○

# Ultimate Alarm Clock II - Flutter App

## About Me:

|  |  |
|---|---|
| **Name:** | Deepanshu Singh |
| **University:** | ITM University, Gwalior, India |
| **Github:** | Deepanshuigtm (Deepanshu Singh) · GitHub |
| **LinkedIn:** | https://www.linkedin.com/in/deepanshu-singh-6a57b1235/ |
| **Resume:** | https://drive.google.com/file/d/1IWSMyu32LJUFvA5__nAdFmPO8WQZyo-9/view |
| **Location:** | Gwalior, Madhya Pradesh, India |
| **Email:** | 2207deepanshu@gmail.com |
| **Phone no:** | +91 8103702636 |
| **Timezone:** | IST(GMT +5:30) |
| **Project:** | Ultimate Alarm Clock II \| CCExtractor |
| **Proposal:** | https://github.com/Deepanshuigtm/ccextracter-gsoc-proposal |

## Abstract:

This project aims to build an Alarm Clock app that develops an innovative alarm clock with intelligent functionalities. This advanced clock is designed to intuitively dismiss alarms based on user activity on their phone, **real-time weather conditions**, and additional customizable parameters. It introduces various challenges to ensure users are fully awake, offering a unique approach to morning routines. Moreover, it incorporates a feature **allowing users to set shared alarms**, promoting collaboration and coordination among multiple individuals. This project **aims to redefine the traditional alarm clock experience, enhancing wake-up routines with modern technology and personalized settings**

## Goals:

Here are some goals for the project:

- Enable users to **create profiles** with different alarm settings for various scenarios, including **workdays, weekends, and vacations**

    ○

- Users can **add alarms to specific profiles**, such as assigning an alarm to the working days profile if it's suitable. Additionally, **they can remove alarms from a profile if needed**, providing flexibility in managing alarm configurations according to their preferences .
- I**f there are no violations of YouTube's Terms of Service**, then I want to create a new feature where **users can search for their favorite music** using the youtube_explode Flutter package. They can then set the searched music as their alarm ringtone
- **Dashboard feature** will provide users with a comprehensive **overview of their alarm usage statistics on a weekly or monthly basis**, allowing them to track their adherence to set alarm plans and analyze their snooze habits
- The **Smart Traffic feature** utilizes **real-time traffic data** to estimate the **time required for the user to reach their destination**. It factors in current traffic conditions, distance to the destination, and the user's preferred mode of transportation (e.g., driving, walking, public transit). The **alarm is then set to go off before the scheduled time**, taking into account the estimated travel time, so that the user can leave with enough buffer to arrive at their destination punctually.
- App will introduce a **guided tour that familiarizes users with the app's basic functionalities**. It will walk them through various screens and components, highlighting where to find existing alarms, **how to create new alarms, utilize the timer feature, stopwatches, and more.**
- **Implement Firebase Cloud Messaging for notifications**: **alarms with 10-minute reminders** and redirect to alarm upon click, weekly and monthly **report notifications**, and **weekly leaderboard showcasing top performers among friends**, elevating user engagement and performance monitoring.
- Setup **automatic CI/CD** on Github
- Setup **issue and PR template** on Github

## Implementation:

App implementation is divided into the following **Milestones**

**(all dated mentioned below are of 2024)**

- **Community bonding period (April 19 to May 20)**
    - Interact and engage in the community, get to know my mentors and fellow students. Also, I would try to learn about other ongoing projects in the organisation, especially those related to flutter.
    - I will be discussing the **app flow** in detail with mentors. I would go with UI that is intuitive and easy to use and app flow that is efficient as much as possible.
    - Since this app would be published on the **Google Playstore**, I would start preparing assets like icons.

