MOBILE COMPUTING PROJECT

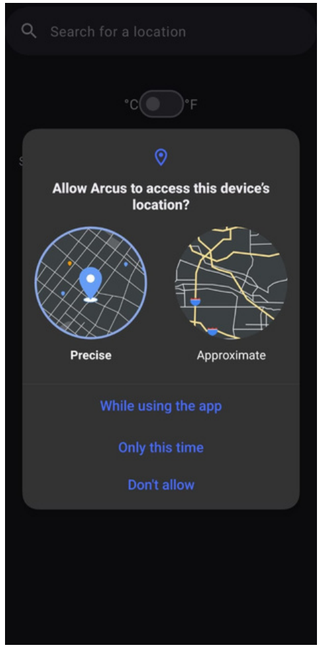
Weather App

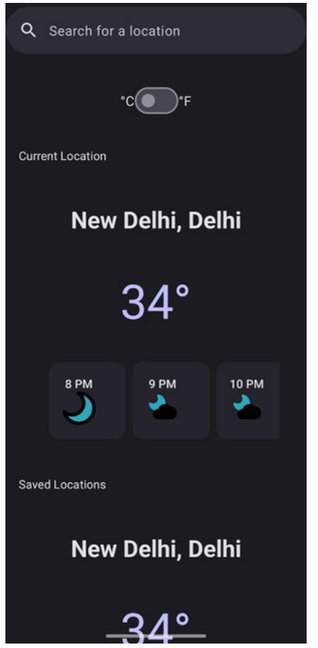
Aabhas Chaddha (2021369)
Aditi Singla (2021372)
Kanishk Kukreja (2021393)
Sameer Budhiraja(2021416)

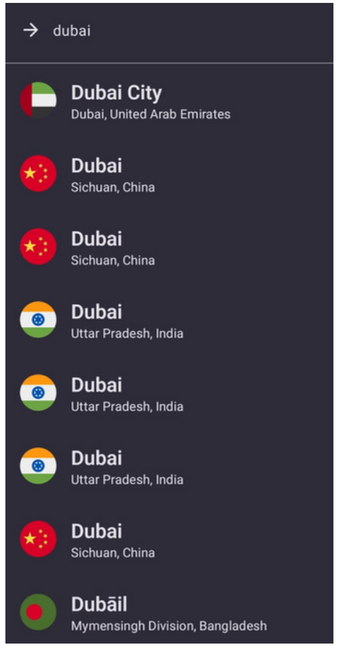
ACTIVITIES

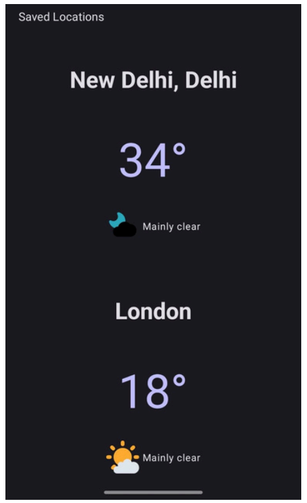
HOME SCREEN

- At the launch of the app, the app requests the user to share his/her location to get weather information. The user can choose between a precise or approximate location and the extent of the permission.
- The home screen displays the weather info of the current location/ the selected location.
- The app's home screen lets the user decide the format in which the temperature should be displayed.
- It also contains the search bar for the user to manually enter the location whose weather info needs to be fetched.
- The saved location, and their weather info is displayed as well on the home screen.



