### Adaptive Client Email

### TEAM MEMBERS :

### S.KIRUVABATHI(1E0F17C0C1CD505986345B635BE0CB52)

### R.KANIGA(5C46859DF85C76BB7D2457ED1CF4A142)

### K.SOWBAKKIYALAKSHMI(544C6F3316B0E8EB1224FCA0F0B267CE)

### T.MANIKANDAN(1579EE54F9891F882C041444247F6497)

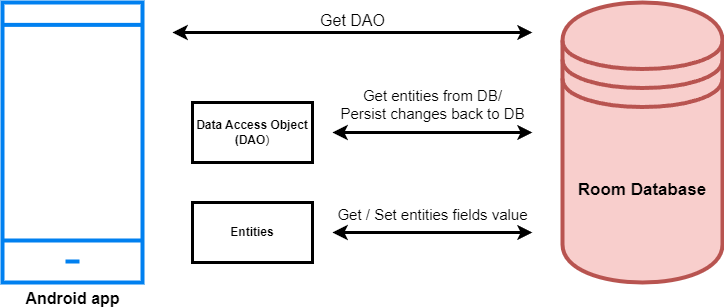
### M.RAMKUMAR(74BEB8621D244AC9C514BA1D1929B7FB)

### Adaptive Mail: A Flexible Email Client App

Project Description:

Adaptive Mail app is a sample project that demonstrates how to use the Android Compose UI toolkit to build a conversational UI. The app simulates a messaging interface, allowing the user to send and receive messages, and view a history of previous messages. It showcases some of the key features of the Compose UI toolkit, data management, and user interactions.

Architecture



Learning Outcomes :

By end of this project:

* You’ll be able to work on Android studio and build an app.
* You’ll be able to integrate the database accordingly.

Project Workflow:

* Users register into the application.
* After registration , user logins into the application.
* User enters into the main page
* User can View previously sent emails.
* User can give subject and email body to send email.

Note:

To complete the project you need to finish up the tasks listed below:

Tasks:

1.Required initial steps

2.Creating a new project.

3.Adding required dependencies.

4.Creating the database classes.

5.Building application UI and connecting to database.

6.Using AndroidManifest.xml

7.Running the application.

**Required initial steps**

In this milestone, we will go through the required initial steps

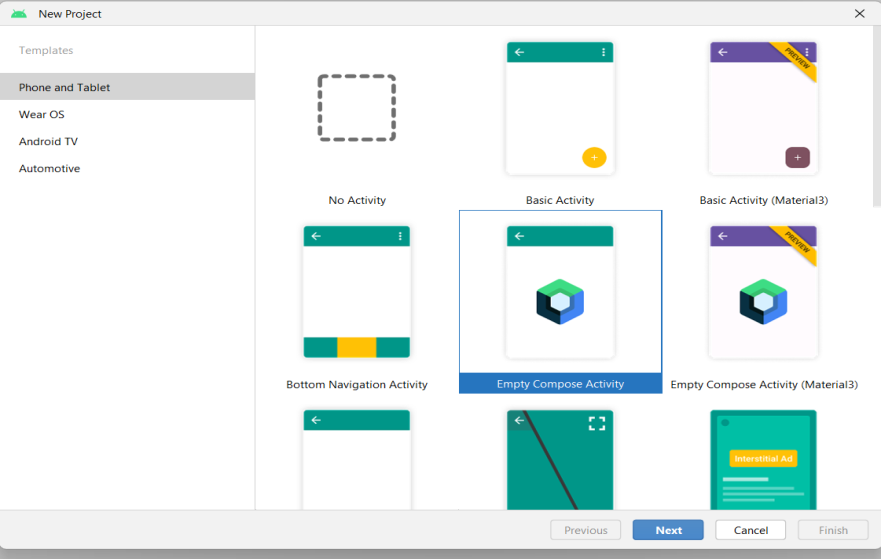
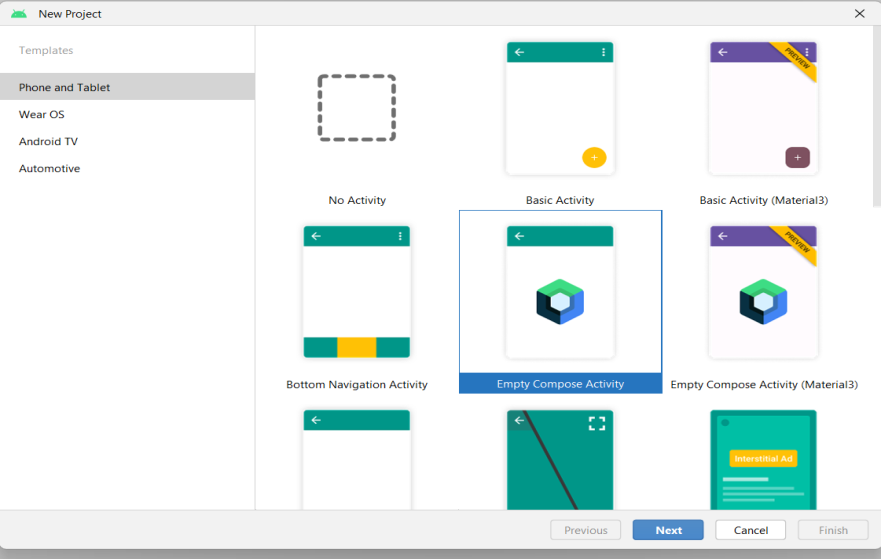
<https://developer.android.com/studio/install>

### Creating a new project

Creating a new project.

