Ultimate Alarm Clock- Flutter App

Personal Information

Name: Akshat Kedia

Email: akshatkedia9@gmail.com

Phone: +919771006409

GitHub Handle: https://www.github.com/Axhaat

LinkedIn: https://www.linkedin.com/in/akshat-kedia-8787531b1

Skype name: live:.cid.c471d3d7fb9daa8f

Slack username: Akshat Kedia

Proposal: https://github.com/Axhaat/Gsoc-proposal-Alarm-App

Address: Tiruchirappalli, Tamil Nadu, India

Time Zone: IST (GMT +5:30)

Project: Ultimate Alarm Clock | CCExtractor

Education

University: National Institute of Technology, Tiruchirappalli, India

Degree: B-Tech in Instrumentation and Control Engineering

Expected Graduation Date: 30 June 2025

Synopsis

Chosen Project: Ultimate Alarm Clock | CCExtractor
Project size: 350 hours
Mentor: Nishant Singhal

Abstract:

The Ultimate Alarm App is a Flutter-based mobile application designed to provide users with **a free, ad-free, and open-source** solution to their alarm needs. Unlike many other alarm apps on the market, which require expensive subscriptions and are packed with annoying ads, the Ultimate Alarm App offers a robust, reliable, and customizable alarm system that is entirely free to use. This app is perfect for people who are looking for a reliable alarm app that is easy to use and customize to their needs. With its user-friendly interface, the Ultimate Alarm App allows users to set alarms quickly and easily, with the ability to customize alarm sounds, snooze settings, and repeat schedules. Apart from all this, it has some additional features that set it apart from the other app (these features are mentioned under the feature section below). Moreover, users can also choose from a variety of alarm sounds, including their favourite music.

One of the key features of the Ultimate Alarm App is its open-source nature. This means that the code used to build the app is available to anyone who wants to see it, modify it, or use it for their own projects. Open-source software is generally considered more transparent, reliable, and secure than proprietary software, making the Ultimate Alarm App an excellent choice for users who value these attributes. Another advantage of the Ultimate Alarm App is its simplicity. The app is designed to be lightweight and efficient.

Overall, the Ultimate Alarm App is an excellent choice for anyone looking for a reliable, customizable, and open-source alarm app that won't break the bank. Whether you're a heavy sleeper who needs multiple alarms or someone who simply wants a better way to wake up in the morning, the Ultimate Alarm App has you covered.

Vision (Why Ultimate Alarm App):

The Ultimate Alarm App is a powerful, open-source application designed to provide a seamless and intuitive user experience. Unlike other alarm apps that require expensive subscriptions and are cluttered with ads, the Ultimate Alarm App is completely free and open source, allowing users to customize and personalize their experience without any limitations. The app is built on the Flutter platform, providing a fast and responsive user interface that is easy to use and navigate. Apart from some basic features like setting multiple alarms, snoozing function, choosing from a range of custom tones and themes, and adjusting volume, it has some additional feature like alarm conditional to location or alarms conditional to phone activity or alarm conditional to weather. The app is designed to be highly accurate, ensuring that users never miss an important appointment or event under specified conditions.

The open-source nature of the Ultimate Alarm App means that developers can contribute to its development, adding new features and functionality over time. This helps to ensure that the app remains up-to-date and relevant, providing users with a superior experience that is always improving. In summary, the Ultimate Alarm App is a free, open-source, and ad-free application that provides a reliable and customizable alarm experience.

Motivation:

As an enthusiastic developer with a strong passion for open-source projects, I have consistently pushed myself to go beyond my comfort zone and explore new challenges. With over a year of experience in App development and contributions to multiple open-source projects, I am confident in my ability to achieve the proposed goals of this project. In addition to completing various Android development courses, I prioritize staying up to date with the latest best practices in the industry. My skills and experience make me an ideal candidate for this project. I am a committed team player who can learn quickly and adapt to any situation. I have been actively working on this project for the past 3 weeks, and I plan to continue doing so even after the GSoC period. I have already begun working on the project design and have even created a prototype for it. I believe that my passion, skills, and dedication make me a strong candidate for this project, and I am excited about the opportunity to contribute to its success.

