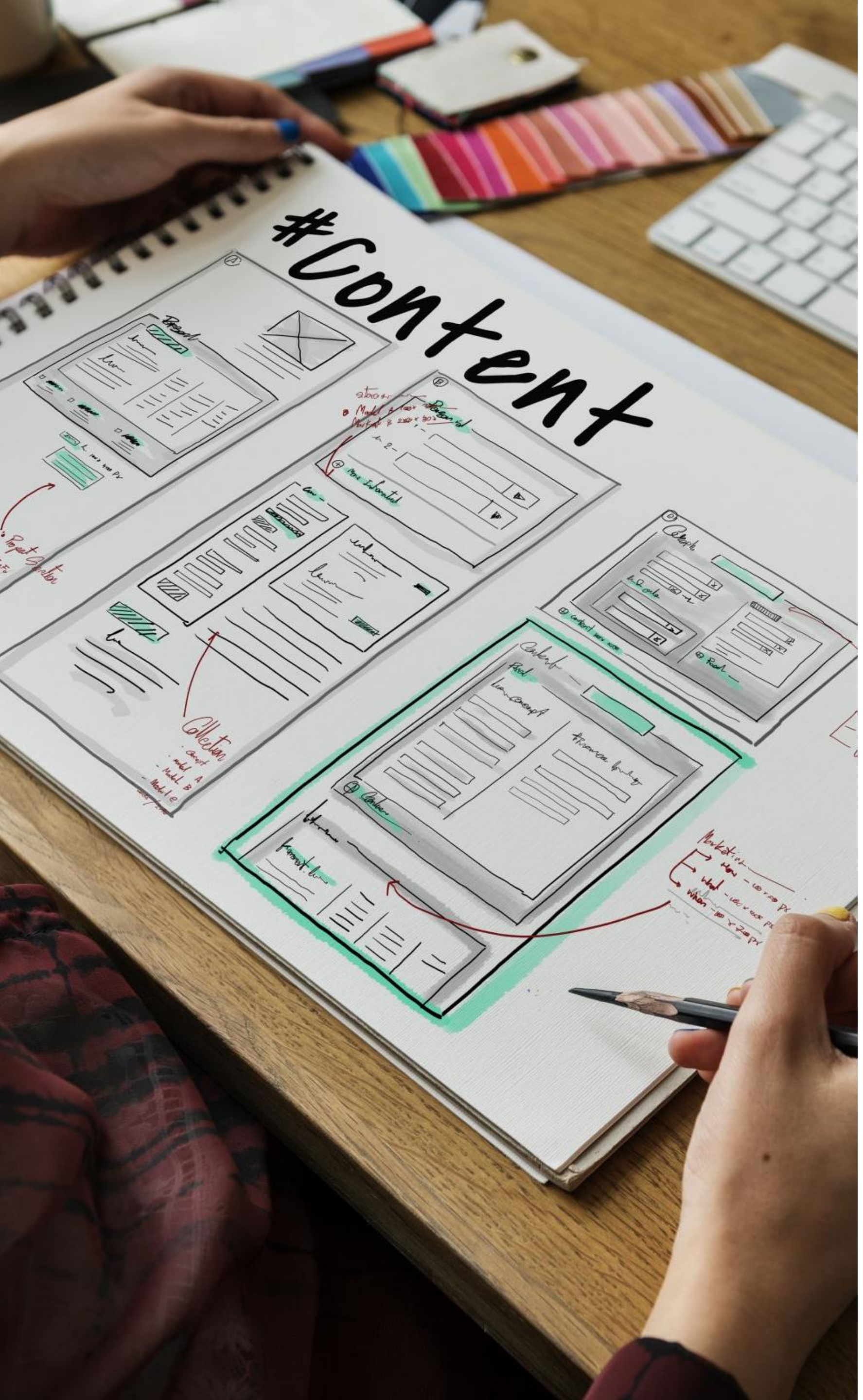
Project Title: Money Matters: A Personal Finance Management App