Step 1 : Android studio > File > New > New Project > Empty Compose Activity

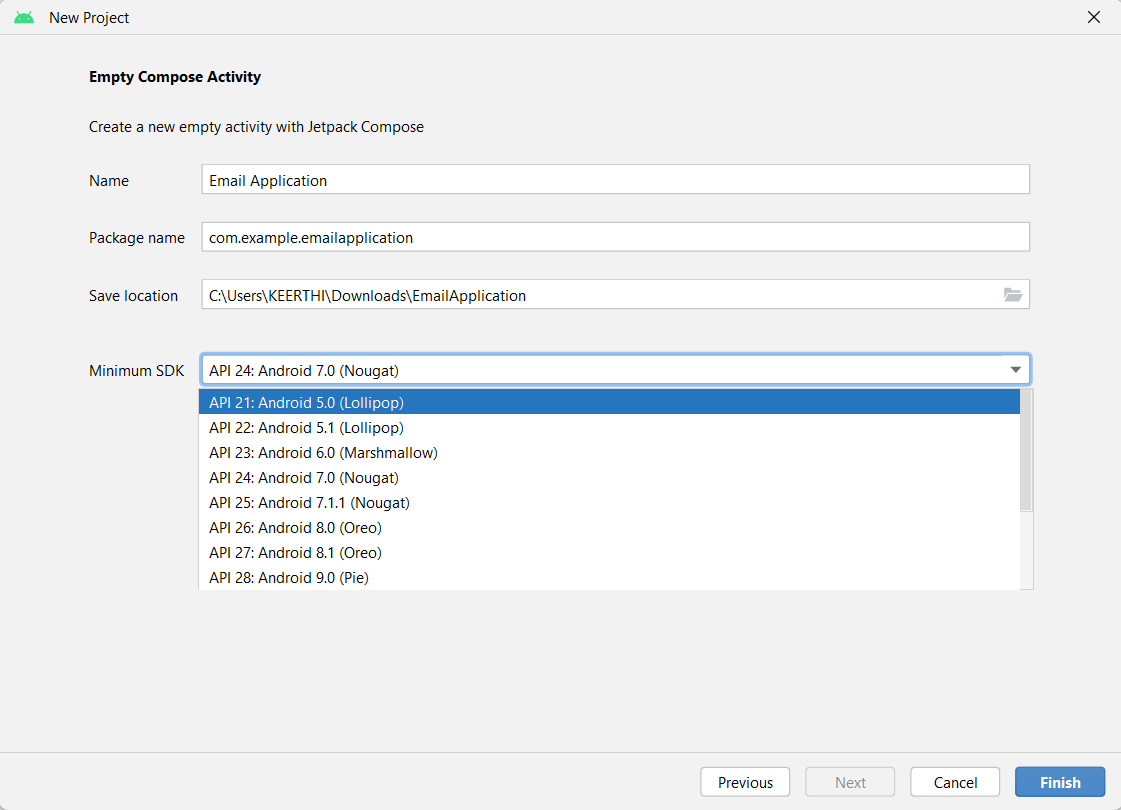
Step 2 : Click on Next button.



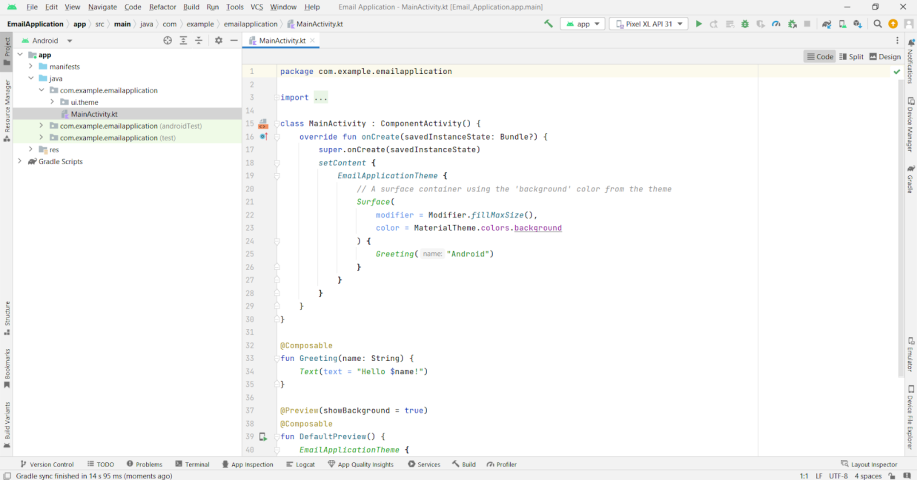
Step 3 : Give name to the new project.