Detailed Proposal Description

Architecture

Here is a proposed architecture that would be best for the Ultimate alarm app: -

1. Front-end (Flutter): Flutter is a cross-platform framework for building high-performance, high-fidelity, apps for iOS and Android, and it provides several built-in widgets that can be used to implement standard features.

2. Alarm Manager Plugin: Use the flutter_alarm_manager package to implement standard features like setting alarms, snoozing, and stopping alarms. This package provides a way to schedule alarms to run at specific times, even when the app is not running.

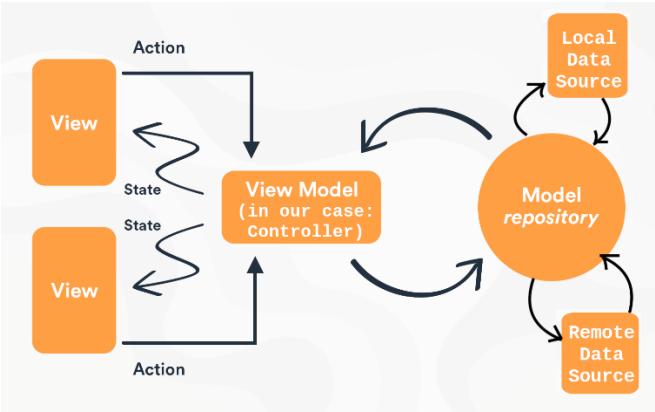
3. Cloud-Based Backend: To store user preferences and settings, we will use a cloud-based backend such as Firebase or AWS, which allows us to store user data securely and sync it across devices.

4. Geolocation Plugin: Use the geolocator package to retrieve the user's location and use it to trigger an alarm when the user enters or exits a specific location.

5. Weather API: Use a weather API like OpenWeatherMap to retrieve weather data based on the user's location and use it to trigger an alarm based on specific weather conditions.

6. Shared Preferences: Use the shared_preferences package to store user preferences and settings securely on the user's device.

7. Push Notifications: Use the firebase_messaging package or flutter_local_notification package to send push notifications to the user's device even when the app is not running.



Architecture

Since we will be building the app from the scratch, here is a **list of features** we will be working on:
Standard features: -

1. **Add** an alarm with fields Alarm name, Alarm remark, Date and time, and additional settings.
2. **Delete** an alarm.
3. Insistent **notification**
    a. Device will keep vibrating and sounding off.
    b. Upon clicking on the notification, the user is directed to the page where the alarm was set.
4. **Edit** an alarm.
5. **Choice of sound** when the alarm rings.

Non-standard features: -

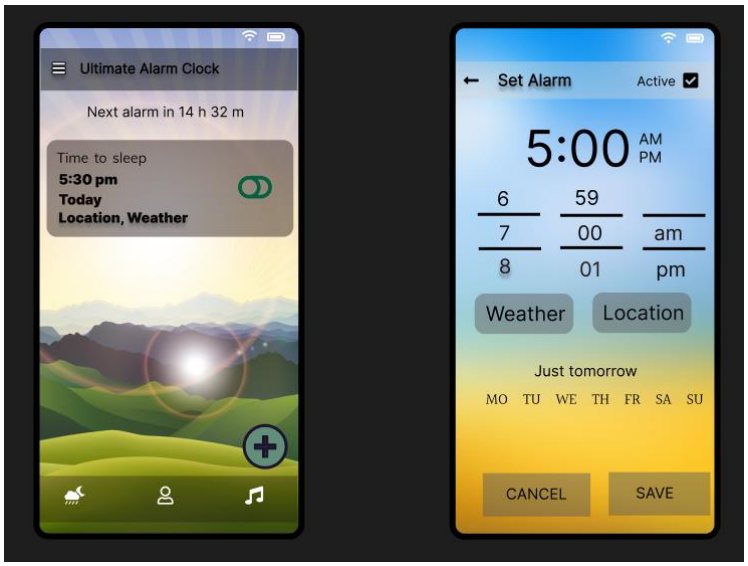
1. Alarm conditional to **location**.
For example: Ring only at 4:25 pm if and only if I'm at work, because I need to leave. Don't ring otherwise. Ring only at 7:30 am if and only if I'm at home, because I need to wake up.

2. Alarm conditional to **phone activity**.
For example: Ring only if I haven't used the phone in the past 10 minutes. Otherwise, don't because I'm already awake.