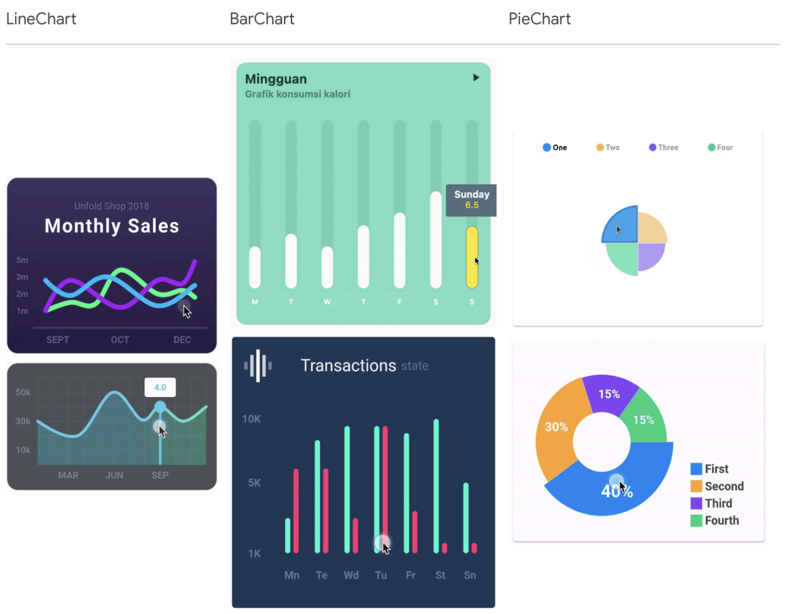
    -

- **MILESTONE 1 (June 13 to July 26)- Create profile setup**

    - Users can create a new profile by completing a form, providing details such as the **profile title (e.g., "Weekend Alarm")** etc . This information will be stored in the database.
    - Then Users can **add alarms to these profiles by accessing the alarm section**, where all alarms are listed on the home page. From there, they can select an alarm and add it to one of their profiles
    - To facilitate editing of profiles, distinct methods will be employed. Adding or editing profiles will involve **creating a new collection in Firestore specifically for profiles**. Each profile will contain alarm IDs associated with it
    - **Schema of User Profile**
        - The user profile schema will include a **unique automatically generated ID** from the database, ensuring each profile is uniquely identified within the database.
        - Schema will include a **title field of type string**, allowing users to input a descriptive title when creating a profile.
        - **Description field, which will be optional.** Users can choose to provide additional details about their profile if they wish.
        - The **isShareable** field will be of type boolean, allowing users to specify whether they **want the profile to be shared or not**.
        - Within each profile, there will be another **collection named "alarmsProfile,"** which will **contain all the alarm IDs that the user has added to that specific profile.**

- ○

- **MILESTONE 2 (June 27 to July 12)- Dashboard feature**



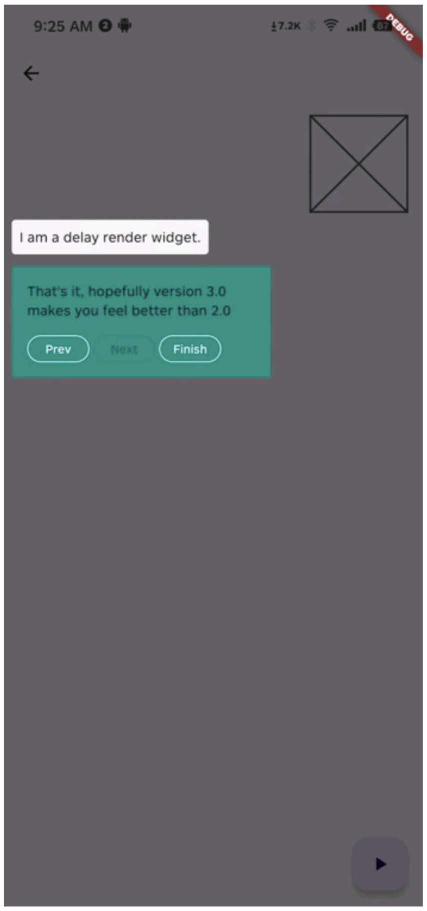
FL Chart is a highly customizable Flutter chart library that supports Line Chart, Bar Chart, Pie Chart, Scatter Chart, and Radar Chart