Step 4 : Give the Minimum SDK value

Step 5 : Click Finish



Main activity file

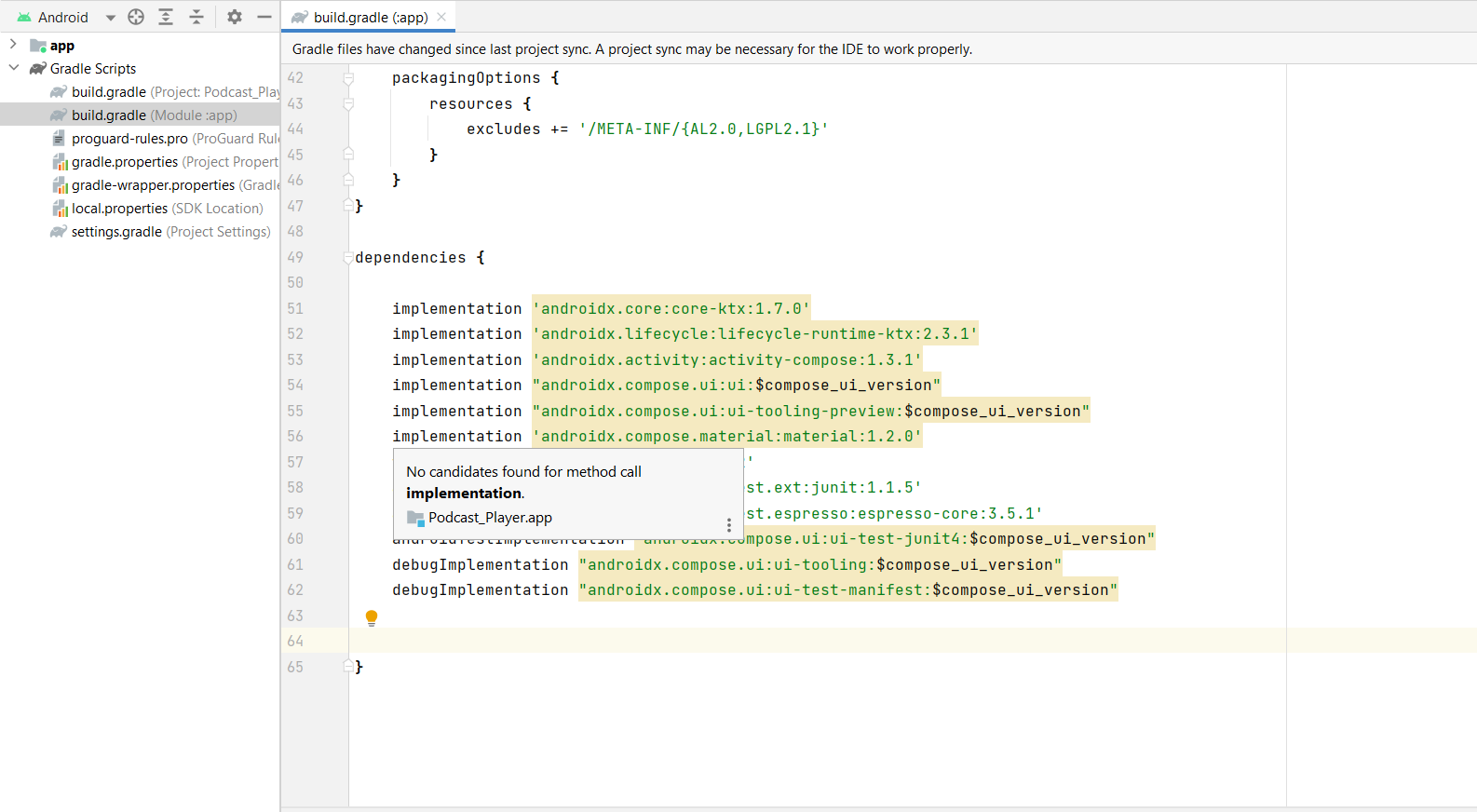


### Adding required dependencies

This milestone explains about adding required dependencies.

### Gradle scripts > build.gradle(Module :app)

**Gradle scripts > build.gradle(Module :app)**



### LITERATURE SURVEY

### Problem Statement:

### This fragmentation makes it difficult for businesses to create email messages that look and function properly across all devices and platforms. In addition, recipients have different preferences and behaviors when it comes to email, so a one-size-fits-all approach to email marketing is no longer effective.

### Proposed Solution:

### •The purpose of adaptive email is to provide a solution to the challenges posed by the diversity of devices, platforms, and customer preferences in email marketing.Users

## Email client vs webmail

Having an email client includes downloading actual software and installing it on your Desktop or Mobile device and configuring, mostly manually, one of your email’ protocols (POP3, SMTP, or IMAP). These protocols manage the delivery and transmission of emails, ensuring that email clients can send and receive messages efficiently. Although users don't interact with these processes directly, they function behind the scenes to facilitate seamless communication.

While it may seem cumbersome to add additional details to the app, compared to the simple cloud-based email management offered by the likes of [Google](https://www.techradar.com/tag/google) and Yahoo; an email client offers much more functionality and flexibility. Here are some of the major differences between an email client and its Webmail counterpart.

### Adding room dependencies

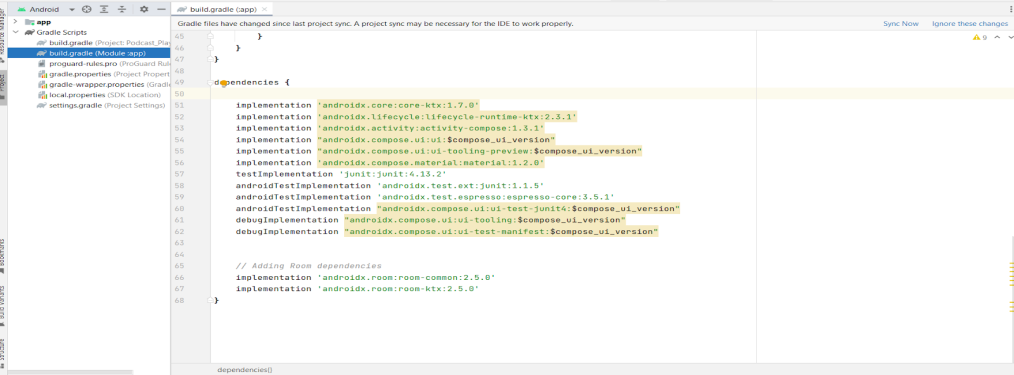
Adding room dependencies

Add the below code in dependencies

// Adding Room dependencies

implementation 'androidx.room:room-common:2.5.0'

implementation 'androidx.room:room-ktx:2.5.0'