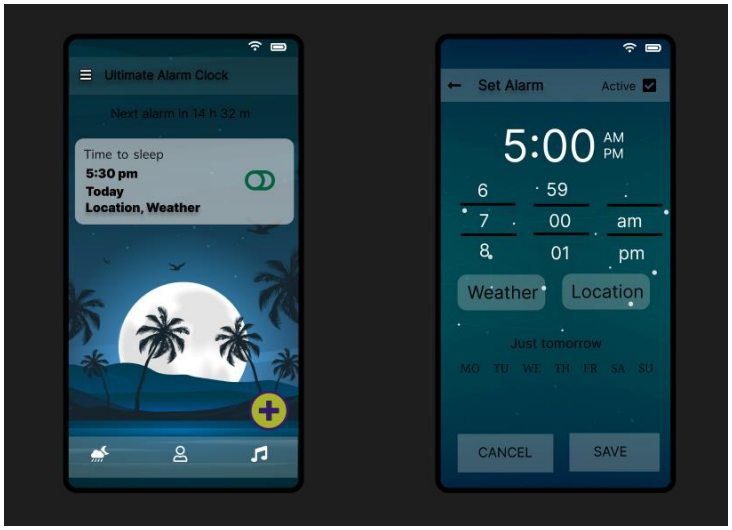
3. Alarms conditional to **weather**.
For example: Ring only if it's sunny, otherwise I'm not going anywhere :-)

4. **Shared alarms** so other people can adjust them.
For example: If I have to pick someone up let that person remove the alarm if they no longer need picking up.

Here is the Tentative **UI** for the App- Design (I have created a tentative UI for the app, final one would be made under the guidance of the mentor)



Day mode



Night mode

Here is the **goal** of the project:

- **UI for the App** to set, delete, edit an alarm with some other features.
- We will discuss Firebase in-build **plugins/packages/APIs** like
    - flutter_alarm_manager_plus plugging and AVAudioPlayer class
    - geolocator package
    - shared_preferences package.
    - firebase_messaging package
    - OpenWeatherMap API

- Adding **non-standard features** as mentioned above with the help of the flutter packages.
- **Notification** to the user to display the description and provide the option to snooze or disable based on all the conditions set (user can set an alarm).
- Once a feature is completed, I would be adding the **unit tests** for the implemented feature.
- Setup **automatic CI/CD** on GitHub
- Setup **issues and PR template** on GitHub

Timeline/Implementation

App implementation is divided into the following milestone:
(All dates mentioned below are of 2023)

❖ Community Bonding period (May 4 to May 28)

➔ Communicate and bond with fellow students and mentors. Also, I would try to learn about the ongoing projects in the organisation especially those related to flutter.

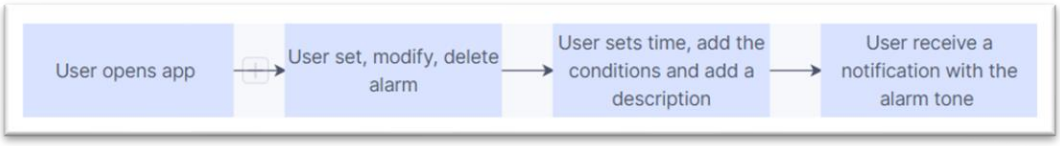
➔ Create specific issues for the project after discussing the **app flow/architecture** with the mentor.

➔ Getting familiar with the app architecture, APIs, and packages.

➔ I would explore more about the tools that can be used to do the project and discuss the final choice of tools/libraries with the mentos.

➔ Since this app would be published on the Google play store, I would start preparing the assets like icons , etc.

➔ Finalise deadline and milestone with the mentor and modify if any need arises.



Deliverable:

➔ Community bonding report and report about the experience, and app architecture

❖ Milestone 1 (May 29 to June 12)- Discussion on the design and workflow of the app and setup of basic UI

- ➔ Coding officially begins.
- ➔ Set up the flutter project and repo on GitHub.
- ➔ Create the app's final design/UI or modify the design/UI that has already been designed based on the feedback from the mentor.
- ➔ We will use widgets like Text, Container, Column, Row, RaisedButton, etc. to build your UI.
- ➔ After getting confirmation on the design, I will go on to the coding part of it.