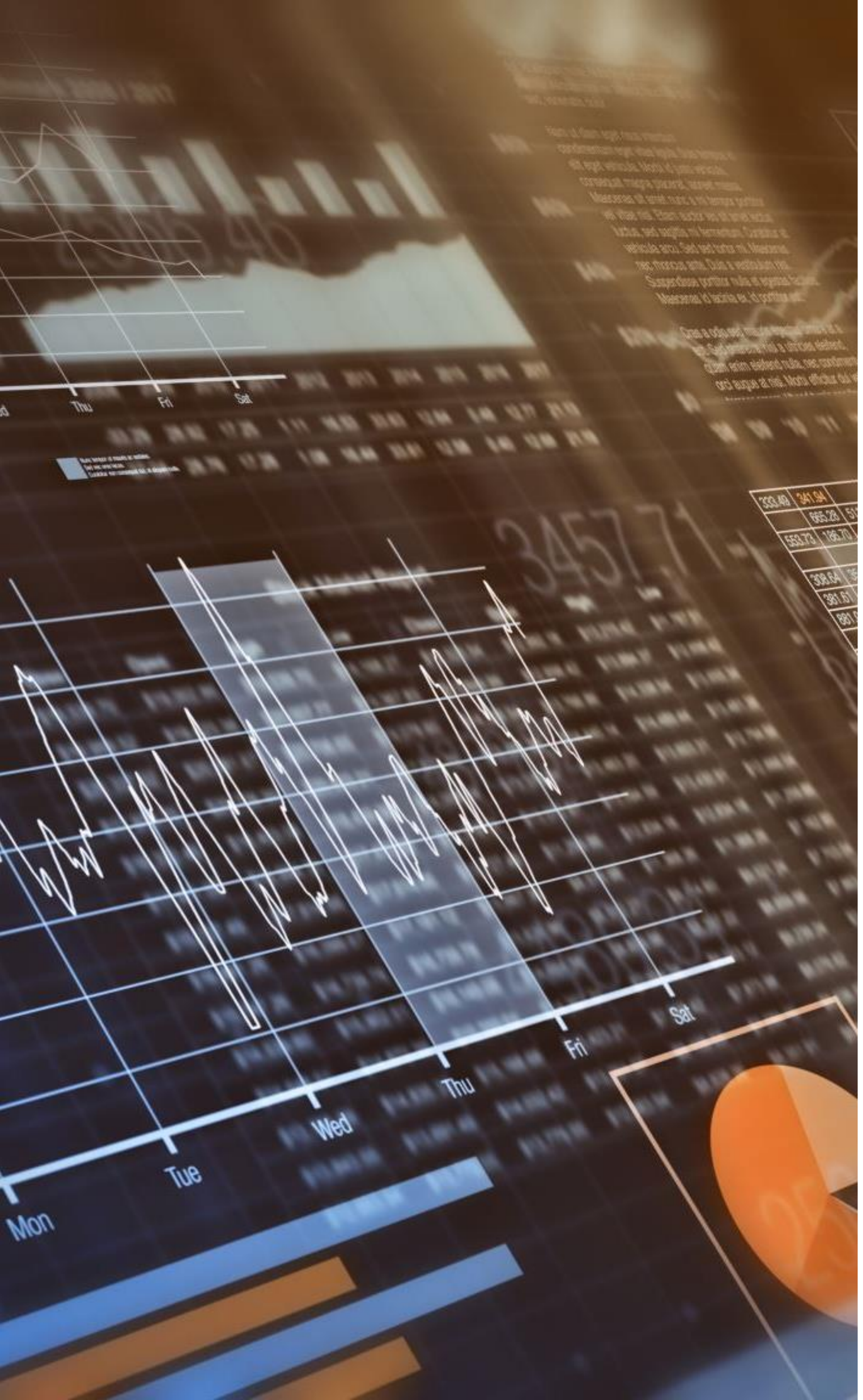
- **Project Description:**
- Money Matters is a personal finance management app designed to help users track their expenses, manage accounts, and get a clear overview of their financial status. Users can set budgets for different expense categories (e.g., groceries, entertainment) and monitor their progress. The app enables users to take control of their finances, avoid overspending, and improve their financial planning.