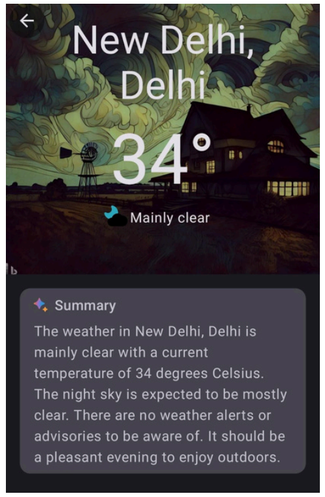


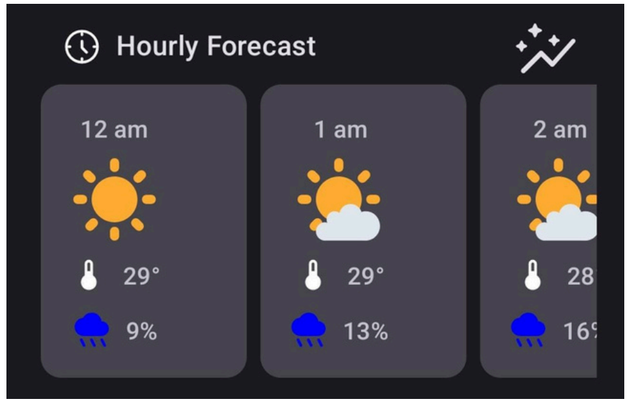


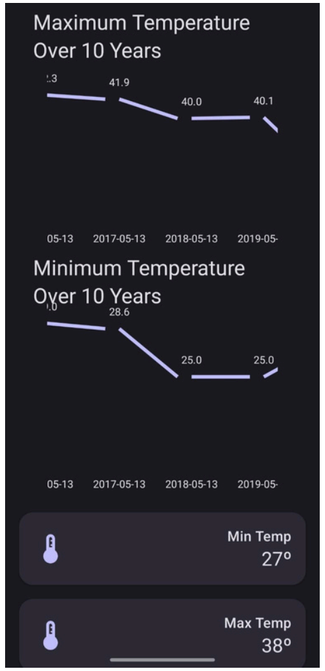


WEATHER SCREEN

- The Weather Screen contains all the detailed data of the current weather condition, hourly forecast, 7-day forecast, and comparison between current and historical weather.

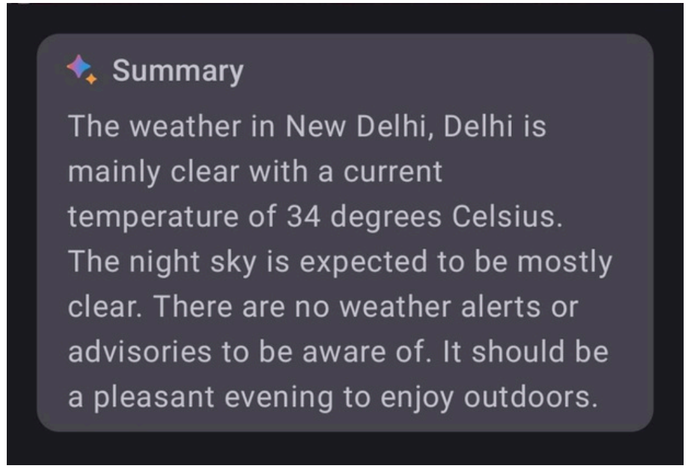






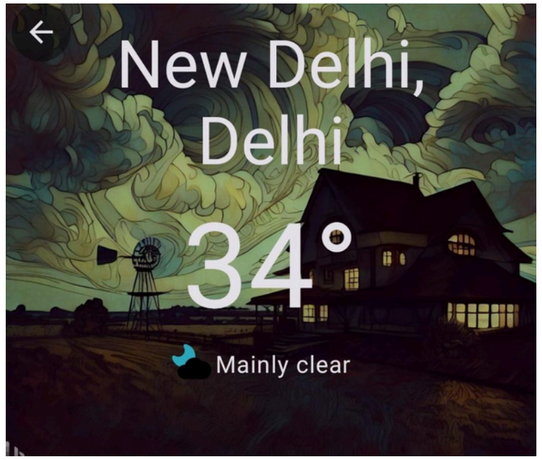
WEATHER SUMMARY

- The app employs Gemini API to produce descriptive info and analysis of the current weather conditions.



