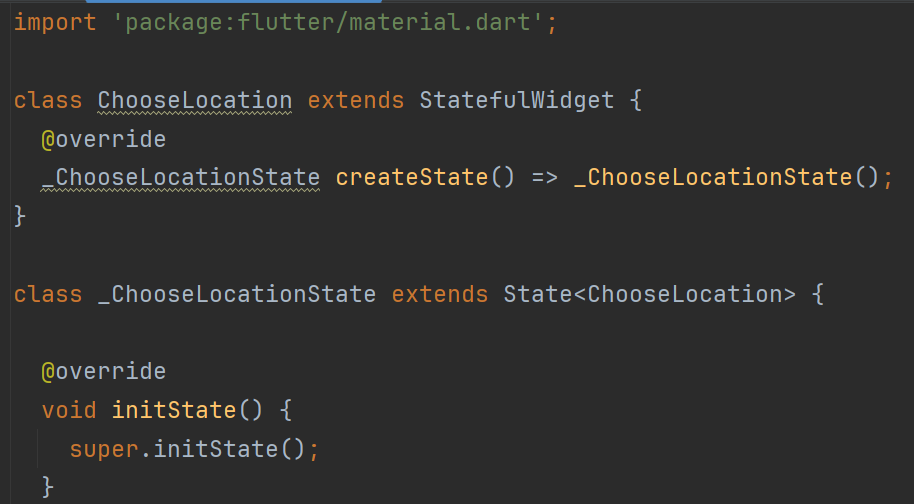
**FLUTTER**

* **runApp(MaterialApp())**- it is beneficial as it has some good default widgets like Scaffold
* **Dart** language is used
* **Scaffold** - It acts as a top-level container for organising the main visual elements of a Material Design app, such as the app bar, navigation drawer, floating action button, bottom navigation bar, and the main content area (body)
* **initialRoute**- it is a property of the MaterialApp widget that lets you specify which screen (route) should load first when the app starts
* **Routes** - other pages, including the home screen, which can be navigated to
* **Import** - helps import code and functions from other Dart files
* **StatelessWidget** -it is a widget that does not store or change any data during its lifetime. It builds its UI once and stays the same unless it’s rebuilt by a parent widget.
* **StatefulWidget**- it is a widget that can change its appearance or behaviour over time in response to events or user actions. It has a separate State class where you handle dynamic data and call *setState()* to update the UI.
* **async** and **await**- they are used in Dart to handle asynchronous operations like fetching data without blocking the main thread.  
   async marks a function as asynchronous, and await pauses its execution until a *Future* completes.
* **jsonDecode** -converts a JSON-formatted string into a Dart object like a Map or List
* **SafeArea**- it is a widget in Flutter that insets its child by sufficient padding to avoid intrusions by system UI elements, such as notches (on iPhones), status bars, or navigation bars
* **Pubspec.yaml** - it has dependencies like versions of packages and dependencies can added like location of images and urls.Once changes are made, we need to click on get dependencies



The code begins by importing the **Flutter Material** package.The ChooseLocation class extends **StatefulWidget**,the parent class. The createState method is overridden to return an instance of the private state class **\_ChooseLocationState**, which manages the widget's state. Inside **\_ChooseLocationState**, the initState method is overridden and calls super.initState(), ensuring that any initialisation logic from the parent class is executed. This structure sets up the foundation for a screen or component in Flutter that may need to update its UI dynamically in response to user actions or data changes. The function **initState()** run only when the widget is made.



This is an example of formatting and style in android studio.The code returns a Scaffold, which provides the basic visual layout for the screen. Inside the Scaffold’s body, a SafeArea widget ensures that the content is displayed within the safe, non-obstructed area of the device’s screen. The SafeArea contains a Column widget, which arranges its children vertically. Within the Column, there is an ElevatedButton.icon widget that displays a button with both an icon and a label. When this button is pressed, it uses the Navigator to push a new route named '/location', allowing navigation to a different screen.

**POSTGRESQL**

There are several GUI software for working with PostgreSQL like pgAdmin4 but I prefer working on the terminal. Some commands which I learnt:-

* **psql** – Start the PostgreSQL command-line client
* **CREATE DATABASE** – Create a new database
* **\l** – List all databases
* **\c [dbname]** – Connect to a specific database
* **DROP DATABASE** – Delete a database
* **CREATE TABLE** – Create a new table
* **\dt** – List all tables in current database
* **\d [tablename]** – Describe a table structure
* **DROP TABLE** – Delete a table (be careful)
* **INSERT INTO** – Insert new records into a table
* **SELECT \* FROM [table]** – Get all data from a table
* **WHERE** – Filter rows based on conditions
* **AND** – Combine multiple conditions
* **OR** – Match any of multiple conditions
* **NOT** – Negate a condition
* **ORDER BY** – Sort results by column
* **DISTINCT** – Return unique values
* **LIMIT** – Restrict number of rows returned
* **OFFSET** – Skip the first N rows
* **FETCH** – Retrieve a subset of rows (alternative to LIMIT)
* **IN** – Match any value in a list
* **BETWEEN** – Match values within a range
* **LIKE** – Match patterns in text (case sensitive)
* **ILIKE** – Match patterns in text (case insensitive)
* **GROUP BY** – Group rows by column(s)
* **HAVING** – Filter groups after grouping
* **MIN** – Find minimum value in a column
* **MAX** – Find maximum value in a column
* **AVG** – Calculate average value in a column
* **SUM** – Sum values in a column
* **COUNT** – Count rows or values
* **ROUND** – Round a numeric value
* **COALESCE** – Return first non-null value
* **NULLIF** – Return NULL if values match
* **NOW()** – Get current timestamp
* **CURRENT\_DATE** – Get current date
* **CURRENT\_TIME** – Get current time
* **EXTRACT** – Extract part of a date/time
* **AGE** – Calculate interval between two dates
* **PRIMARY KEY** – Set a column as primary key
* **UNIQUE** – Ensure column values are unique
* **UPDATE** – Modify existing rows
* **ON CONFLICT DO NOTHING** – Ignore conflicts on insert
* **ON CONFLICT DO UPDATE** – Update on conflict (upsert)
* **FOREIGN KEY** – Create a relationship between tables
* **REFERENCES** – Refer to another table’s column
* **INNER JOIN** – Join tables based on matching values
* **LEFT JOIN** – Join tables, keeping all left table rows
* **RIGHT JOIN** – Join tables, keeping all right table rows
* **FULL JOIN** – Join tables, keeping all rows from both
* **BIGSERIAL** – Large auto-incrementing integer
* **SEQUENCE** – Create a sequence for custom auto-increment
* **\q** – Quit psql session
* **\i [file]** – Execute SQL from a file
* **ALTER TABLE** – Modify table structure
* **ADD COLUMN** – Add a column to a table
* **DROP COLUMN** – Remove a column from a table
* **RENAME COLUMN** – Rename a column
* **RENAME TABLE** – Rename a table
* **ALTER COLUMN TYPE** – Change column data type
* **SET NOT NULL** – Make column non-nullable
* **ADD CONSTRAINT** – Add a constraint to a table
* **DROP CONSTRAINT** – Remove a constraint from a table
* **CREATE FOREIGN TABLE** – Create a foreign table
* **DROP FOREIGN TABLE** – Remove a foreign table