- ○ **Design the UI layout for the dashboard,** incorporating graphs to visualize day-by-day user progress with respect to alarm usage, and **overall user progress**. Users can discern on which days they are performing well or poorly
- ○ Package to create charts https://pub.dev/packages/fl_chart/install
- Usage shown on GitHub - https://github.com/imaNNeo/fl_chart/blob/main/repo_files/documentations/line_chart.md
- ○ Integrate **data retrieval and processing functionality** to fetch user alarm usage like User actions related to the alarm include **snooze time**, the **time taken to finish challenges**, and the **time taken to stop or complete the challenge**. We will calculate and graph the duration of these actions statistics from the database.
- ○ **Database Schema**: Design a database schema to store user alarm usage data. This schema should include tables to store information about alarms, challenges, and user actions related to alarms (such as snooze time, completion time, etc.)
- ○ **Data Retrieval**: queries to retrieve user actions related to alarms from the database. This may involve querying tables such as alarm logs, challenge logs, and user logs.
- ○ **Data Processing**: Once the data retrieve from the database, process it to calculate the duration of each action. For example, you can calculate the snooze time by subtracting the time when the alarm was snoozed from the time when it was stopped or completed

- ○

  - **Statistics Calculation**: Calculate statistics such as average snooze time, average completion time for challenges, etc., based on the processed data.
  - **Graph Generation**: Use a library like fl_chart package that mentioned above to create graphs showing the distribution of snooze time, completion time, etc. You can create histograms, line plots, or any other type of visualization that best represents the data.
  - **Testing**: Test the functionality thoroughly to ensure that data is retrieved accurately from the database, processed correctly, and that the graphs are generated as expected.

- **MILESTONE 3 (July 12 to July 26)- App Tour Guide**


I am a delay render widget.

That's it, hopefully version 3.0 makes you feel better than 2.0

  - **App Tour Guide Implementation**
  - Introduce a guided tour that familiarizes users with the app's basic functionalities. It will walk them through various screens and components, **highlighting where to find existing alarms, how to create new alarms, utilize the timer feature, stopwatches, and more.**

-

- Package to install : Add the **flutter_intro: ^3.1.2** package link https://pub.dev/packages/flutter_intro
- **Initialize Tour Functionality**: function to **handle the guided tour functionality**. This function will be triggered when the user requests to start the tour, either through a button or as part of the onboarding process.
- **Define Tour Steps**: Define the steps of the tour, **specifying the screens or components to highlight** and the corresponding messages or instructions to display. Each step will consist of a target widget and associated content.
- **Implement Tour UI**: Design and implement the UI elements for the guided tour, including tooltips, **overlays, or pop-ups that indicate the highlighted areas** and display the tour messages.
- **Trigger Tour**: Determine the trigger points for starting the tour. **This will be at first login, or via a dedicated "Take Tour"** button in the app's settings or onboarding screens.
- **Handle User Interactions**: Implement logic to **handle user interactions during the tour, such as swiping to navigate between tour steps, skipping steps**, or ending the tour prematurely.
- **Persist Tour State**: Optionally, consider persisting the tour state to ensure that users do not see the tour repeatedly once they have completed it. You can use shared preferences or a similar mechanism for storing and retrieving this state.
- **Test and Iterate**: Test the guided tour feature thoroughly on different devices and screen sizes to ensure a consistent and user-friendly experience. Gather feedback from users and iterate on the design and content as needed to improve clarity and effectiveness.