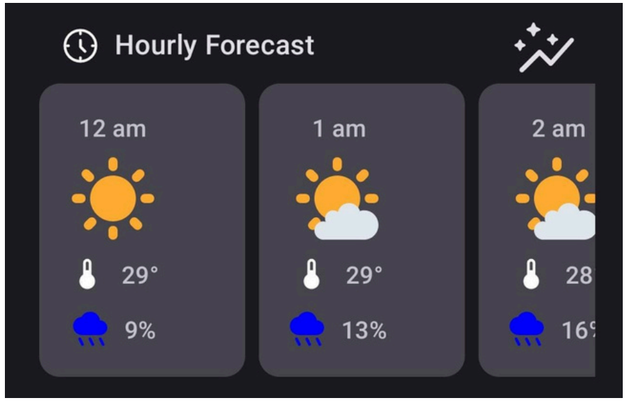
SAVING LOCATION

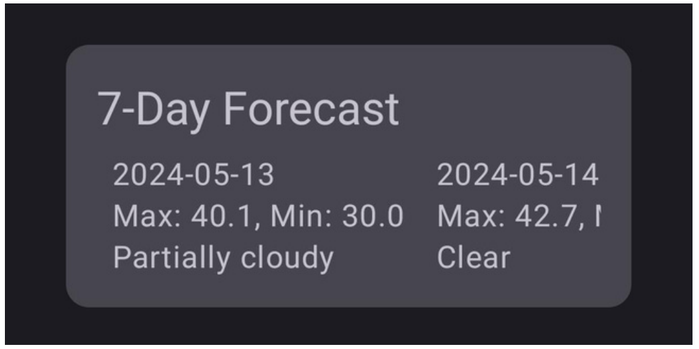
- The app also provides the option to save frequently accessed locations by using the add symbol in the top right corner of the weather screen, thereby allowing users to switch between different places without repetitive input quickly.
- This is displayed on the home screen.



WEATHER FORECAST

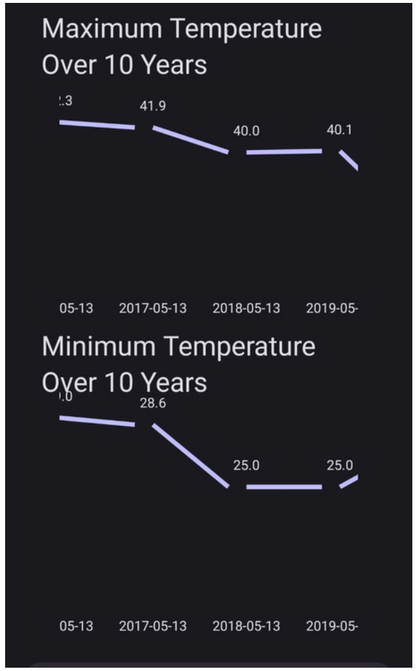
- The app provides both short-term hourly (24-hour) and long-term (7-day) weather forecasts for the user's selected location.
- Users can view graphical representations, such as charts or graphs, depicting temperature trends and precipitation chances over the forecast period.
- This feature helps users plan their activities, travel, and daily routines based on anticipated weather conditions, whether it's deciding on outdoor plans or scheduling travel arrangements.