- PRESENTED BY:
- V.ARAVINTH
- M.ARAVINTH
- P.ABINAYA
- B.BASKAR



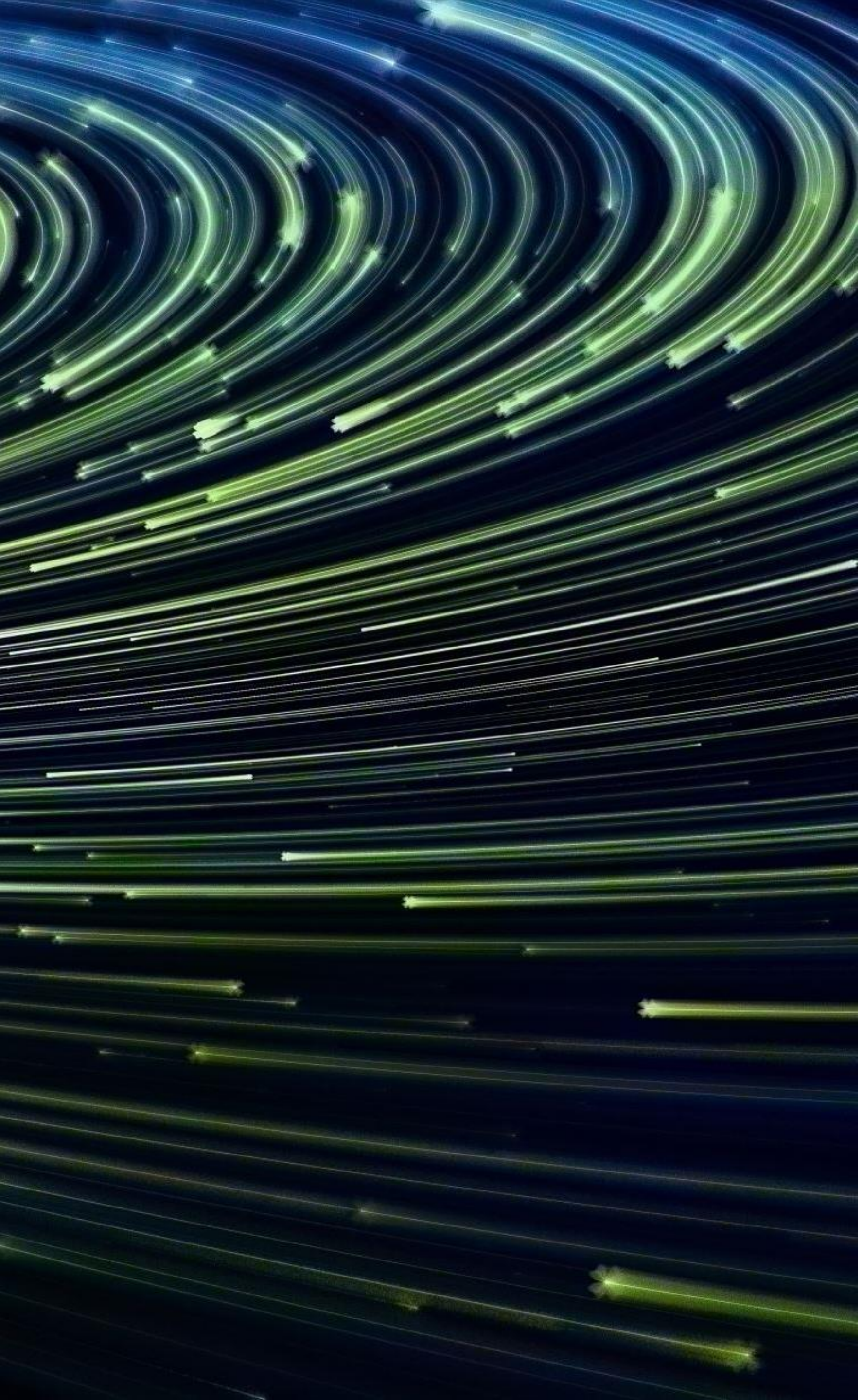
Architecture:

- **Frontend (UI):** The app's user interface will be built using Android Studio, adhering to Material Design principles for an intuitive and clean layout.
- **Backend (Database):** A local SQLite database will store user data such as expenses, accounts, and budget information. The app will also support basic data synchronization with online cloud storage if necessary.
- **Data Flow:** Users input their expenses and accounts, set budgets, and the app will provide real-time tracking and visualization of their spending against the set budget.