-

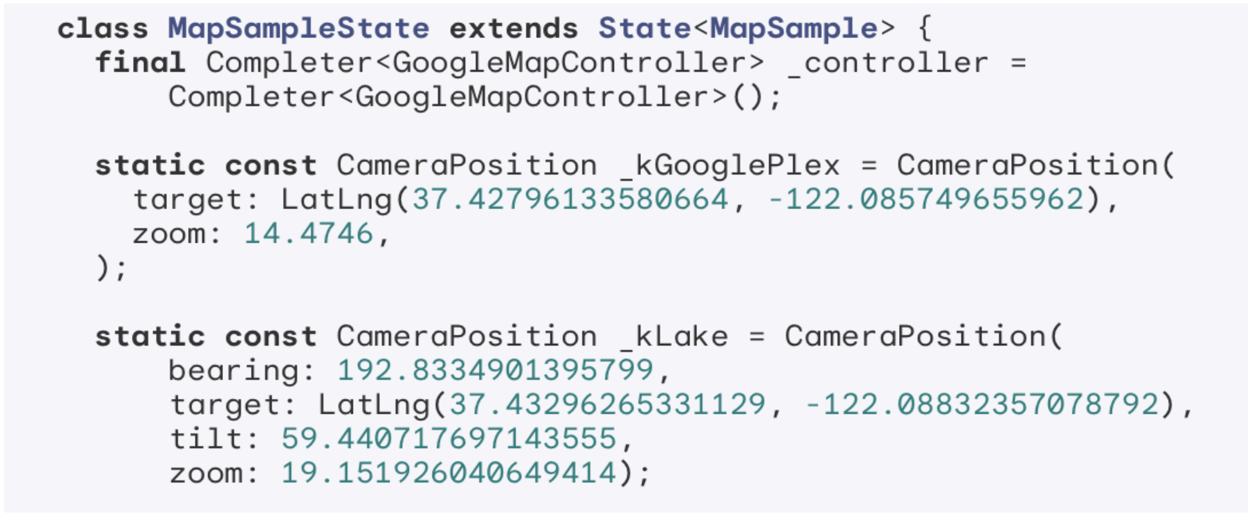
- **MILESTONE 4 ( July 26 to August 21 ) - Firebase Cloud Messaging for notifications**

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

import 'package:firebase_core/firebase_core.dart';
import 'package:firebase_messaging/firebase_messaging.dart';
import 'package:flutter/foundation.dart';
import 'package:flutter/material.dart';
import 'package:flutter_local_notifications/flutter_local_notifications.dart';
import 'package:http/http.dart' as http;