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

class AlarmApp extends StatefulWidget {
  @override
  _AlarmAppState createState() => _AlarmAppState();
}
class _AlarmAppState extends State<AlarmApp> {
  TimeOfDay _selectedTime;

  @override
  void initState() {
    super.initState();
    _selectedTime = TimeOfDay.now();
  }
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Alarm App'),
      ),
      body: Container(
        padding: EdgeInsets.all(16.0),
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: <Widget>[
            Text(
              'Set Alarm',
              style: TextStyle(
                fontSize: 24.0,
                fontWeight: FontWeight.bold,
              ),
            ),
            SizedBox(height: 20.0),
            RaisedButton(
              child: Text(
                'Select Time',
                style: TextStyle(fontSize: 18.0),
              ),
              onPressed: () {
                _selectTime(context);
              },
            ),
            SizedBox(height: 20.0),
            Text(
              'Selected Time: ${_selectedTime.format(context)}',
              style: TextStyle(fontSize: 18.0),
            ), ],),), );}

  Future<void> _selectTime(BuildContext context) async {
    final TimeOfDay picked = await showTimePicker(
      context: context,
      initialTime: _selectedTime,
    );
    if (picked != null && picked != _selectedTime) {
      setState(() {
        _selectedTime = picked;
      }); }}}
```

This code creates a simple UI with a button to select the time for the alarm and displays the selected time once it has been set. When the user taps the button, a time picker dialog is shown using the showTimePicker function provided by Flutter. Once the user selects a time, the selected time is stored in the _selectedTime variable and displayed in the UI.

❖ Milestone 2 (June 13 to June 27)- Add standard functionality to the app.

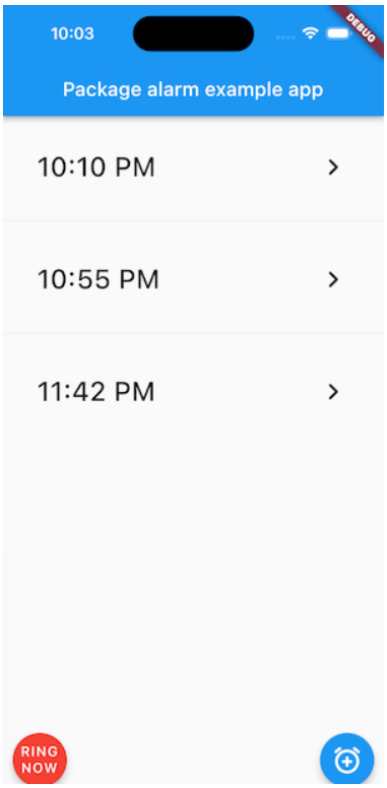
➔ We will be using **flutter_alarm_manager_plus** or **flutter_alarm_clock** for setting up the basic functionality of the app.

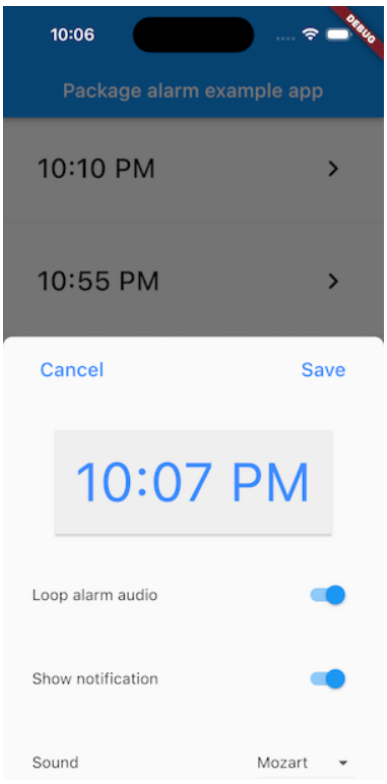
➔ **flutter_alarm_manager_plus** package is used to build an app that allows users to **set alarms, customize the alarm sound, and choose the days of the week** when the alarm should be active.

➔ Also, I would learn more about this plugin and research its querying and scaling ability.

➔ **Steps** for the setup

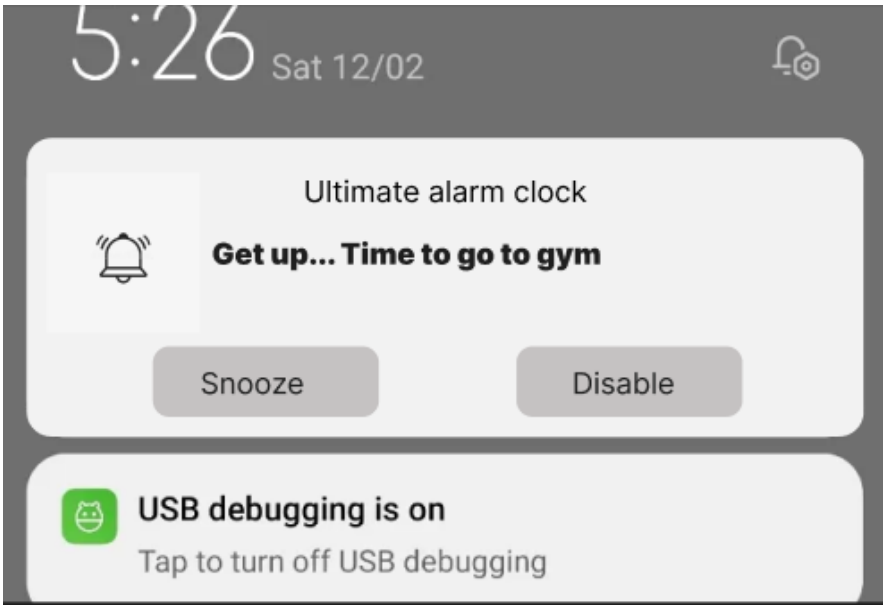
i) Add the necessary dependencies to your pubspec.yaml
ii) Import the relevant packages.
iii) Initialize the flutterAlarmManager
iv) Define a function to set an alarm.
v) Define a function to handle the alarm when it triggers.