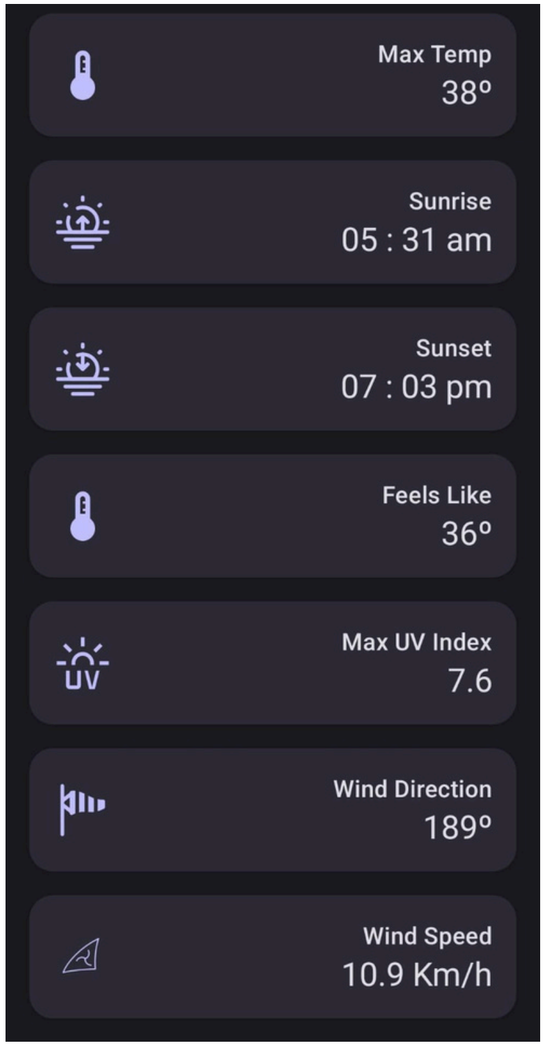
CURRENT VS HISTORICAL WEATHER

- The app utilizes historical weather data obtained from an API to compare current weather conditions with long-term averages for the same location for the same date across 10 years.



LOCATION-BASED WEATHER UPDATES

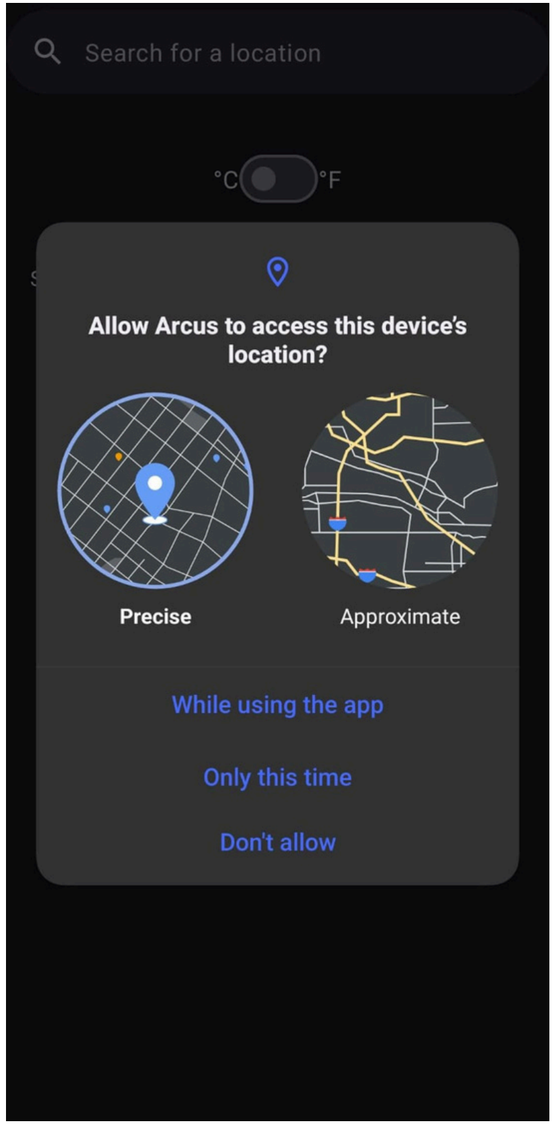
- The app uses the device's GPS service to pinpoint the user's current location and fetches real-time weather data from a reliable API.
- Displays essential weather parameters such as minimum & maximum temperature, sunrise, sunset, feels like, max UV index, wind direction, wind speed, and weather conditions (e.g., sunny, cloudy, rainy) for the user's current location.