import 'firebase_options.dart';
import 'message.dart';
import 'message_list.dart';
import 'permissions.dart';
import 'token_monitor.dart';
```

  - **The notification would be delivered by Firebase cloud messaging.**
  - **Dependency required :  firebase_messaging: ^14.7.19**
  - **Implement alarm notifications**: Create a service to send alarm notifications with 10-minute reminders. On click, redirect users to the alarm screen.
  - **Create weekly and monthly report notifications**: Develop services to send notifications for weekly and monthly reports summarizing user performance. Allow users to customize notification settings for these reports.
  - **Implement Dashboard notifications**: Set up a service to send notifications to Users summarizing their recent alarm usage patterns, statistics, and trends. This feature enhances user engagement and encourages users to stay informed about their alarm habits, helping them maintain a healthy and productive routine
  - 
  - **Create notification settings**: Develop a user interface for notification settings where users can toggle on/off for each notification type (alarms, weekly reports, monthly reports).
  - **Store user notification preferences**: Save user notification preferences in the database to apply them when sending notifications.
  - **Handle notification preferences**: Modify notification sending logic to respect user preferences. Only send notifications for enabled types.
  - **Test notification functionality**: Thoroughly test notification sending and handling, ensuring that users receive the correct notifications based on their preferences.
  - **Optimize notification delivery**: Continuously monitor and optimize notification delivery to ensure timely and reliable notifications.
  - Discuss the required changes for the completed milestones with the mentors.
  - All the features would have been tested before, but now the app would be tested as whole in the **production environment**. Ensuring that none of code is breaking and will be solving any issue raised by mentor.

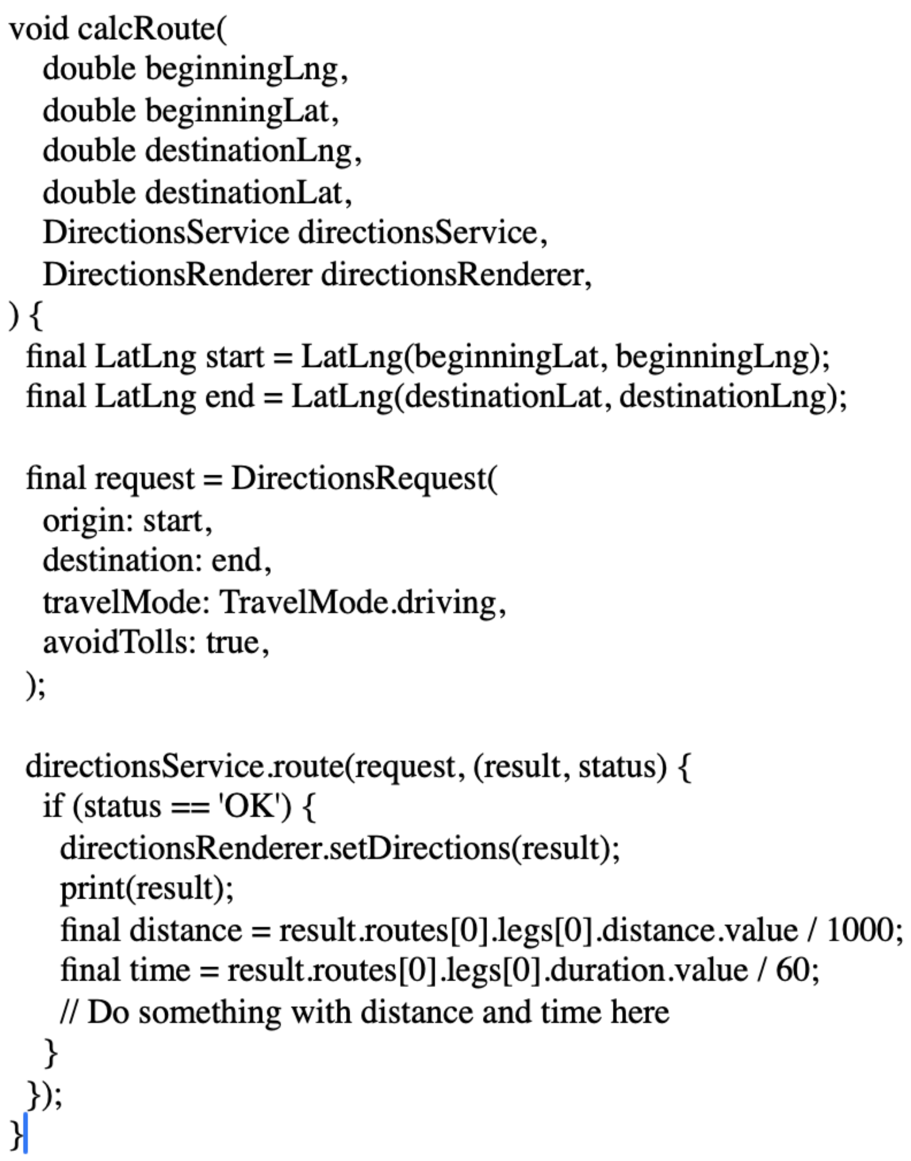
○

- **MILESTONE 5 (August 22 to September 10) - Feature “Smart Traffic feature”**
- **Required Dependencies:** To fetch real-time traffic data and calculate travel time. You can integrate this using the google_maps_flutter package: Google Maps Flutter.
    - **Get API Access :** Obtain access to a reliable traffic data API provider. Google Maps API offers comprehensive traffic data that I can integrate into the app.
    - **Implement Location Services :** Integrate location services into the app to fetch the user's current location and their destination. The **user should be able to input their destination either manually or by selecting it from a map.**
    - **Fetch Traffic Data :** traffic data API to fetch **real-time traffic conditions between the user's current location and the destination.** This data will help estimate travel time accurately.
    - **Calculate Travel Time :** Once the traffic data and the user's preferred mode of transportation, **calculate the estimated travel time considering factors like distance, traffic congestion, and average speed.**
    - **Set Alarm Accordingly :** Adjust the alarm time based on the estimated travel time. **Subtract the estimated travel time from the desired arrival time** to determine the optimal departure time. Set the alarm to go off before this departure time, allowing the user enough time to get ready and travel to their destination.
    - **User Interface :** Design a user-friendly interface where users can set alarms. Include an **option to enable the Smart Traffic feature**. Provide options for them to input their destination and select their preferred mode of transportation.
    - **Traffic-time calculation code for reference :**



