MINISTRY OF COMMUNICATIONS AND INFORMATION

DIGITAL EGYPT PIONEERS INITIATIVE (DEPI).

SOFTWARE DEVELOPMENT TRACK

Mobile Application Track

PROJECT: METRO GO APPLICATION

Using Flutter

SUPERVISED BY

Eng. Hany Nemr

2024-2025

PROJECT TEAM WORK

1	Mariam Mahmoud Ahmed
2	Rawda Ahmed ibrahim
3	Radwa Hany Mohammed
4	Aml Saied Abdo

Abstract

The Metro Go App is a smart mobile application designed to improve daily transportation for Cairo's metro users. The app allows users to input their starting and destination stations to receive the shortest and all routes, number of stations, ticket price, estimated travel time, and real-time station alerts. Developed using Flutter (Dart), the application highlights how technology can transform public transportation into a more efficient and user-friendly experience.

Acknowledgment

We extend our sincere gratitude to the Ministry of Communications and Information Technology for launching the "Digital Egypt Builders Initiative," which provided us with this outstanding opportunity to develop our skills bring this project to life. We also express our deep appreciation to CLS Company for its dedication and for providing us with a highly experienced professional in mobile application development — Eng. Hany Nemr — whose guidance played a vital role in enhancing our learning journey. His broad expertise, practical insights, and valuable feedback greatly contributed to the successful execution of this project with high quality. At the conclusion of this journey, we offer our heartfelt thanks and respect to Eng. Hany Nemr for his continuous support and commitment, and we are truly honored to have learned from him during this inspiring educational experience.

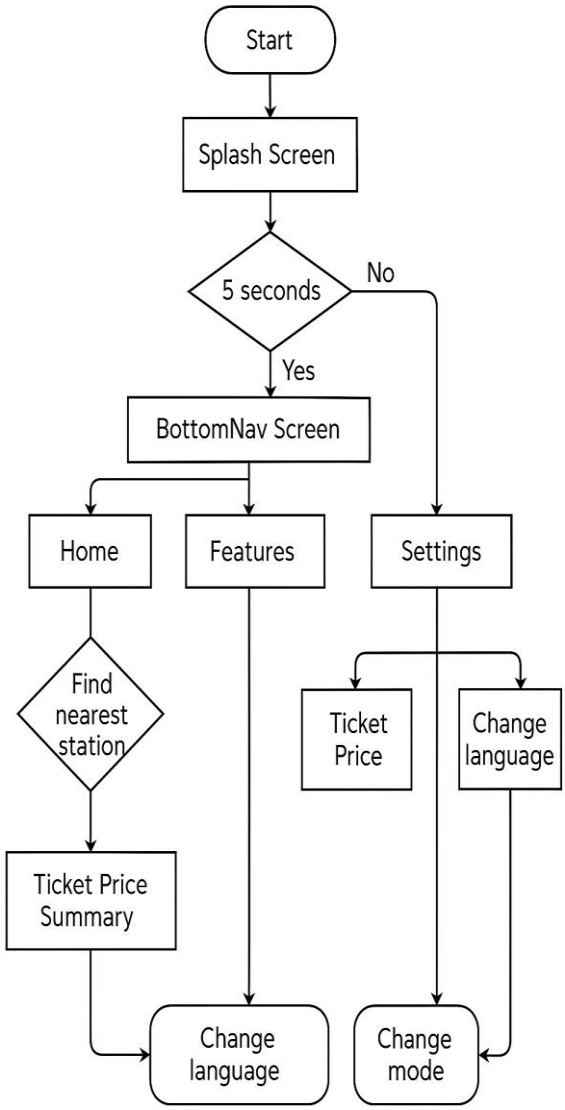
What is the Cairo Metro App?

The Cairo Metro App is an innovative mobile application developed as part of a graduation project to address a real-world challenge faced by millions of daily metro users in Cairo. The application is designed to simplify commuting within Egypt's largest and busiest metro system by offering a smart and interactive platform. Through this app, users can input their current metro station and their intended destination, and the app will instantly calculate all possible travel routes between the two points. It then selects the most optimal route, usually the shortest one in terms of the number of stations. In addition, the application displays valuable travel information such as the estimated time of arrival, total distance to be covered, ticket pricing, and even provides alerts as the user approaches each station. This tool enhances not only navigation but also planning, especially for newcomers, tourists, or anyone unfamiliar with the intricacies of Cairo's metro lines. The app serves as a daily companion to help users avoid confusion and delays, making urban transportation smoother and more accessible.

Importance of Using Technology in Cairo Metro

The use of digital technologies in public transportation is no longer a luxury—it has become a necessity in major cities around the world. Cairo, being one of the most populous cities globally, experiences intense pressure on its public transportation systems. By integrating technology into the Cairo Metro system, we can enhance efficiency, reduce congestion, and offer passengers a more seamless and reliable commuting experience. Digital applications help solve daily commuting problems such as lack of information, missed stops, inefficient route planning, and long travel times. Our mobile application directly addresses these issues by offering real-time routing, optimized path selection, and clear travel data. Furthermore, the app promotes sustainability by encouraging the use of public transportation through convenience, which may help reduce traffic congestion and pollution in the city.

Flow chart:



The link include all codes about our project:

https://github.com/mariamelgandour/MetroGo-

Technologies We Used:

FLUTTER APPLICATION:

What is Flutter?

Flutter is an open-source UI software development kit created by Google. It allows developers to build natively compiled applications for mobile, web, and desktop using a single codebase. Flutter's layered architecture helps developers control every pixel on the screen, enabling beautiful and customized designs.

What is Dart?