Learning Outcomes:

- **By the end of this project**:
- You'll have hands-on experience with Android Studio and app development.
- You'll understand how to integrate local databases like SQLite into an Android app.
- You'll be able to implement core functionalities like user authentication, CRUD operations for financial data, and UI design.



Project Workflow:

1.User Registration:

Users register with basic information such as name, email, and password.

2.Login:

After registration, users can log in to the app using their credentials.

3.Main Page:

After logging in, users enter the main dashboard where they can add expenses, view their financial status, set budgets, and see progress.

Tasks Involved:

- **Required Initial Steps:**

  Set up Android Studio and configure the necessary SDK versions.

- **Creating a New Project:**

  Create a new project with the desired app name, package name, and basic structure.

- **Adding Required Dependencies:**

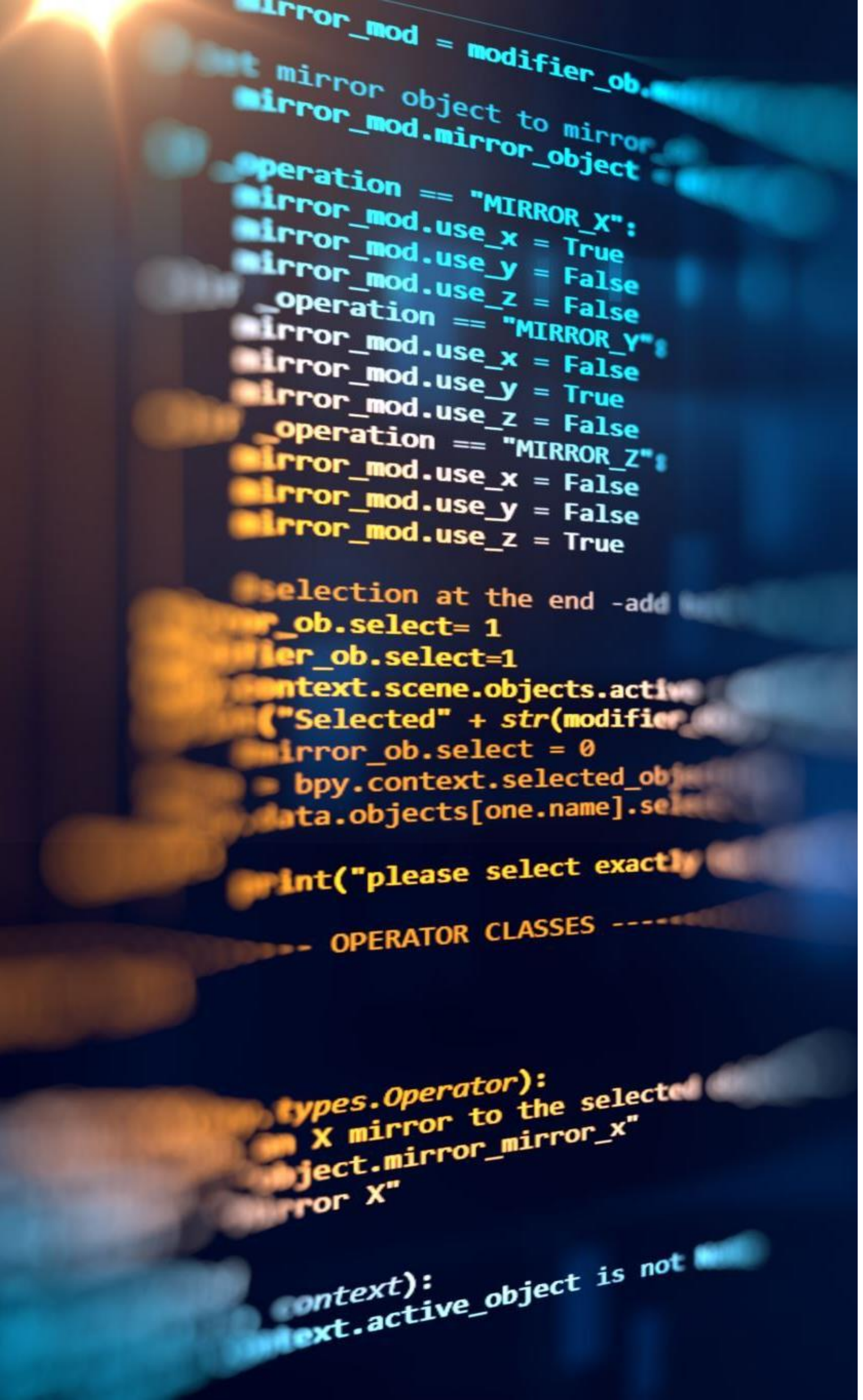
  Add dependencies for SQLite database integration, Material Design components, and any other libraries needed for the project.