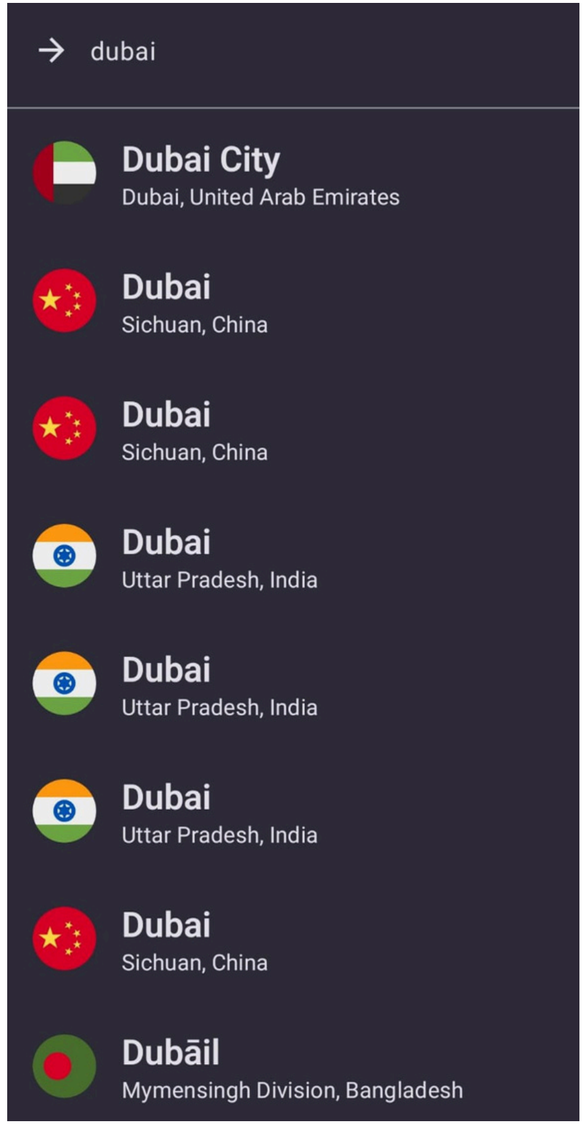


ANDROID FEATURES

LOCATION DETECTION USING SENSORS

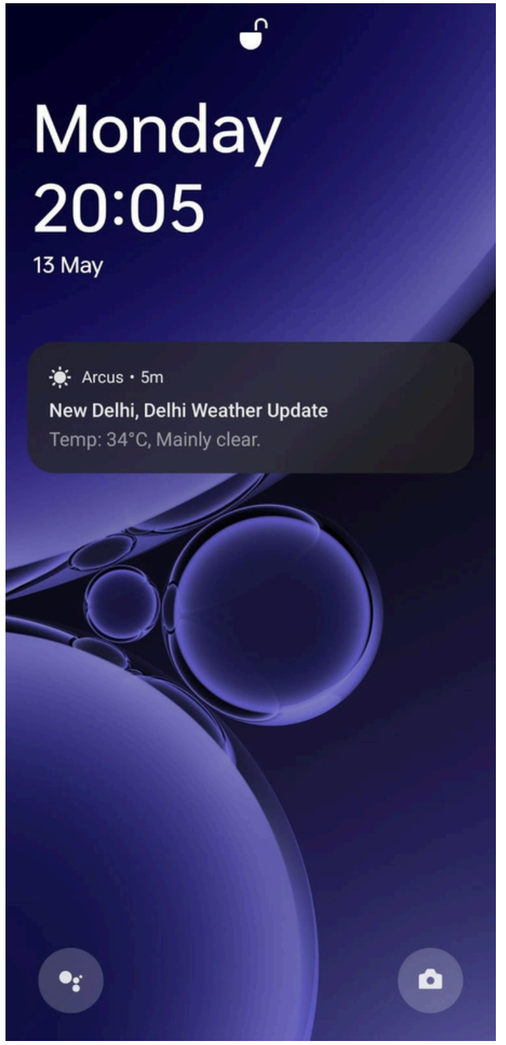
- The app offers two options for detecting the user's location: automatic GPS detection or manual input.
- Automatic detection uses the device's GPS sensors to pinpoint the user's location accurately.
- Alternatively, users can manually enter their location if they prefer or if GPS is unavailable.





NOTIFICATION

- Beyond just opening the app, the user can opt to receive timely notifications about significant weather changes.
- These notifications serve as proactive alerts, keeping users informed about any sudden shifts in weather conditions that may affect their plans or safety.



DATABASE

- The database serves as a repository for storing vast amounts of weather data sourced from external APIs. This includes current weather conditions, historical weather records, forecasts, and other meteorological information.
- The app maintains a database of frequently accessed locations, allowing users to save and revisit their favorite or commonly visited places with ease.

Additional Features

- We have used two Open Meteo and Visual Crossing for weather information.
- We have used Hilt Model to simplify the process of injecting dependencies into your app's components, such as activities, fragments, and view models.
- Implementing Moshi, made it easy for us to convert JSON data into Java or Kotlin objects

THANK YOU