Dart is the programming language used by Flutter. It is optimized for UI development and offers features like asynchronous programming, strong typing, and object-oriented capabilities. Dart compiles to native code for optimal performance on mobile platforms. 3.3 Packages We Used in This Application To enhance functionality and simplify development,

we used the following packages:

cupertino_icons: ^1.0.8

get: ^4.7.2

flutter_native_splash: ^2.4.6

dartx: ^1.2.0

geolocator: ^14.0.0

video_player: ^2.9.5

curved_labeled_navigation_bar: ^2.0.6

timelines_plus: ^1.0.6

What is the Importance of Responsive Design in Flutter?

Responsive design ensures that the app adapts to various screen sizes and
orientations, providing a consistent experience across devices. It's crucial in Flutter
development to avoid layout issues and offer a user-friendly interface regardless of the
device used.

Tools of Responsive Design in Flutter

Flexible

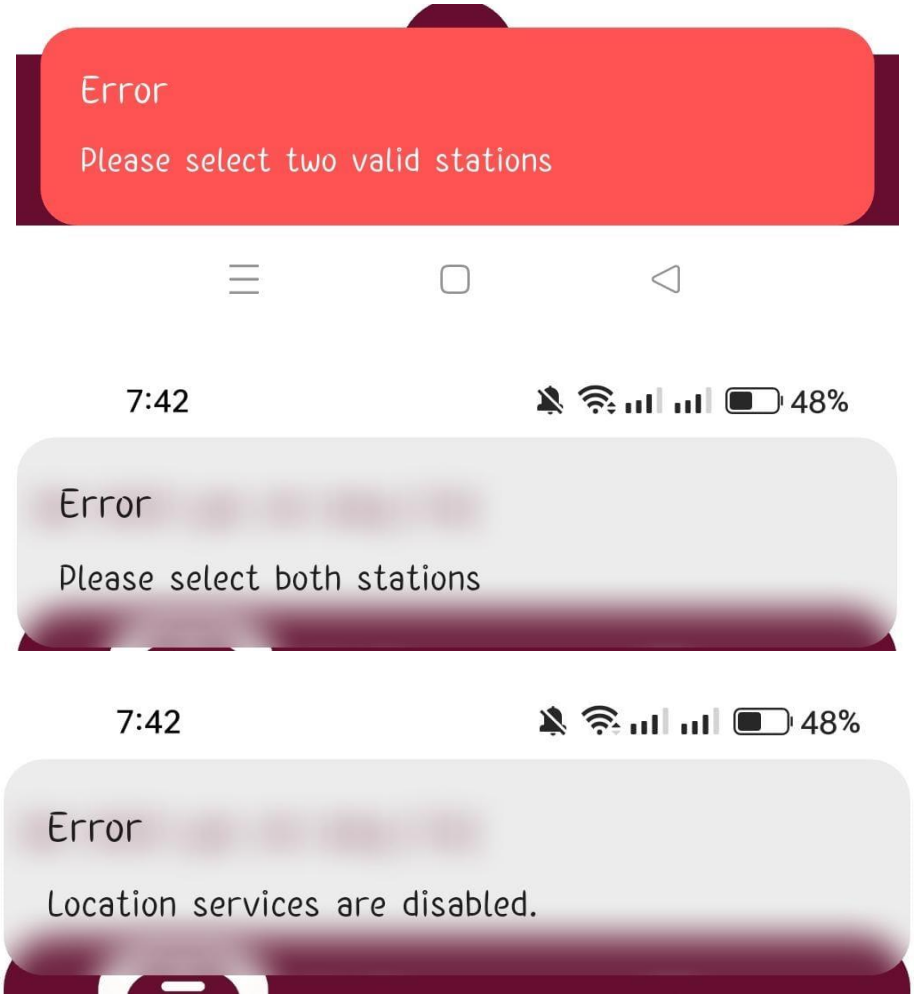
- Allows a widget to take up **only the needed space** within a Row or Column.
- Prevents overflow while enabling flexibility in layout.

Expanded

- Forces a widget to **take all remaining space** in a Row or Column.
- Useful when you want to evenly distribute space among children.

About our Application

Screens Implementation:

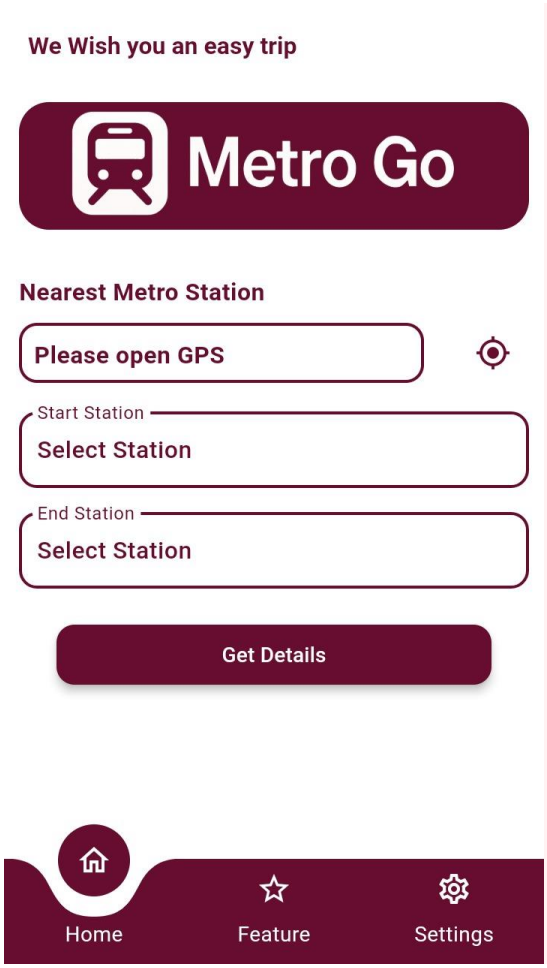


.