❖ Milestone 3 (June 28 to July 12)- Working on the push notification feature.

- ➔ We will be using either of **flutter_local_notification** or **Firebase_messaging** for the push notification feature.
- ➔ Notification would offer two functionalities either to snooze or to cancel.
- ➔ For a moment we would continue with the flutter_local_notification package since user will just be dealing with the local and would not involve remote server.
- ➔ To add buttons to a notification, you can use the addAction method of the **AndroidNotificationDetails** class. We will be adding a **snooze and disable buttons** to the floating notification for ease to the user.



Deliverable:

- ➔ Documentation of all above modules

❖ Milestone 4 (July 13 to July 27)- Working on Firebase server setup and feature - Alarm conditional to phone activity.

- ➔ Improvements based on the feedback received from mentors, other community members.
- ➔ Writing unit test for above implemented modules.
- ➔ Setup the firebase project and setup Firebase cloud messaging and firebase firestore and learn about the firestore datebase. Firebase firestore would be used for storing the date.
- ➔ Working on the alarm conditional to phone activity. (i.e., Alarm would ring only if there has been no activity for last few minutes/10 minutes).

❖ Milestone 5 (July 28 to August 18)-Working on more non-standard features like alarm conditional to location and weather.

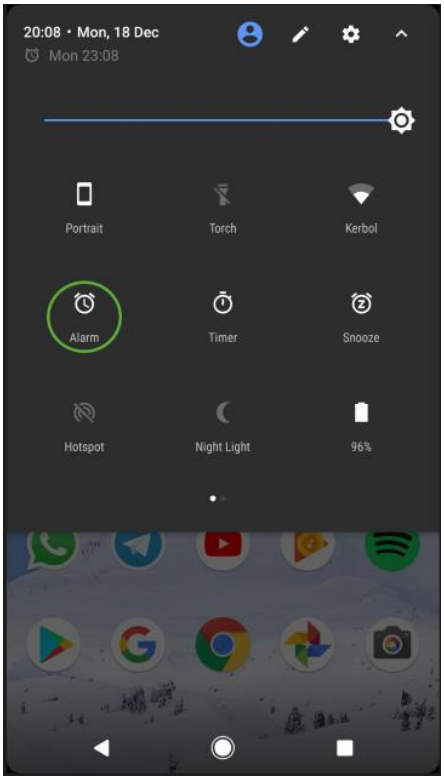
- ➔ First, we will need to integrate a location plugin into your Flutter app. There are several location plugins available, such as the geolocator plugin, which allows us to access the user's current location.
- ➔ Once we have access to the user's location, we can then use a weather API to get the current weather condition at that location. There are several weather APIs available, such as OpenWeatherMap or AccuWeather.

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

final apiKey = 'your_api_key_here';
final location = 'New York'; // or location coordinates

final url =
    'https://api.openweathermap.org/data/2.5/weather?q=$location&appid=$apiKey';

Future<Map<String, dynamic>> getWeatherData() async {
  final response = await http.get(Uri.parse(url));

  if (response.statusCode == 200) {
    return jsonDecode(response.body);
  } else {
    throw Exception('Failed to load weather data');
  }
}

// Call this function to get the weather data and display it in your app
void displayWeatherData() async {
  final weatherData = await getWeatherData();

  final temperature = weatherData['main']['temp'];
  final description = weatherData['weather'][0]['description'];

  // Display the extracted information in your app
}
```