- **Creating the Database Classes:**

  Create database helper classes to handle user data, including expenses, accounts, and budgets. Implement CRUD operations (Create, Read, Update, Delete).

- **Building Application UI and Connecting to the Database:**

  Develop the user interface using XML layouts, integrating elements like buttons, text fields, and lists. Connect the UI to the database using appropriate Android components (e.g., RecyclerView, ViewModel).

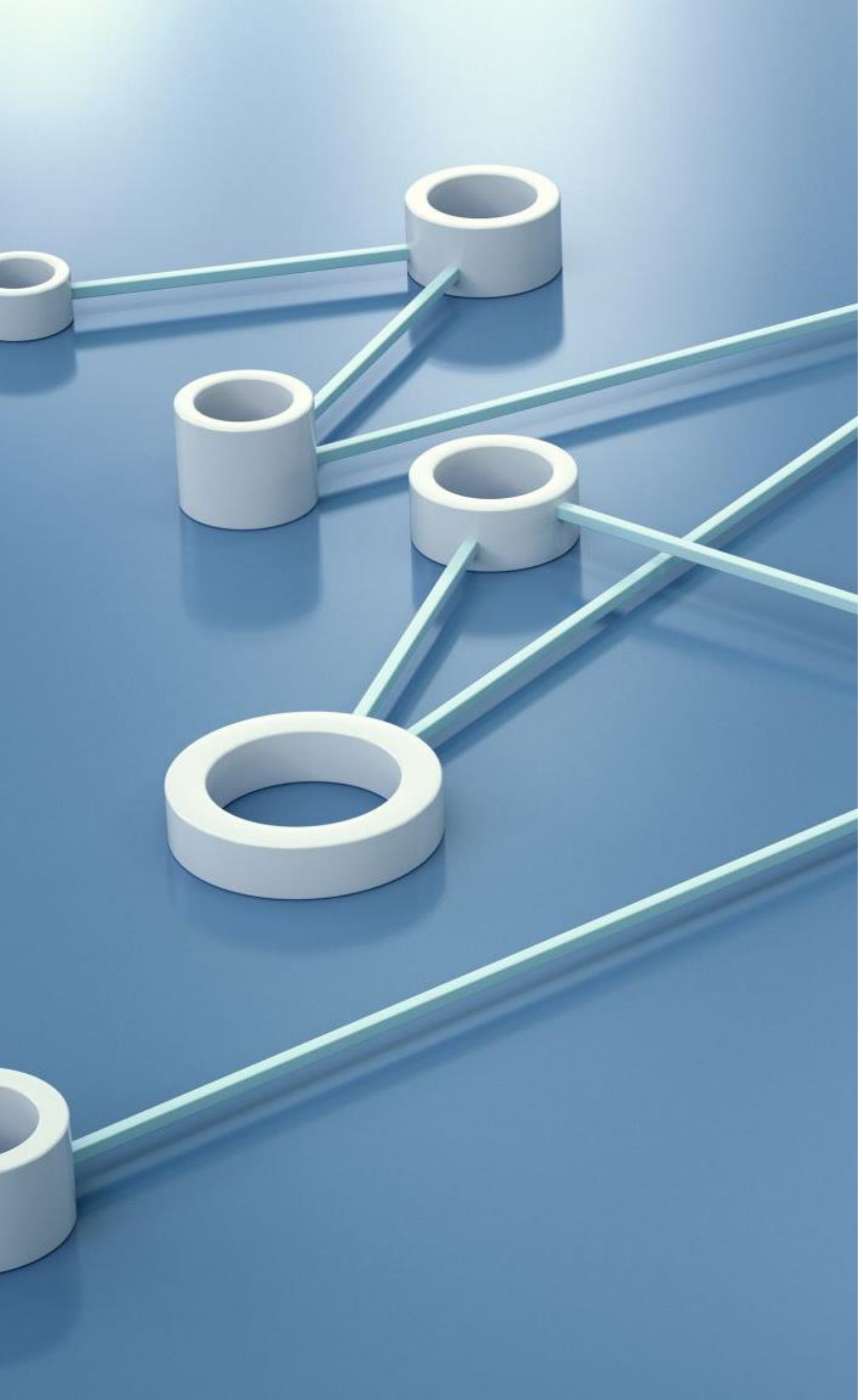