Splash & Home Screen :

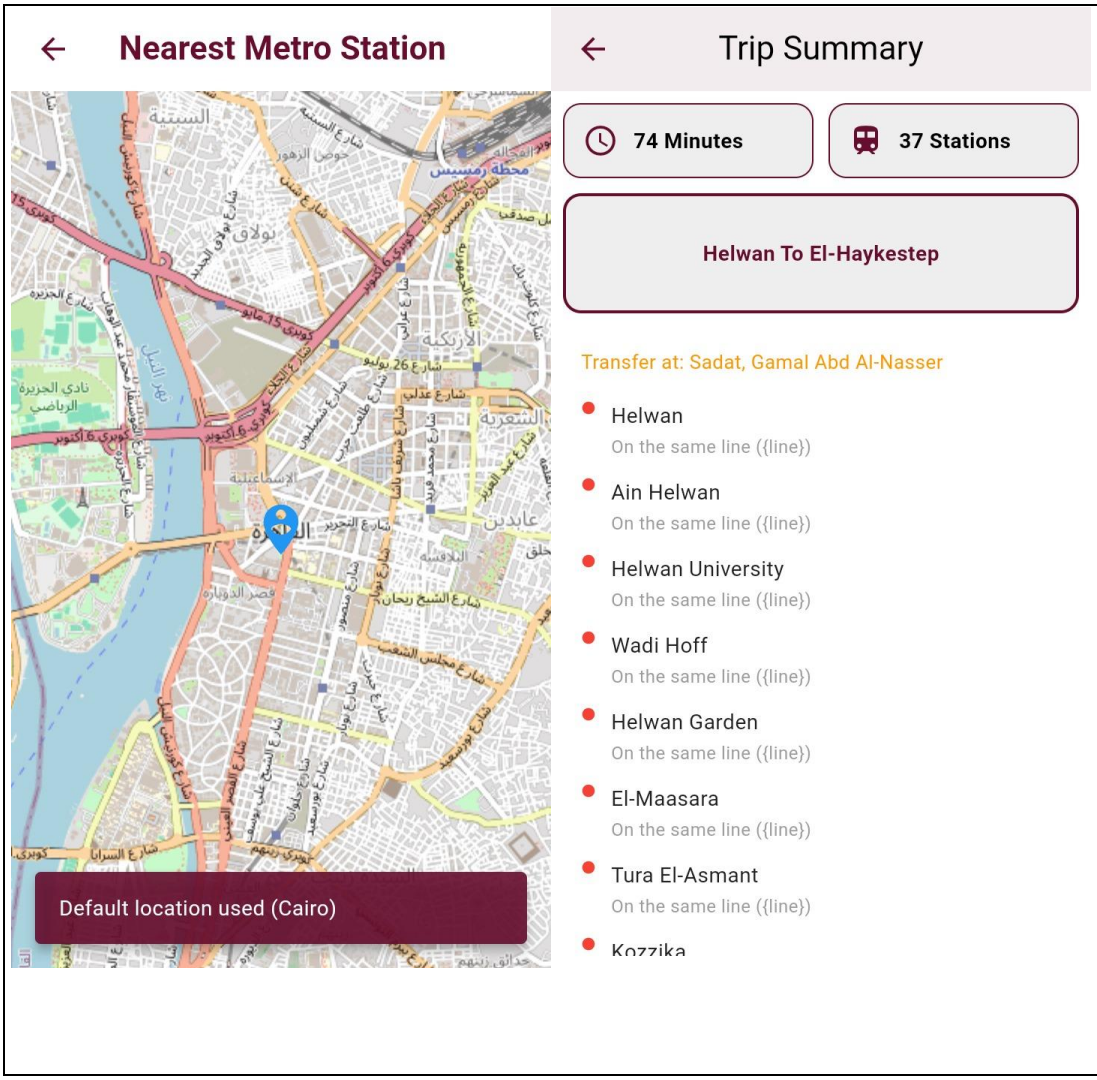
Splash Screen : Implemented to express about main goal of app and navigate to Home Screen

Home Screen : implemented to allow user know the nearest metro station from location and allow user to enter to stations and when pressed in get details take to Summary Ticket Price Which know the user no. of stations, time that will take, paths available and exchange stations