- ➔ We will need to sign up for an account with the weather API provider and obtain an API key. Once we have the API key, we can make API requests to get the weather data for the user's location. Once we have the user's location and the current weather condition, we can then check if the location matches the specified location and if the weather condition matches the desired weather condition. If the location and weather condition match, we can then trigger the alarm to ring.
- ➔ We can use Flutter's built-in alarm and notification APIs to trigger the alarm.
- ➔ Finally, we will need to handle cases where the app is not running in the foreground, or the device is in sleep mode. We will use Flutter's background execution APIs to ensure that the app can still monitor the user's location and trigger the alarm even when the app is not active.

❖ Milestone 6 (August 18 to September 1)- Adding sharing feature and Final testing on devices and publishing on play store.

➔ Add app to the quick tile of the phone for easy access. Users can enable and disable app for quick settings. This blog can be referred for the same.



Share alarm feature:

➔ First, we need to integrate a sharing functionality in our Flutter app. We can use the share package available in Flutter to implement this functionality. Once we have the sharing functionality, we need to create a database or use an existing one to store the shared alarms. We will use Firebase Realtime Database or Firestore for this purpose. Then, we need to create a mechanism to generate a unique ID for each alarm that is shared. This ID will be used to access and modify the shared alarm. We will add a button or option in your alarm app to share an alarm. When the user clicks on this button, we generate a unique ID for the alarm and share it with the user's contacts or via social media platforms. Once the alarm is shared, we need to allow others to modify the alarm by providing them access to the unique ID. We will create a separate page in the app where users can enter the unique ID to access and

modify the alarm. We can also add a feature to allow the person who created the alarm to revoke access to the alarm in case the person no longer needs to pick someone up.

Testing and deploying

- ➔ Final testing on devices and deploying in play store.
- ➔ By this time mentor would have approved the design and logo. Before uploading on the play store, I will finalize the icon, banner, and screenshot, and description needed while uploading on play store.
- ➔ Firebase Analytics would also be setup for app. It provides many analytics tools built-in like crashlytics, logging and many more that would help us identify where users are struggling and try to improve in the following weeks.

❖ Optional Milestone:

- ➔ **Automatic CI/CD for repo for releasing builds**.
  If I get free time between, I work on setting up CI/CD for the repo by GitHub Actions for automating many chore tasks like releasing builds, singing up Apks, etc.
  There are various tools listed on flutter deployment doc like Codemagic and Fastlane that could be used for setting up the workflow. Appropriate tools and actions would be implemented after a discussion with my mentor.
- ➔ **PR and issue template for bugs, features or feedback**.
  User can report bugs, request, and give feedback by ready-made template on GitHub. These Template auto-populates the issues/pull-request description field and provide a skeleton framework that contributors can fill out.
  They help provide a baseline standard of information quality and organisational rigour.

- ❖ **I plan to continue contributing to this project after the GSoC'23**

My background/ Technical skills

I am Akshat Kedia, 2nd year Undergraduate pursuing BTech degree at National Institute of Technology, Tiruchirappalli, India. I started with QBasic and then learned C++, Java, and Python. For last one year I have been learning app development (both Java and Dart). I created several apps as a part of competitions and project, including a music player named kMusic. I use git and GitHub every day, and I am well acquainted with using them for version control. Moreover, I am a competitive programmer too. Of late, I have been learning cyber security, Kubernetes, and many more skills to boost my performance.