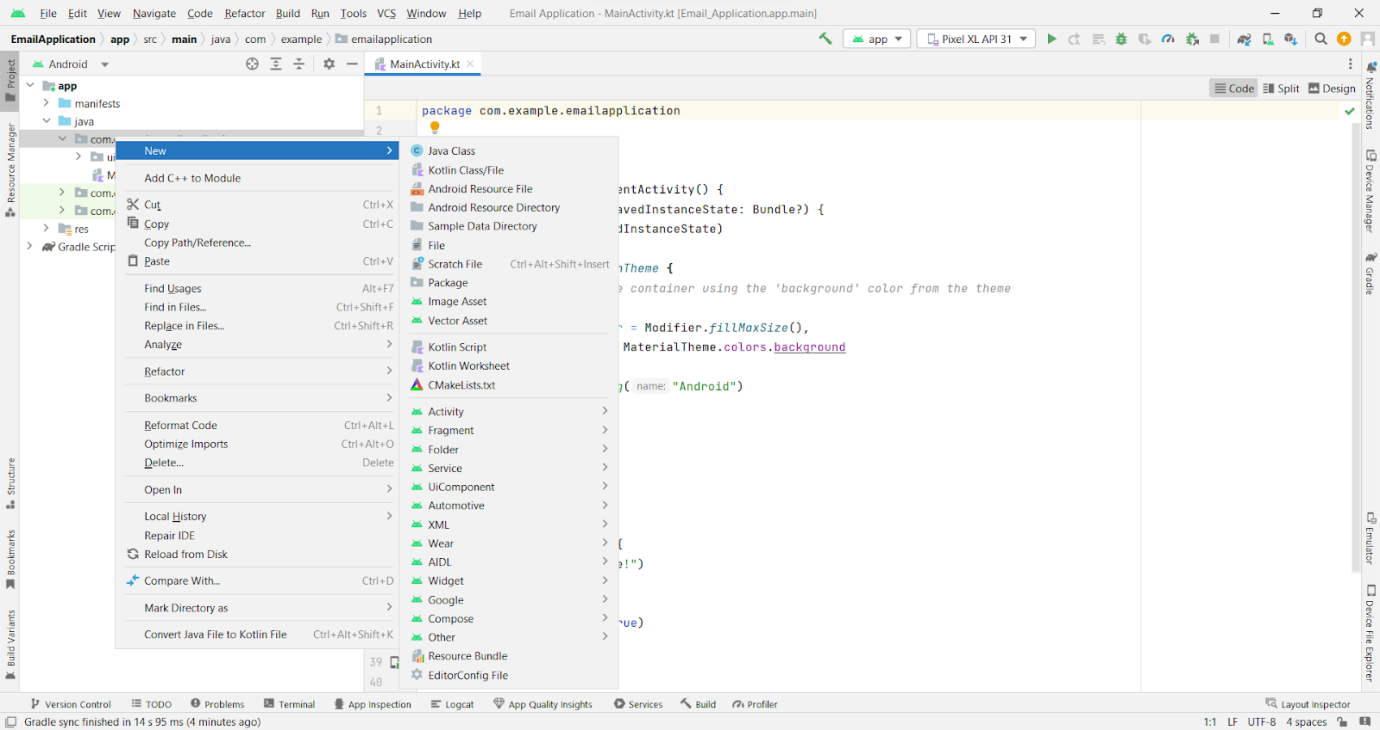
### Creating the database classes

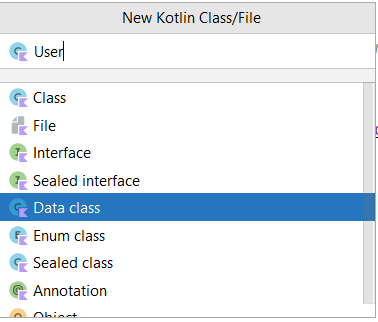
**Creating the database classes**

In this project we will be having two databases, one is for user registration and login, and other is for admin to view the details of survey forms.

### Database 1 (Create User data class)

Create User data class





User data class code:

<https://github.com/smartinternz02/AdaptiveEmailApp/tree/main/app/src/main/java/com/example/emailapplication/User.kt>

## Email Templates for Asking for Documents from Clients

Email is an essential tool for communicating with clients, especially when asking for important documents.

Using our well-crafted email templates can make your and your client's lives better by making your services smoother and more efficient.

Here are 10 email templates, each tailored for different situations in requesting documents from clients. Each template includes reasons why it's important and how it can be effectively used.

### ****Template #1: Initial Request for Documents****

This template sets the tone for your professional relationship. It's polite and clear, providing clients with specific details about what you need.  
***Email Template:***  
**Subject:**Documents Needed for [Service Name] Processing  
Hi [Client's Name],  
I hope you're well. To proceed with your [service name], could you please send us the following documents: [list documents]? Let us know if you have any questions about these. Thanks for your cooperation!

### ****Template #2: Request for Complex Documents****

Some documents you'll be collecting are complex and it's best to let your client know that you're aware of that. This template helps clarify exactly what you need which should help reducing confusion.

***Email Template:***  
**Subject:** Additional Information Required for Your File  
Hi [Client's Name],  
To ensure accurate processing of your file, we need more detailed documents, specifically [describe documents and purpose]. Could you please check and send these at your earliest convenience? Thank you for your attention to this detail.

### ****Template #3: Reminder for Pending Documents****

Reminders are crucial for timely submissions. This template is a gentle nudge for clients to send over documents.

***Email Template:***  
**Subject:**Reminder: Pending Documents for [Service Name]  
Hi [Client's Name],  
We noticed that we still haven't received the following documents from you: [list documents]. To keep your file moving, please send them as soon as possible. Thanks for your prompt response!

### ****Template #4: Urgent Request for Immediate Action****

Sometimes, you need documents urgently. This template conveys urgency without being pushy.

**But, be careful not to come across as a phishing or scam email**, as many illegitimate emails have a similar urgency to them. To avoid this, you could be sure to include your email signature, or change the tone of the template to match the style of speaking your clients are used to.

***Email Template:***  
**Subject:** Urgent: Immediate Action Required for Document Submission  
Hi [Client's Name],  
We urgently require [list documents] to proceed further. Please send them by [specific deadline] to avoid any delays. We appreciate your immediate attention to this matter.

### ****Template #5: Follow-Up After Initial Request****

Following up after your initial request reinforces its importance. This template keeps the conversation going.

***Email Template:***  
**Subject:**Following Up on Earlier Document Request  
Hi [Client's Name],  
I hope this message finds you well. I wanted to follow up on my previous email regarding [list documents]. Your timely submission is essential for us to move forward.

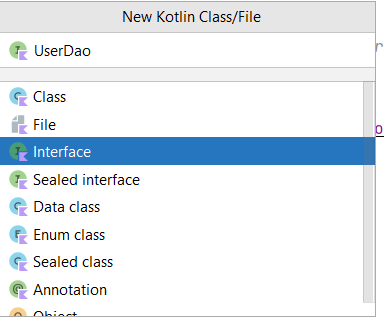
Use this template to get some clarity on a part of your client's documents.