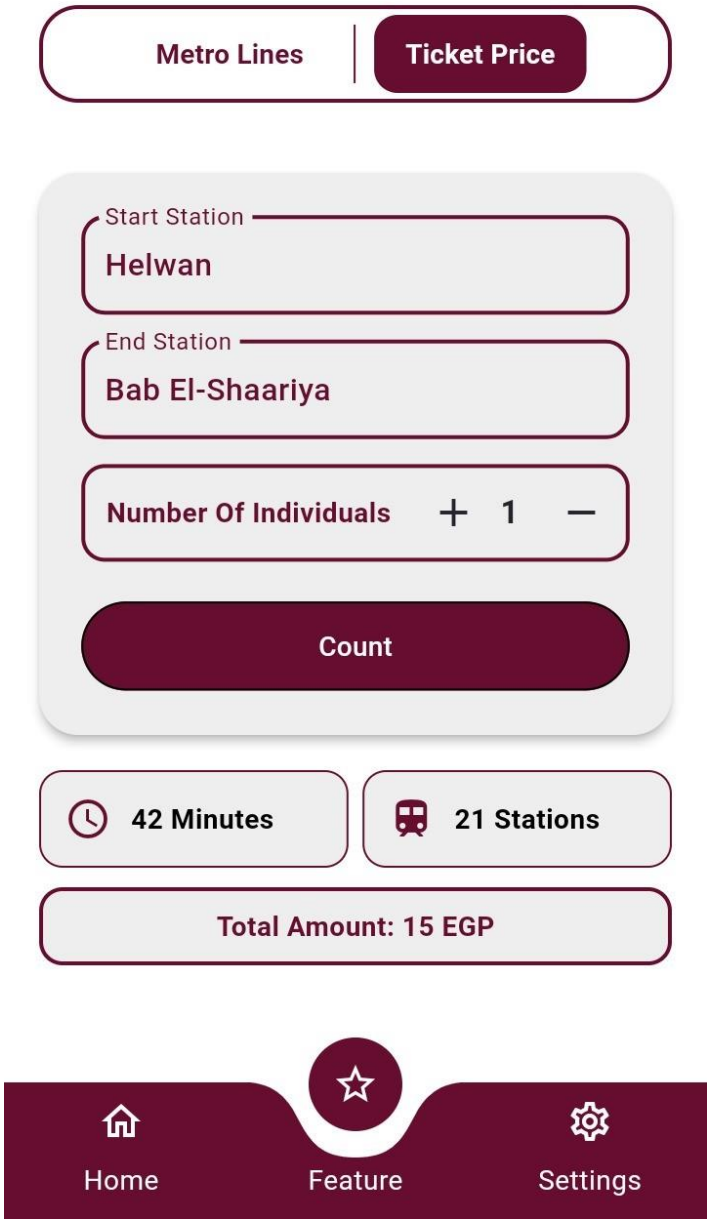



We Wish you an easy trip
Metro Go
Cut
Copy
Paste
Share
Ain
Ain Helwan
Ain Shams
Home
Feature
Settings
We Wish you an easy trip
Metro Go
Select Station
ma
Massara
Maspero
Alf Maskan
Home
Feature
Settings

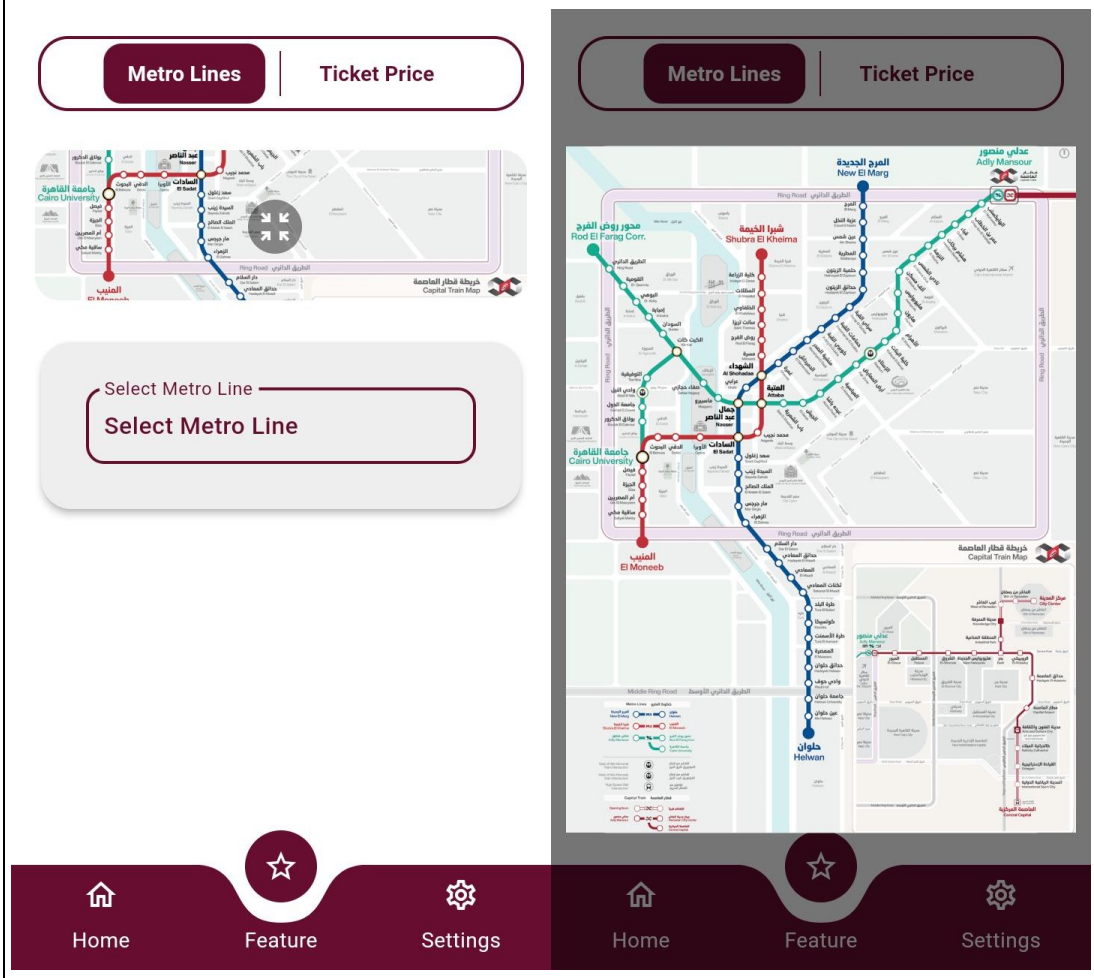
Nearest Location to Metro Station & Metro Ticket Summary :