I am also very active in open source, college technical events and hackathons. I have participated in Hacktoberfest'23, DWOC (Delta Winter of Code), and TRI-NIT Hackathon.

I am a member of TeCOS, the official open-source community of our college, where we participate and at the same the time encourage college students to participate. Being in this community, I got the work on a project called PaiNITTe, an initiative to help student get access to academic resources. My recent focus is on the TeCOS initiative, encouraging the open-source culture in our college. I am very active in open source. I try to participate in many of the open-source events happening. I participated in KWOC, DWOC, and Hacktoberfest previously and this is my attempt in GSOC.

Projects

Contribution Projects-

1. PaiNITTe

PaiNITTe is an open-source website that provides academic resources for the students of NIT Trichy. The website is designed to offer a platform for students to access various academic-related activities such as past question papers and class tests. One of the key features of PaiNITTe is its database of CTs (Class Tests) and end-semester question papers. This database is constantly updated with the latest question papers, allowing students to access them easily and use them for exam preparation. Overall, PaiNITTe serves as an excellent resource for NIT Trichy students to access academic materials and prepare for their exams. Its open-source nature and collaborative approach make it an ideal platform for knowledge sharing and learning. I have added contributions to it and added a few features to it. (click to see the contribution)
PaiNITTe is made using Docusauras 2.

GitHub link: https://github.com/Axhaat/painitte

Personal Projects:

1. kMusic

kMusic is an open-source music player designed for music lovers and enthusiasts. It supports a wide range of audio formats, allowing users to play their favourite music files without any hassle thus making it a great choice for anyone looking for a reliable and versatile music player. I developed this as one of my personal projects.
KMusic is made on Java.

GitHub link: https://github.com/Axhaat/kMusic

2. Jarvis AI

Jarvis AI Project is a Python-based project that aims to provide personalized assistance to users based on their queries. The project uses libraries of python to understand user queries and generate appropriate responses. The functionality of the project includes features such as setting reminders, playing music, searching the internet, controlling smart home devices, and answering general knowledge questions.
Jarvis AI is programmed in Python.

GitHub link: https://github.com/Axhaat/Jarvis-AI

3. To-do App

The To-do app is an open-source project designed for personal use. It is a software application designed to help individuals organize their tasks and responsibilities. The app typically offers a simple and intuitive user interface that allows users to create to-do lists, add tasks to these lists, and provides a check box and option to delete once the task is completed. I am also trying to add few more feature like users can also prioritize tasks based on their importance and track their progress as they complete each item on their list and things like ability to categorize tasks, collaborate with others on shared lists, and integrate with other productivity tools.
It is written in Kotlin.

GitHub link: https://github.com/Axhaat/todo-app

Contributions to open source

Beacon - https://github.com/CCExtractor/beacon/pull/202
Taskwarrior-flutter - https://github.com/CCExtractor/taskwarrior-flutter/issues/142
openMF/mobile-wallet - https://github.com/openMF/mobile-wallet/issues/1368
openMF/mifos-mobile-cn - https://github.com/openMF/mifos-mobile-cn/pull/235
Hacktoberfest -Successfully completed Hacktoberfest 2022.
Processing-android - https://github.com/processing/processing-android/pull/730
CodingPractice-Hacktoberfest - https://github.com/csubhasundar/CodingPractice-Hacktoberfest22/pulls?q=is%3Apr+author%3A%40me+is%3Aclosed

Commitment:

Weekly Commitment: 40-45 hours

During the first two months of my college semester break, I plan to work full-time on this project. However, August will be a busy month for me as I will be returning to campus and starting my college internship season. Therefore, I have given more weeks to complete milestone 4 to make up for any potential time constraints. Although I do not foresee any other gaps or absences, I will inform my mentor in advance in case of any unforeseeable emergencies.

Anyways, I am more than ready **to work well past my committed time** if needed.

Other Commitments:

Here are some of the things I would do that may help me prevent personal matters from affecting my GSoC performance:
1.Make sure to allocate enough time for my GSoC project each day or week, depending on my project requirements and timelines.
2.Keeping in touch with my mentor regularly and informing them of any personal matters that may affect my project timeline.