- **Using AndroidManifest.xml:**

    Define required permissions (e.g., internet access, storage) in the AndroidManifest.xml file.

- **Running the Application:**

    Test the app on an emulator or physical device, debug, and refine the user experience.



Key Activity And It's Power:

- UserDao.kt;

-To Acces User DataBase We Need A Magic Stick That's refered An Object Called Dao

-DAO-> Stands For DATA ACCESS OBJECT

AppDataBase.kt;

-For Creating Instance For Database

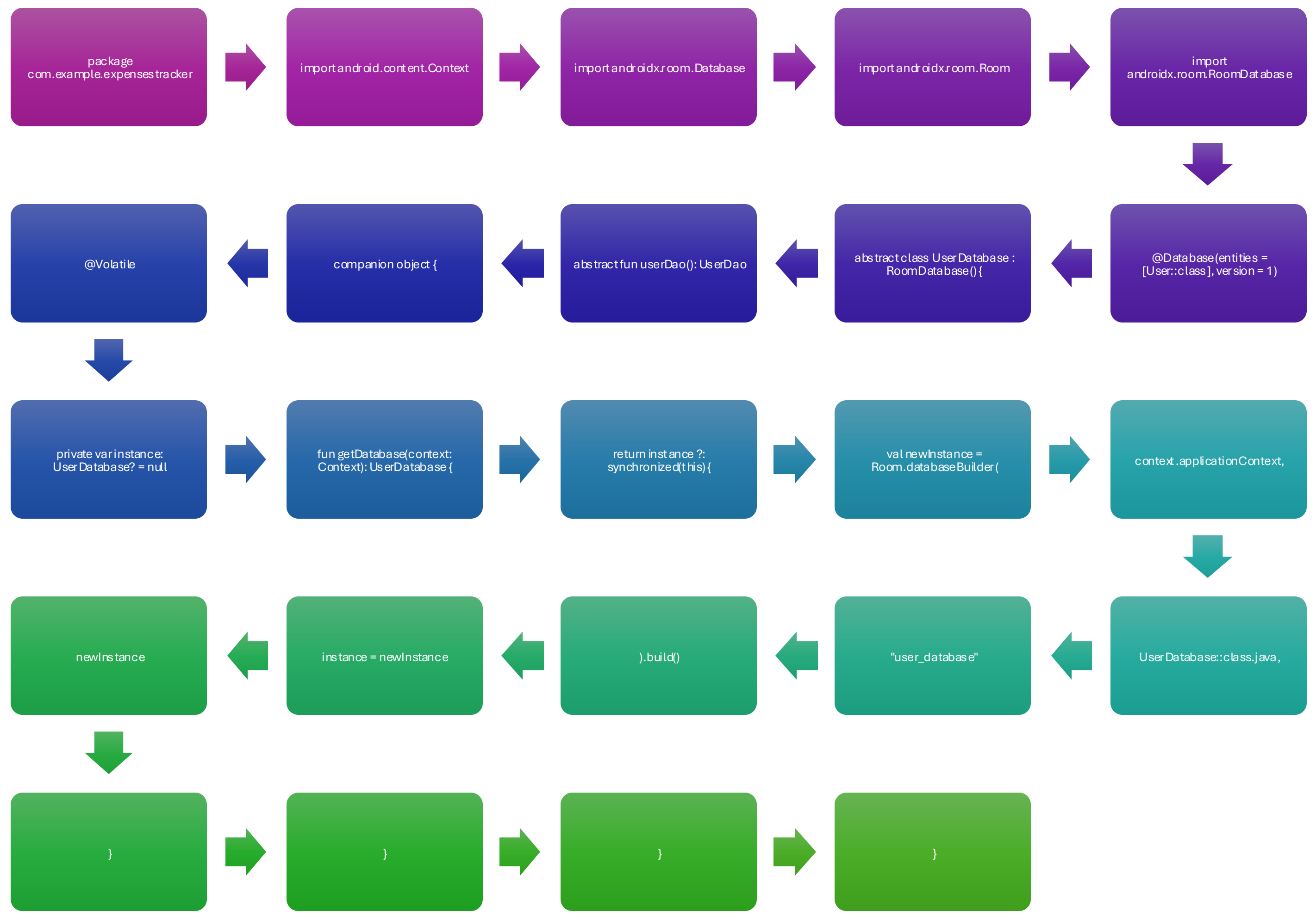
Login.kt;