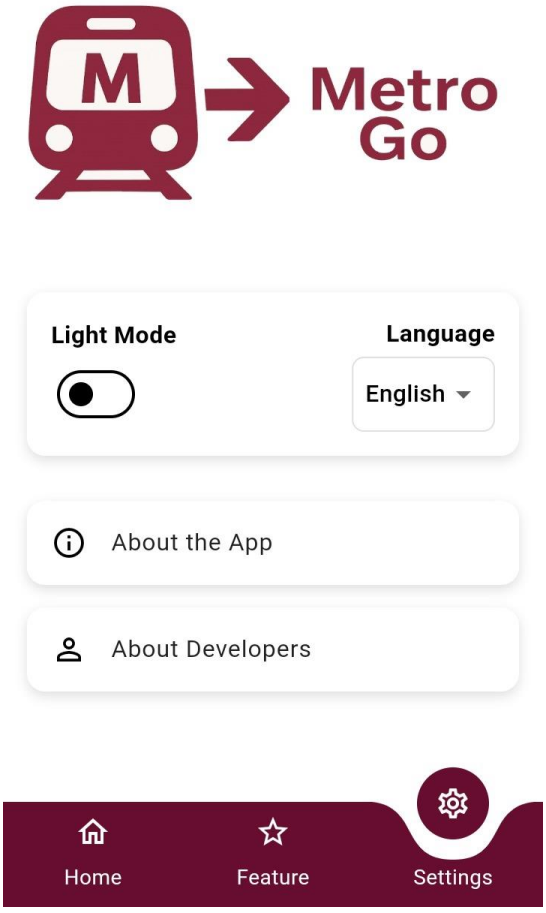
Ticket Price Feature : allow user to know the price of ticket and know it also if no.of.individuals more than Two person .



Metro Lines And map for Lines : allow user to know Station of line and show Map of metro to know more about stations and exchange Stations

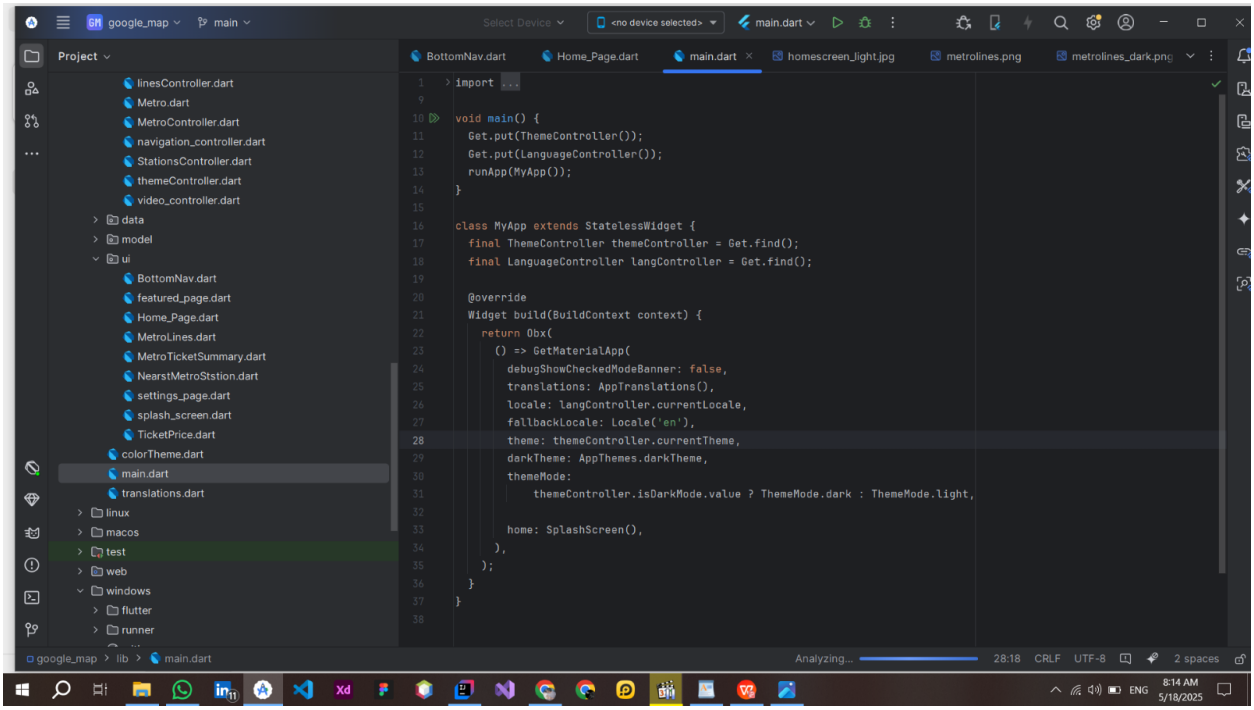


Settings Screen: allow user to exchange language of app (english ,arabic) and mode of pp (dark ,light)



Code Explanation :

main. dart :



Metro controller : The `MetroController` manages the core logic of the Cairo Metro Map app. It handles route calculation, station counting, estimated time, pricing, and transfer detection using the GetX state management package.

Functions Summary:

1. *updateValues({ newStations, newMinutes, newPrice })*

Updates the observable values for station count, estimated minutes, and ticket price.

2. *_buildGraph()*

Creates a graph-based representation of all metro lines to be used in route-finding algorithms.

3. *findPathBetweenStations(String start, String end)*

Finds a single valid path between two stations using Breadth-First Search (BFS)

4. *findAllPathsBetweenStations(String start, String end)*

Finds **all** possible paths between two stations using Depth-First Search (DFS).