***Email Template:***  
**Subject:** Clarification Needed on Received Documents  
Hi [Client's Name],

### Create an UserDao interface

Create an UserDao interface



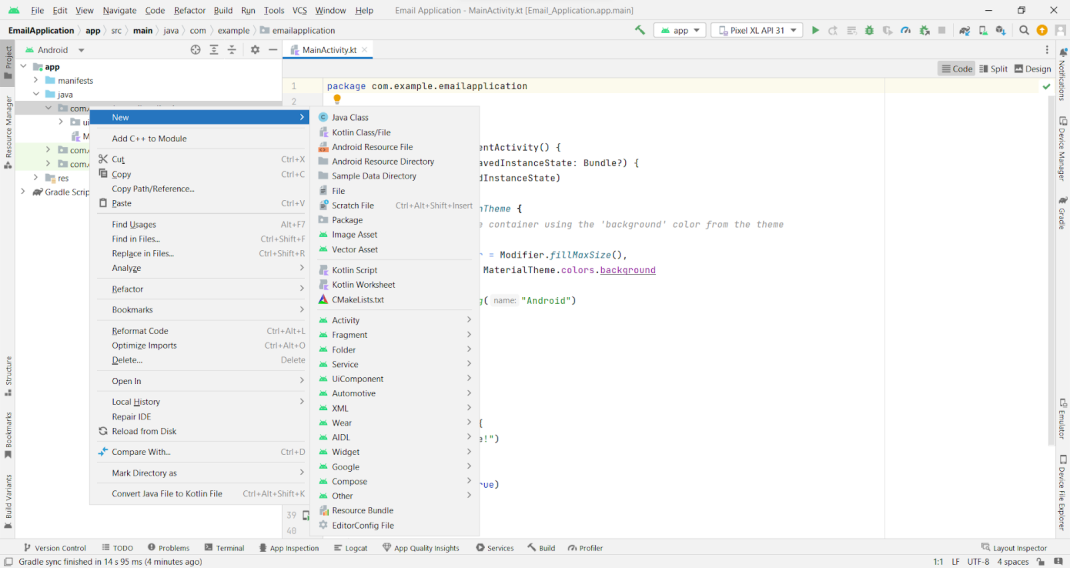


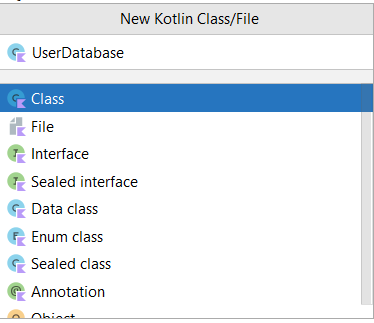
UserDao interface code:

<https://github.com/smartinternz02/AdaptiveEmailApp/tree/main/app/src/main/java/com/example/emailapplication/UserDao.kt>

### Create an UserDatabase class

Create an UserDatabase class



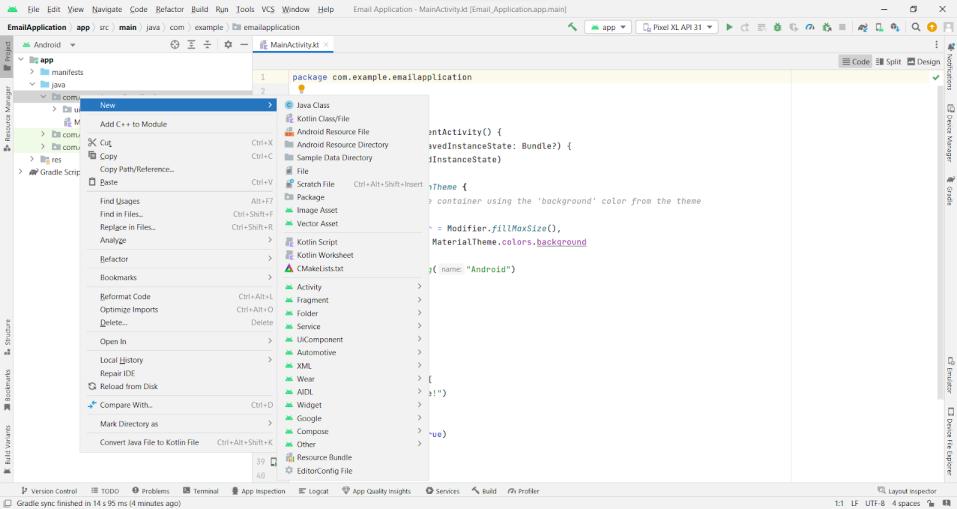


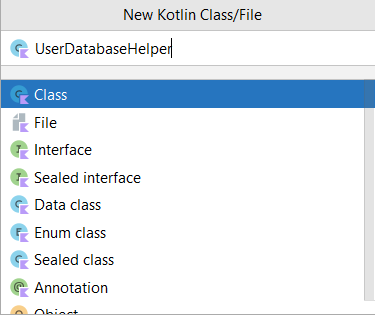
UserDatabase class code :

<https://github.com/smartinternz02/AdaptiveEmailApp/tree/main/app/src/main/java/com/example/emailapplication/UserDatabase.kt>

### Create an UserDatabaseHelper class

Create an UserDatabaseHelper class





UserDatabaseHelper class code :

<https://github.com/smartinternz02/AdaptiveEmailApp/tree/main/app/src/main/java/com/example/emailapplication/UserDatabaseHelper.kt>