```
class MapSampleState extends State<MapSample> {
  final Completer<GoogleMapController> _controller =
      Completer<GoogleMapController>();

  static const CameraPosition _kGooglePlex = CameraPosition(
    target: LatLng(37.42796133580664, -122.085749655962),
    zoom: 14.4746,
  );

  static const CameraPosition _kLake = CameraPosition(
      bearing: 192.8334901395799,
      target: LatLng(37.43296265331129, -122.08832357078792),
      tilt: 59.440717697143555,
      zoom: 19.151926040649414);
```

◦



```
void calcRoute(
  double beginningLng,
  double beginningLat,
  double destinationLng,
  double destinationLat,
  DirectionsService directionsService,
  DirectionsRenderer directionsRenderer,
) {
 final LatLng start = LatLng(beginningLat, beginningLng);
 final LatLng end = LatLng(destinationLat, destinationLng);

 final request = DirectionsRequest(
  origin: start,
  destination: end,
  travelMode: TravelMode.driving,
  avoidTolls: true,
 );

 directionsService.route(request, (result, status) {
  if (status == 'OK') {
   directionsRenderer.setDirections(result);
   print(result);
   final distance = result.routes[0].legs[0].distance.value / 1000;
   final time = result.routes[0].legs[0].duration.value / 60;
   // Do something with distance and time here
  }
 });
}
```

- **Optional MILESTONE 6 (September 11 to September 25) - YouTube Music Alarm Integration**

```
// ignore_for_file: avoid_print
import 'package:youtube_explode_dart/youtube_explode_dart.dart';

Future<void> main() async {
  final yt = YoutubeExplode();
  final streamInfo = await yt.videos.streamsClient.getManifest('fRh_vgS2dFE');

  print(streamInfo);