5. *getShortestPath(String start, String end)*

Returns the shortest path (with the least number of stations) between two stations by analyzing all possible paths.

6. *calculateStationsBetween(String start, String end)*

Returns the number of stations between two points and identifies any transfer stations along the route.

6. *calculateStationsBetween(String start, String end)*

Returns the number of stations between two points and identifies any transfer stations along the route.

7. calculateTime(int stationCount)

Estimates travel time based on the number of stations. Assumes 2 minutes per station.

8. calculatePrice(int stationCount)

Returns the ticket price based on station count:

- 1–9 stations → 8 EGP
- 10–16 stations → 10 EGP
- 17–23 stations → 15 EGP
- More than 23 → 20 EGP

9. calculateTotalPrice(int stationCount, int individuals)

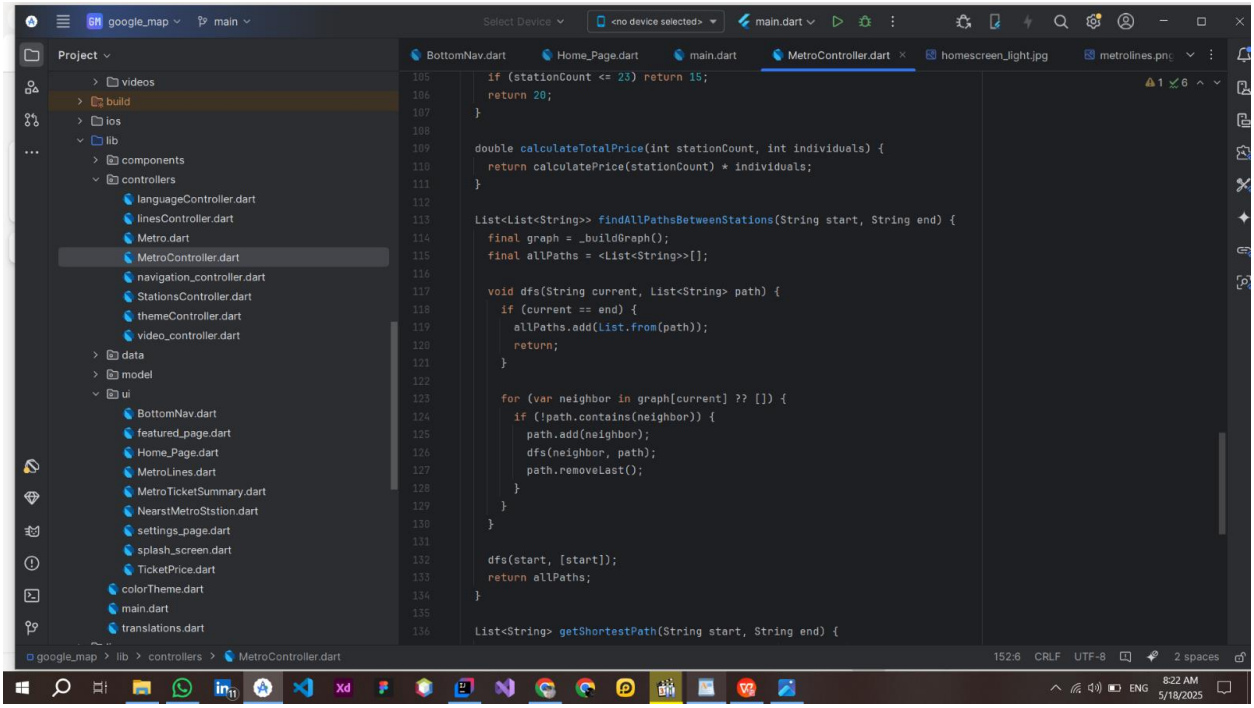
Calculates the total price for multiple individuals.

10. _buildTransfersDescription(List<String> path)

Generates a description of the transfer stations (if any) between metro lines in the given path.

11. getTransfersDescription(List<String> path)

Public method that returns the transfers description for display purposes.



Station Controller

Reactive Variables

- `startStation`, `endStation`: Observable strings representing the start and end stations.
- `selectedLine`: Observable string representing the currently selected metro line.
- `stationNames`: List of all unique station names.

- `stationToLineMap`: Map linking each station to its corresponding line.
- `lineStations`: Stores all stations grouped by line.
- `transferStations`: Map defining transfer stations between any two intersecting lines.

Key Methods

1. onInit()

Initializes the controller and loads all stations with their associated line data.

2. loadAllStations()

- Loads stations from all three lines (First, Second, Third).
- Populates:
    - `stationNames` (unique station list),
    - `stationToLineMap` (for quick lookups),
    - `lineStations` (for graph generation).

3. getLineOfStation(String stationName)

Returns the metro line that a station belongs to.

4. updateSelectedLine(String line)

Updates the selected metro line.

5. getTransferStation(String start, String end)

- Identifies transfer stations between two different lines.
- Returns a user-friendly transfer message (in Arabic).
- Handles one or multiple transfer station options.

6. getRoute(String start, String end)

- Returns the full route between two stations using BFS.
- Translates each station name before returning.
- Used for displaying the trip route in UI.

7. getTransferStations(String start, String end)

- Analyzes the route and identifies transfer points.
- Checks where the line changes occur along the path.