### THEORITICAL ANALYSIS

### Block diagram:

### 

### FLOWCHART

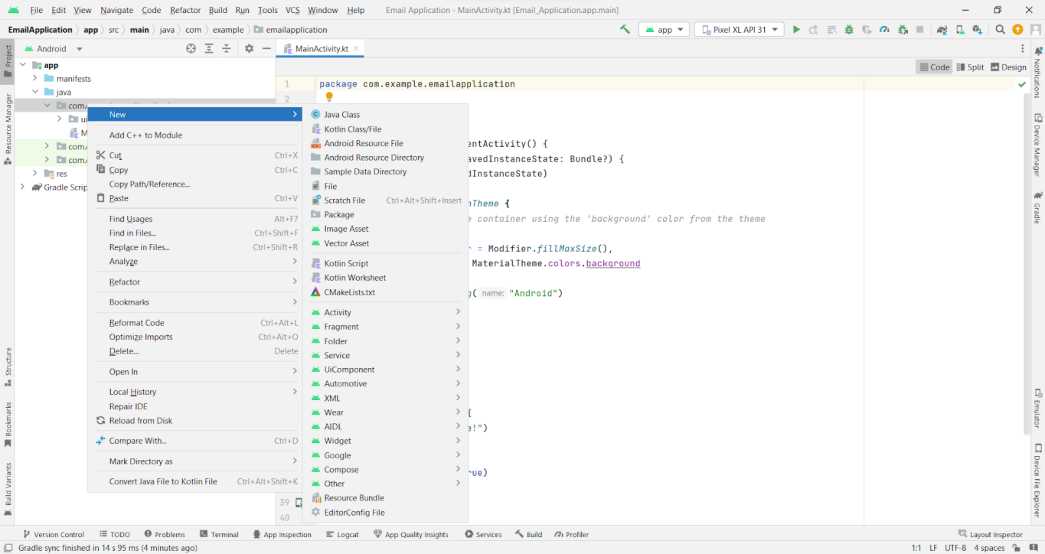
### 

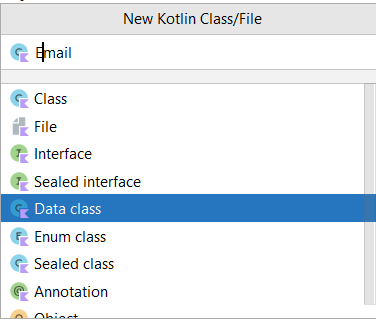
### Database 2

This milestone explains about Database 2

### Create Email data class

Create Email data class



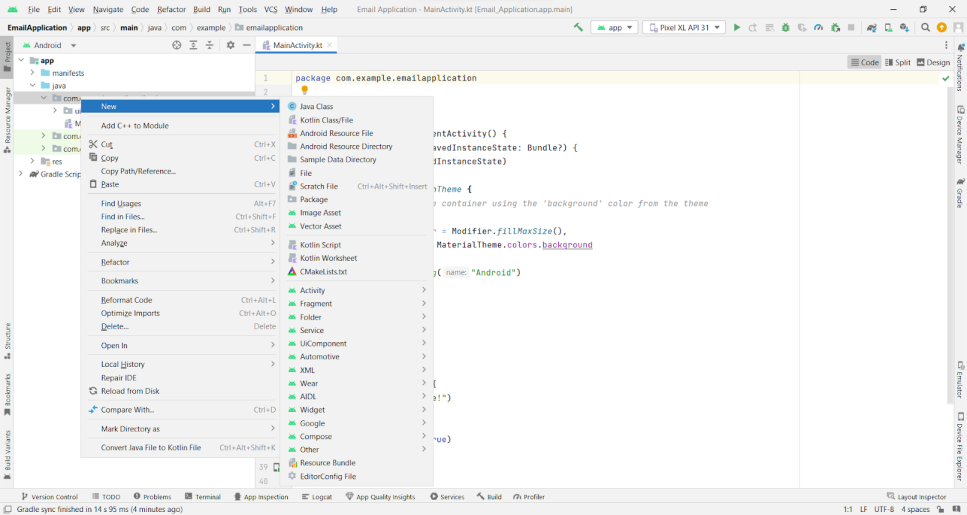


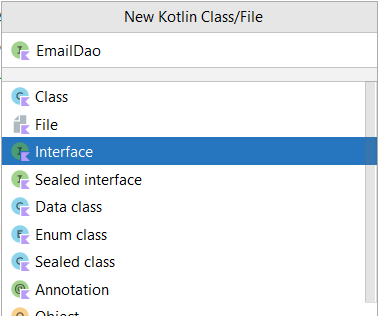
Email data class code:

<https://github.com/smartinternz02/AdaptiveEmailApp/tree/main/app/src/main/java/com/example/emailapplication/Email.kt>

### Create EmailDao interface

Create EmailDao interface



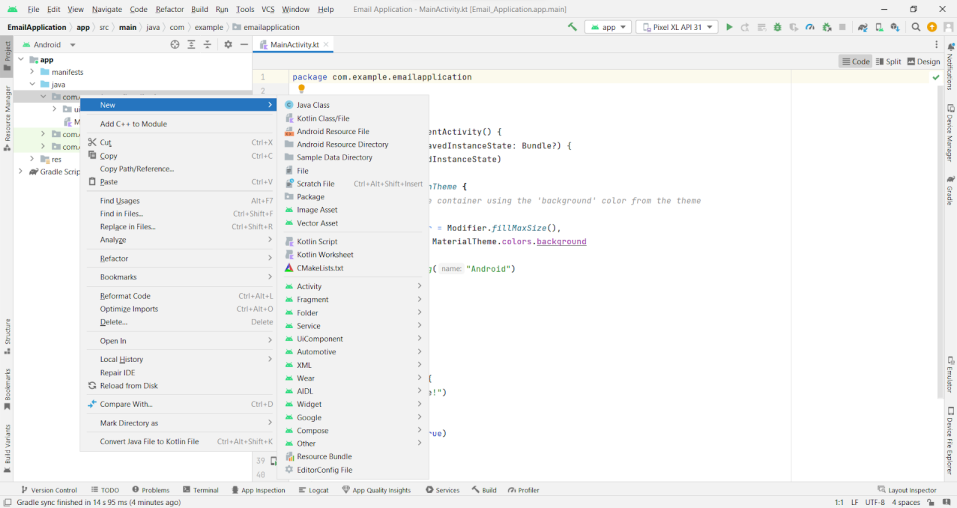


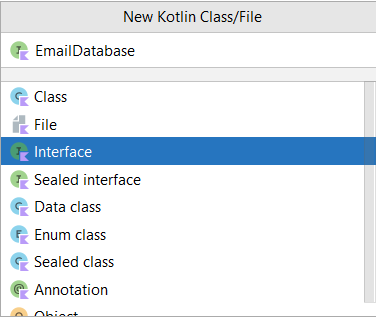
EmailDao interface code:

<https://github.com/smartinternz02/AdaptiveEmailApp/tree/main/app/src/main/java/com/example/emailapplication/EmailDao.kt>

### Create EmailDatabase class

Create EmailDatabase class





EmailDatabase class code:

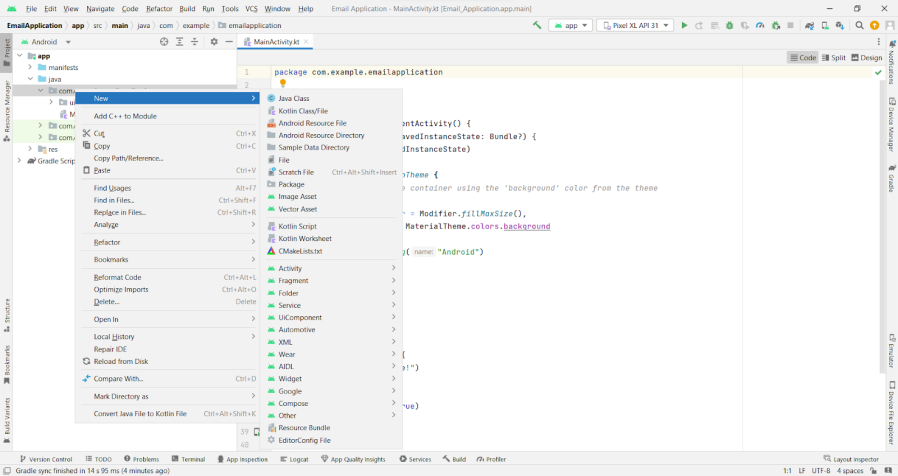
<https://github.com/smartinternz02/AdaptiveEmailApp/tree/main/app/src/main/java/com/example/emailapplication/EmailDatabase.kt>

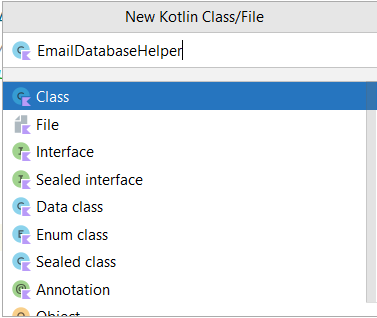
### BrainStroming:

### 

### Create EmailDatabaseHelper class

Create EmailDatabaseHelper class





EmailDatabaseHelper class code:

<https://github.com/smartinternz02/AdaptiveEmailApp/tree/main/app/src/main/java/com/example/emailapplication/UserDatabaseHelper.kt>

### Building application UI and connecting to database

This milestone explains about Building application UI and Connecting to database.

### Project Scope

To begin with any project, it is important to first identify the objectives of the project. What are the client’s requirements? What are the expectations? Members from both your organization and the client’s side need to collaborate to lay down certain things.

* **Identify Conditions of Satisfaction (CoS):** Stakeholders should identify the conditions of satisfaction. These include the project requirements or goals and the expected outcome. It is quite necessary to understand what you actually want to achieve or else you won’t know how to proceed. To avoid any miscommunication, have every stakeholder approve the CoS document before proceeding to the next phase.
* ***Outline Project Overview Statement (PoS)*:** The project overview statement is basically the outcome from the CoS. This document outlines the final and approved CoS that is agreed upon by all stakeholders. Project managers can refer to this document throughout the project to evaluate the efficacy. As an adaptive framework will be followed, the PoS should be adjusted with the changing CoS after approval.
* **Prioritize Requirements:** The project managers and stakeholders collaborate to decide the overall scope of the project. This is mainly about the order of tasks. The requirements are organized and PMs and analysts assign the priorities in a realistic manner. Careful consideration is necessary to avoid omitting any critical requirements. Try to evaluate the consequences of not meeting certain requirements for business and you can prioritize them in a better way.

### The Cycle Schedule

As the project scope is now clear, it’s time to break down the project into mini-projects or adaptive management cycle/iterations. Each cycle should be planned with the aim to provide one or more deliverables. To create the cycle schedule effectively, PMs can follow four simple steps:

* Define the tasks from the work breakdown structure.
* Group the tasks and assign them to team members.
* Create an effective work schedule.

Pay attention to your goal here, i.e. to define and plan the tasks your team will be working on. Individual tasks should be defined according to the priority set earlier. Set the deadlines while assigning tasks to the team members and make sure to carefully check all the task dependencies as that can hamper the efficiency.

### The Cycle Build

The actual work begins in this phase. The team members start working on the allotted tasks. As the team progresses, cycles can be adjusted. All the pending tasks that hadn’t been completed due to changes in some requirements move to the next cycle. The key components here are:

* Starting the work as decided.
* Monitoring the progress and adjusting the cycle build.
* Ending the current cycle at the scheduled completion time.
* Shifting the pending tasks to the next cycle.
* Recording the feedback/ideas for improvement.
* Tracking the problems.

The main difference to notice between the traditional and adaptive approach is that the timeline and schedule set here are fixed. If your team misses a deliverable on the deadline, the deliverable is set aside and prioritized accordingly in the upcoming cycles. All the problems faced here in this cycle are addressed in the next cycle. As for the improvement in the process, it is important to communicate clearly with the stakeholders.

### 4. The Client Checkpoint

The client checkout phase is the most essential part of the adaptive framework. It is time to get an evaluation of the deliverables or outcome of the cycle build phase. The client should thoroughly review the quality and provide feedback/suggestions.

Based on this analysis, the project manager can communicate with the client to address and schedule the adjustments or changes needed in the upcoming iteration. The course corrections are properly laid out to avoid the existing errors in later cycles.

From here on, the process keeps on repeating. The project manager and team members return to the cycle plan and then move on to cycle build and the client reviews the cycle. The iteration goes on until the project is completed.

### 5. The Final Report

At the end of the project, it is important to evaluate the success of the project. The project manager, team, and client can all collaborate and communicate to determine the success points and to talk about downfalls if any. Everything is documented and stakeholders share their experiences. This is all useful for future projects and the final report of the project can be referred to later to check the adaptive process or other critical points.

## Benefits of Adaptive Project Management

As we now know the different phases of Adaptive Project Management, it is quite clear to figure out the benefits of this framework. The benefits are for both, clients and project teams. Let’s outline the benefits:

* ***Focus on Client***: The main focus of the team members is to deliver the client’s needs according to the project scope that is documented after an efficient process.
* ***Client-driven***: The framework allows you to include the clients in the project at each stage. This simply means that the meaningful involvement of clients will let them know the progress of the project and changes can be made iteratively.
* ***Focus on Change***: Changes in the project are important for a better solution and deliverables. From the early stages of the project, the development team stays ready to implement changes for improved results.
* ***Prioritization***: As the project scope is about prioritizing the requirements, it eliminates the work that doesn’t add value to the project. This is beneficial for teams to focus on important tasks first and deliver the project on time.
* ***Continuous Feedback***: As there is open communication between the clients and team, everyone can have their say. Continuous feedback helps in driving the project forward in the right direction to deliver positive results.