-For User Login iff uswer already logined,other wise user need to signup using Register.kt

UserDao.kt:

```
•  package com.example.expensestracker

•  import androidx.room.*

•  @Dao
•  interface UserDao {

•      @Query("SELECT * FROM user_table WHERE email = :email")
•      suspend fun getUserByEmail(email: String): User?

•      @Insert(onConflict = OnConflictStrategy.REPLACE)
•      suspend fun insertUser(user: User)

•      @Update
•      suspend fun updateUser(user: User)

•      @Delete
•      suspend fun deleteUser(user: User)
•  }
```

UserDatabase.kt :

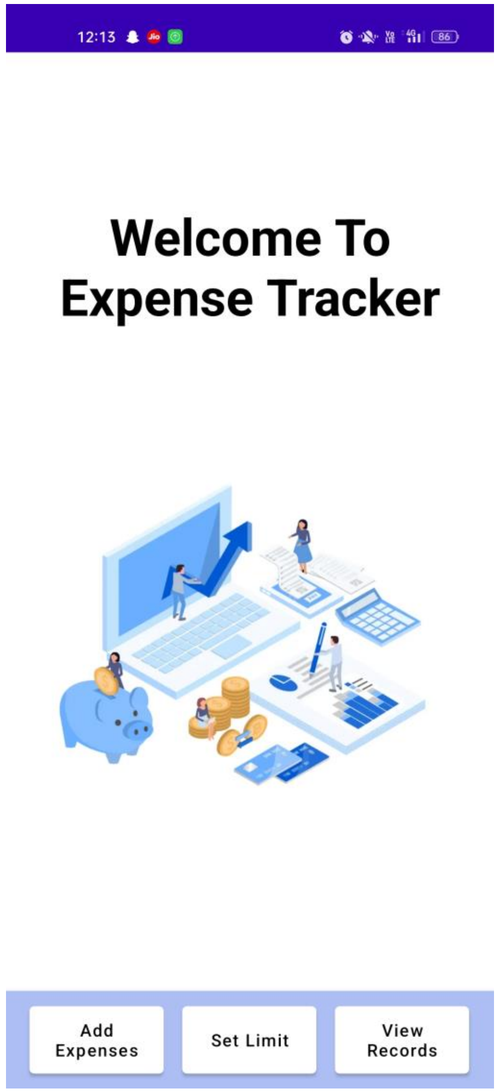


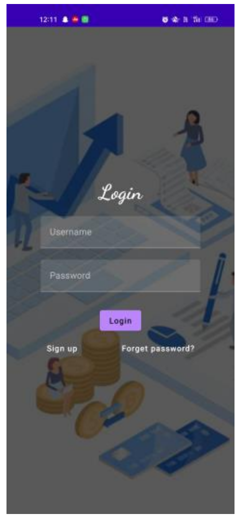
```
package com.example.expensestracker
import android.content.Context
import androidx.room.Database
import androidx.room.Room
import androidx.room.RoomDatabase
@Database(entities = [User::class], version = 1)
abstract class UserDatabase : RoomDatabase() {
abstract fun userDao(): UserDao
companion object {
@Volatile
private var instance: UserDatabase? = null