8. _getOriginalStationName(String translated)

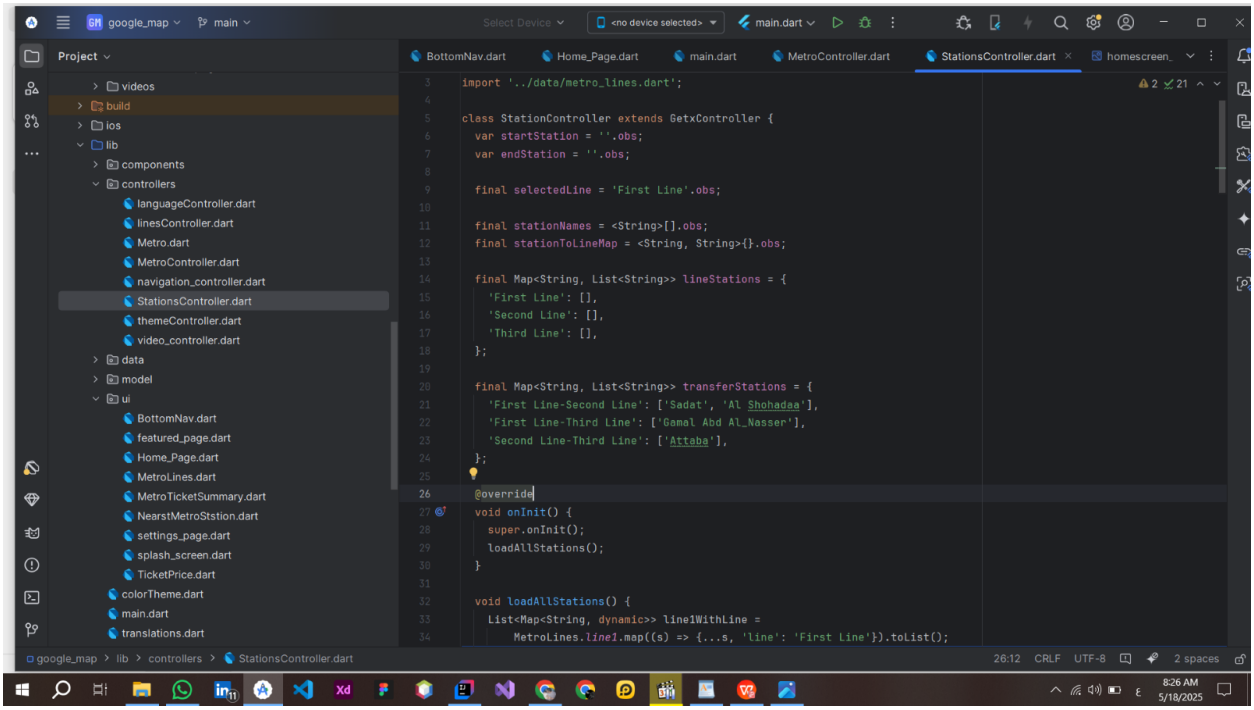
Retrieves the original station name from its translated form for internal logic use.

9. _translateStationName(String station)

Returns the translated station name using GetX's `.tr`.

10. _getConnectedStations(String station)

Returns the neighboring stations (left and right) of a given station on all metro lines.
Used for pathfinding logic.



Lines Controller :

Overview

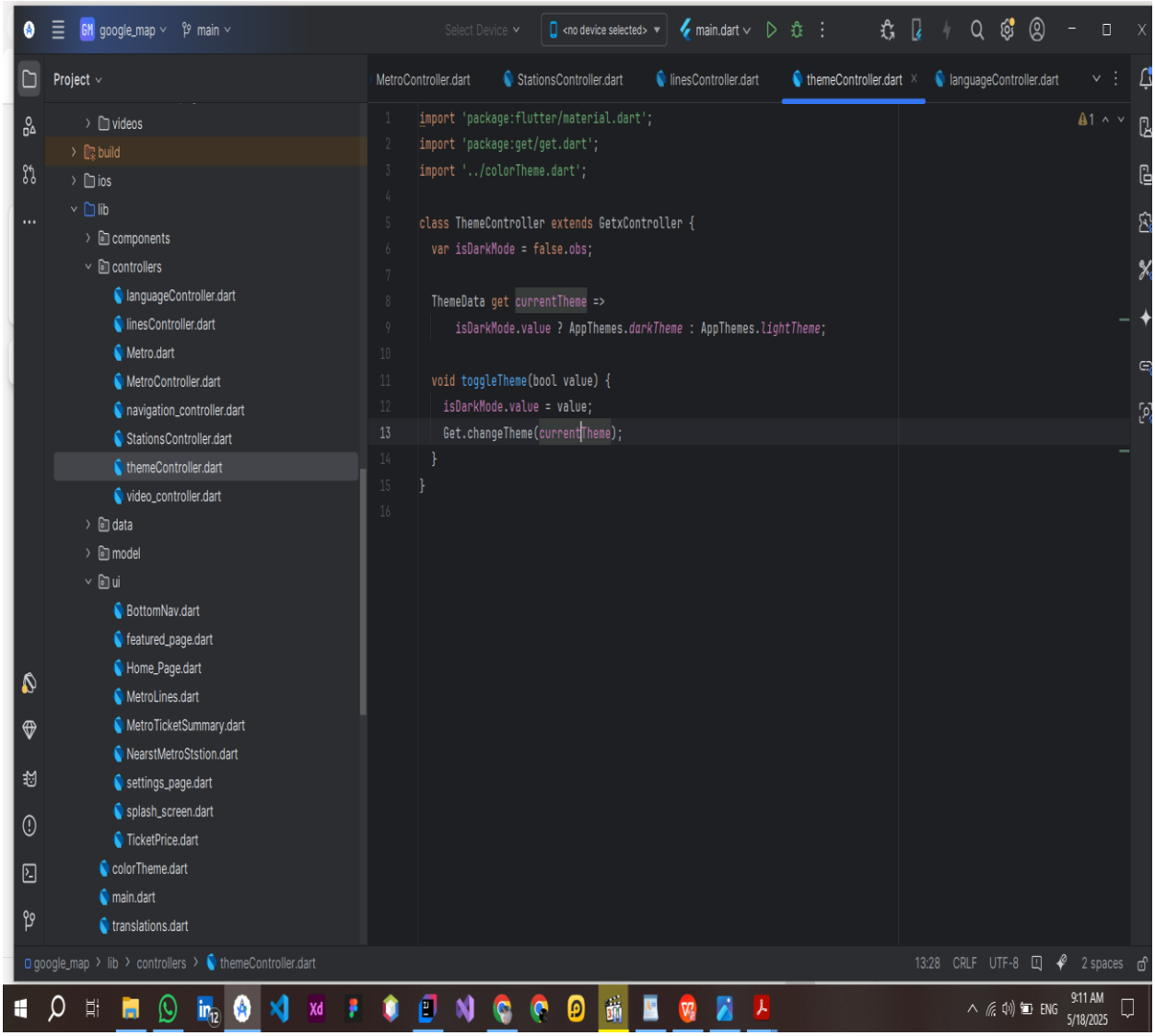
The `LinesController` manages the selected metro line and provides translated line names and stations for the currently active line. It uses GetX for reactive state management and localization.