  // Close the YoutubeExplode's http client.
  yt.close();
}
```

    -

  - **Require Package : youtube_explode_dart: ^2.1.0**
  - Integration of **youtube_explode** Flutter package: Incorporate the youtube_explode package into the project to enable seamless interaction with YouTube's API for music search functionality.
  - **Implement YouTube music search**: Develop a search feature within the app, utilizing youtube_explode to **fetch titles, artists, and video IDs of searched music**.
  - Design user-friendly search interface: Create an intuitive and visually appealing user interface (UI) for the music search functionality, ensuring ease of use for users.
  - **Enable music selection for alarms**: Allow users to select their preferred music from the search results and set it as their alarm ringtone within the app.
  - **Enhance alarm customization**: Provide users with the option to personalize their wake-up experience by waking up to their favorite tunes sourced directly from YouTube.
  - **Ensure reliability and efficiency**: Test the integration thoroughly to ensure smooth functionality and optimize performance for a seamless user experience.

- **MILESTONE 7 (September 25 to October 15) - Final testing on devices and publishing on Play store**

- By this time mentor would approved the design of app. Before uploading on play store, i will finish the final icon, banner, screenshot, and description needed while uploading on play store.
- **The app review takes time, so by that time i will start writing the blog post.**
- App is almost ready, the **app would be listed on play store** for “**early access**”.
- **Firebase Analytics would also be setup for app. It provides many analytics tool built-in like crashlytics, logging, etc. I would see where user are struggling and try to improve in following weeks.**

- **Optional MILESTONE**
  - **Automatic CI/CD for repo for releasing builds**

    If i get free time between i will also work on setting up ci/cd for the repo by **Github Actions** for automating many chore tasks like releasing builds, siging 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 discussion with mentor

  - **PR and issue template for bugs, features or feedback.**

    User can **report bugs, request and give feedback** by ready-made templates on Github. These templates auto-populate the issue/pull-request description field and

◦

provide a skeleton framework that contributors can fill out. They help provide a baseline standard of informational quality and organisational rigour.

## My background / Technical skills

I am **Deepanshu Singh**, **3rd year computer science student at** [ITM University, Gwalior](), India. I've been enjoying web development for the first time ever since I built my **first Python project,** "**Automation**". I have been deeply immersed in web development. Initially, I **began by exploring Flutter** and Python, with my early Flutter projects involving the creation of basic Android **applications in Android Studio**, such as a tic-tac-toe game, quiz app etc.

In my 10th grade, I built many small applications in Python, such as a Stock Portfolio Tracker and a Fitness Tracker. I used to post these **projects on social media** to show my friends and family.Then I became interested in **mobile application development** and built many projects in Java. Additionally, I developed native applications using Flutter. After that, I learned Android development, web development, **Flutter, React, Docker, Django, Flask, and shell scripting**. I began participating in Capture **The Flag (CTF) contests** during one of my college events

Right now I am proficient in **app dev (both native and framework like react and flutter), and full-stack web dev (js, HTML, Django, react, flutter).** I use vscode, android studio and **sublime** as my text editors. Recently i tried learning server hosting, cloud and firebase.

I have gained experience **working with several startups as an intern**, where I contributed to various projects involving the development of mobile applications, web applications, and more. Recently, I have focused on a **Flutter project** where I delved into various state management techniques such as **Bloc and Riverpod** to streamline state management efficiently. Additionally, I have acquired expertise in cloud technologies like **Firebase and Firestore** during my recent projects.

I worked as **Python Automation** Intern in CHAKRABORTY LAKHAR INNOVATIONS PVT.LTD, used technologies like Web scraping using Python, **BeautifulSoup and Selenium** for automation . There I learned agile methodologies to design, **software testing** and debug software applications
Certificate of Internship - Link of certificate
Letter of recommendation - Link of letter of recommendation

I am **Microsoft Learn Student Ambassador**, helping students of my college.  The MSP program enhances students' employability by offering training in skills not usually taught in academia, including knowledge of Microsoft technologies.

◦

I am also a coordinator, responsible for managing, ideating and supervising most things tech in the college. I worked on various campus development initiatives.
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 and this is my attempt in GSOC.** I am also actively involved in encouraging open-source culture in my college, I listed a few initiatives like org review, human of open source, etc under TeCOS.

### AI Task Automation 2024

I have developed an **AI-based web application** that assists users in a wide range of tasks. Users can ask questions by providing context through various media formats such as video files, audio files, PDFs, or text documents. Additionally, the application offers numerous other features aimed at saving users' time and enhancing their productivity.
Containerize Next.js, Flask with Docker. Use Docker Compose for multi-service setup. Optimize images, manage networking. Simplify deployment and scaling
https://github.com/Deepanshuigtm/ml-app.git

**Personal Projects** -
https://github.com/Deepanshuigtm/internshala_job.git
https://github.com/Deepanshuigtm/flutter-chat.git
https://github.com/Deepanshuigtm/webscraping_amazon_bags.git
https://github.com/Deepanshuigtm/guess_My_Number.git

## Contributions to open source:

### CCExtractor Ultimate Alarm Clock

Some of my contribution to **Ultimate Alarm Clock** project are listed below :

- **Fixed a bug that was reported by someone else**. The issue can be found here: https://github.com/CCExtractor/ultimate_alarm_clock/issues/170
- **Feature Swipe-to-Delete alarm** - https://github.com/CCExtractor/ultimate_alarm_clock/issues/271
- **Undo option Swipe-to-Delete alarm -** https://github.com/CCExtractor/ultimate_alarm_clock/issues/314
- **Customizable Undo Duration -** https://github.com/CCExtractor/ultimate_alarm_clock/issues/378

## Commitments:

Initial first 2 months would be lies in my college so i would working part time on this project at that time. After June, I'll begin my last year of college, the fourth year, and will dedicate full-time hours to the project. Occasionally, I'll need to attend college, but apart from those scheduled days, I'll be fully committed. In the event of any emergencies, I'll inform my mentor in advance.

**Weekly Commitment: 40-50 hours**

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