fun getDatabase(context: Context): UserDatabase {
return instance ?: synchronized(this) {
val newInstance = Room.databaseBuilder(
context.applicationContext,
UserDatabase::class.java,
"user_database"
).build()
instance = newInstance
newInstance
}
}
}
}
```

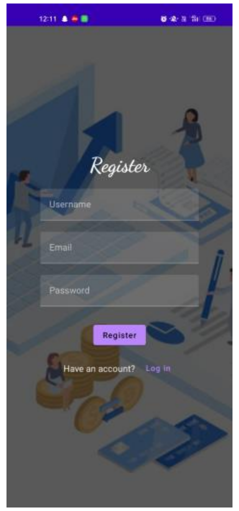
Login.kt:

- class LoginActivity : ComponentActivity() { private lateinit var databaseHelper: UserDatabaseHelper override fun onCreate(savedInstanceState: Bundle?) { super.onCreate(savedInstanceState) databaseHelper = UserDatabaseHelper(this) setContent { ExpensesTrackerTheme { // A surface container using the 'background' color from the theme Surface(modifier = Modifier.fillMaxSize(), color = MaterialTheme.colors.background) { LoginScreen(this, databaseHelper) } } } } }

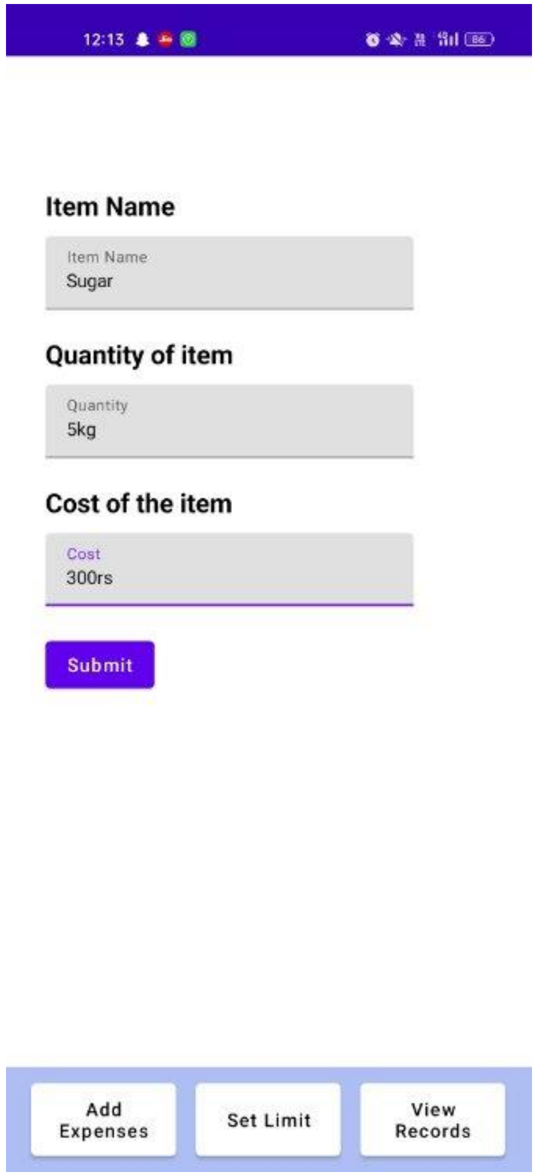
- Work Flow:
- 1.Register
- 2.Login
- 3.MainPage







- THANKING YOU!


12:13

Item Name

Item Name
Sugar

Quantity of item

Quantity
5kg

Cost of the item

Cost
300rs