Reactive Variables

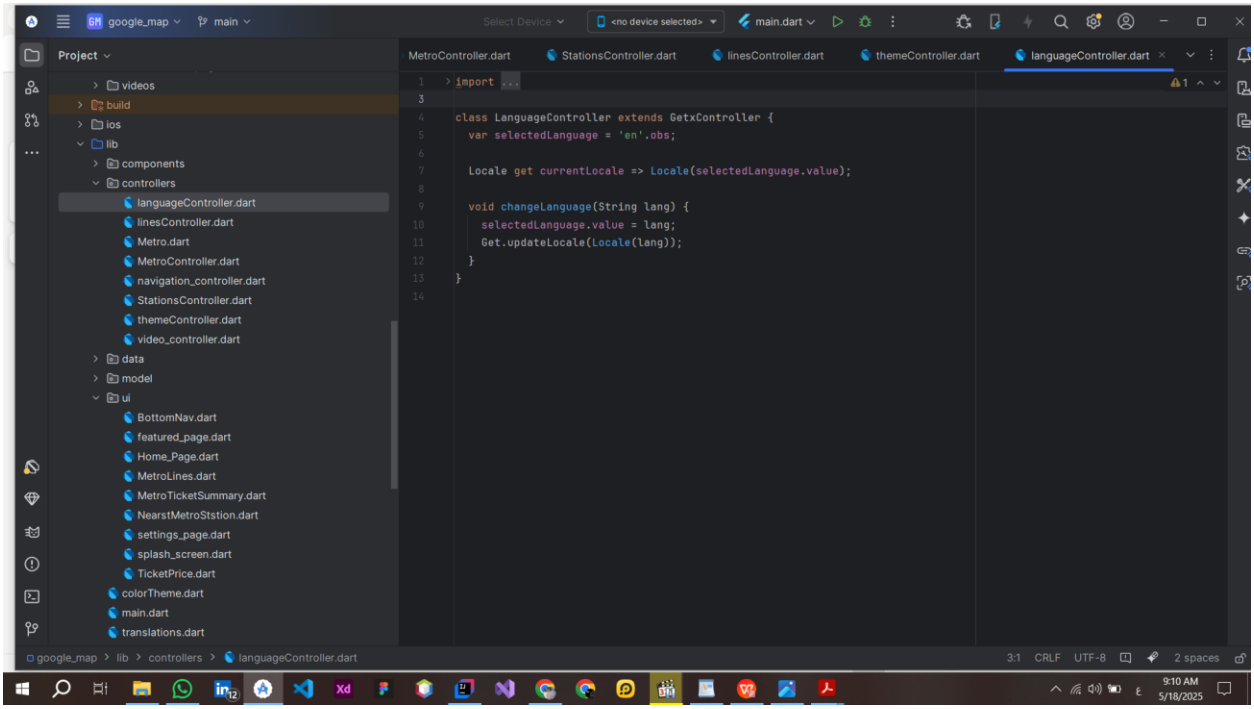
- `selectedLine`: An observable string representing the current metro line selected by the user.
  - Default: `'First Line'`.
  - **Constants**
- `rawLineKeys`: A list of raw translation keys used for metro lines.
  - `dart`
  - `CopyEdit`
  - `final List<String> rawLineKeys = ['first_line', 'second_line', 'third_line'];`
  - **Computed Properties**
  - ***1. `linesNames`***
  - Returns a list of localized line names using the `.tr` translation method.
  - **2. `linesTranslationMap`**
  - Re**3. `lines`**
  - Returns the current selected line as a reactive value.
  - turns a map of raw line keys to their localized names.

- 🔧 **Method**
- ***getStationsForSelectedLine()***
- Returns a list of stations for the currently selected metro line.

Theme Controller :



Language Controller :



Conclusion

The development of the Metro Cairo application represents a significant step toward

enhancing urban mobility through technology. By integrating real-time location tracking,
optimized route planning, and user-friendly interfaces, the app delivers a seamless
commuting experience for metro users in Cairo. Each service, from station detection to
trip progress tracking, was carefully designed with the user in mind, aiming to save time,
reduce confusion, and improve safety during travel.
We believe this project lays the groundwork for future innovations in smart public
transportation solutions and look forward to expanding its capabilities in future