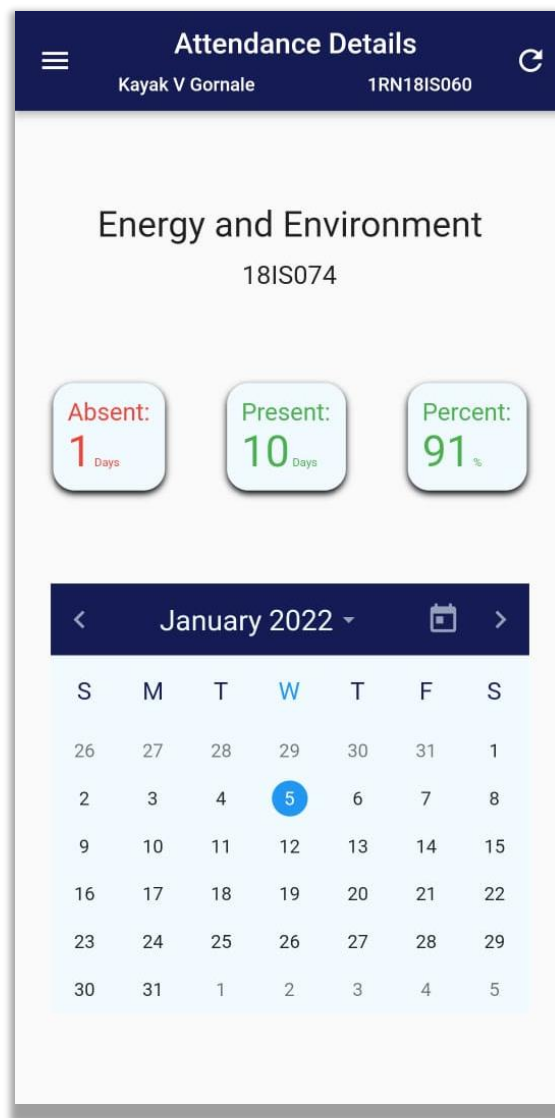


Fig. 6.5 Subject wise attendance screen

7. Conclusion and Future work

7.1 Conclusion

- This application is beneficial for teachers as well as students.
- Reduces human effort that is required.
- Since this a basic application it has a greater scope to add more features as required.

7.2 Future work

- A separate set of screens can be developed for teachers so that they can easily keep track of the attendance of all the students of a particular class.
- A detailed record of the days when the student was absent can be included.
- An option for the teachers to generate report can be included.

References

- <https://flutter.dev/>
- <https://developers.google.com/learn/pathways/intro-to-flutter>
- Beginning Flutter: A Hands On Guide to App Development by Marco L Napoli
- <https://stackoverflow.com/>
- <https://www.geeksforgeeks.org/>