## Think Like a Chef

To achieve success while working on the Adaptive Project Framework, it is important for you to understand a simple analogy: “think like a chef and not like a cook.” A cook usually follows the existing recipes that someone else wrote and finds themselves lost when there is some issue with the taste or an ingredient goes missing. On the other hand, a chef adapts and works with what’s available to deliver the same tasty food.

Similarly, all the projects are unique and a traditional approach to these projects might not be the right option every time. That is why adapting to the changing project requirements and priorities with the available resources is important.

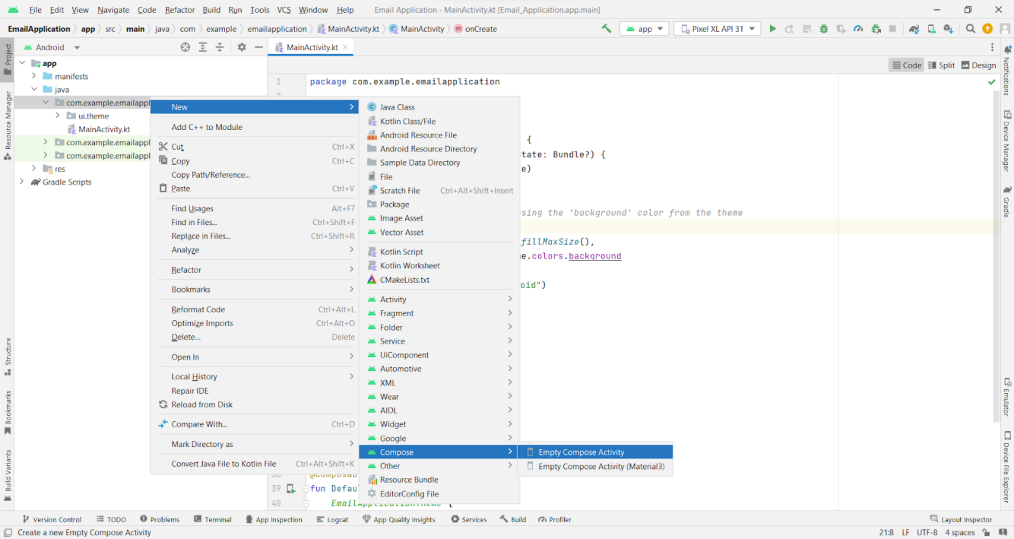
The adaptive framework is all about an agile approach and out-of-the-box thinking to creatively complete the projects.

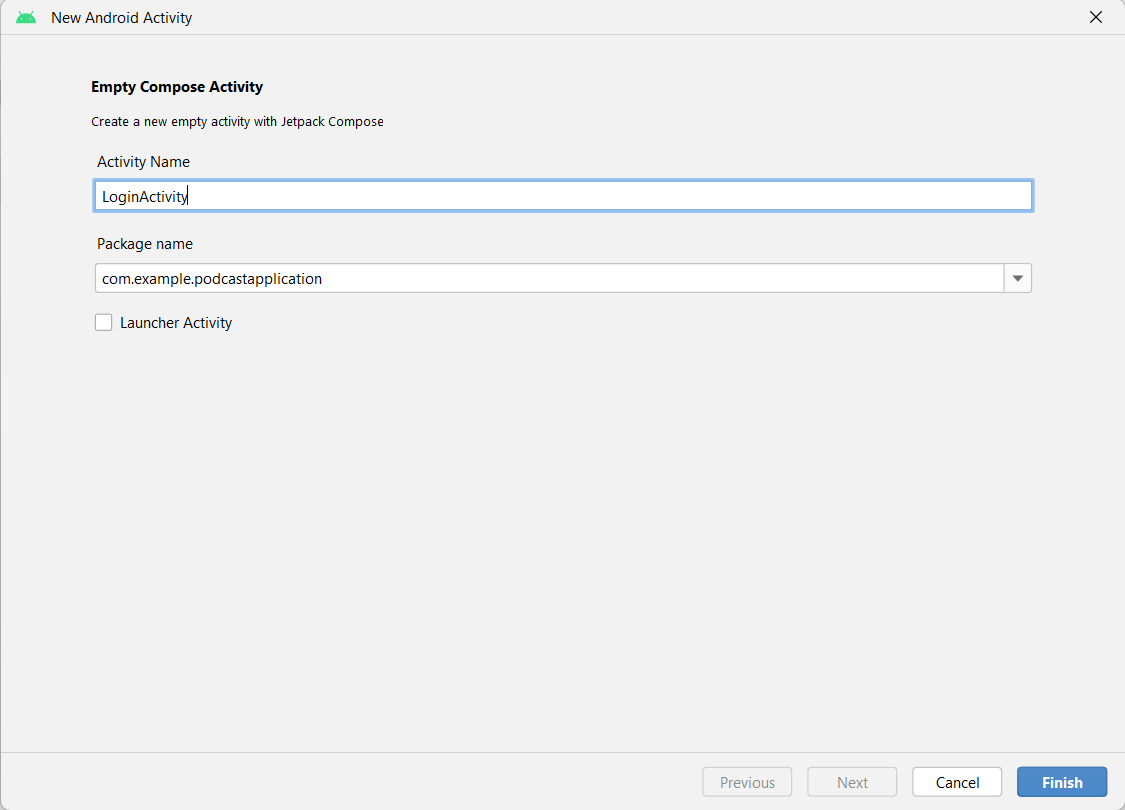
To assist you with adaptive planning and execution, project management software can get things done smartly and quickly [organize projects](https://www.proprofsproject.com/blog/10-tips-to-organize-projects-without-failing-to-meet-deadlines/).

The project managers and clients can easily track the progress of the project cycle and can collaborate with team members to provide feedback. So, combine APF with a smart project management tool and drive your projects to success.

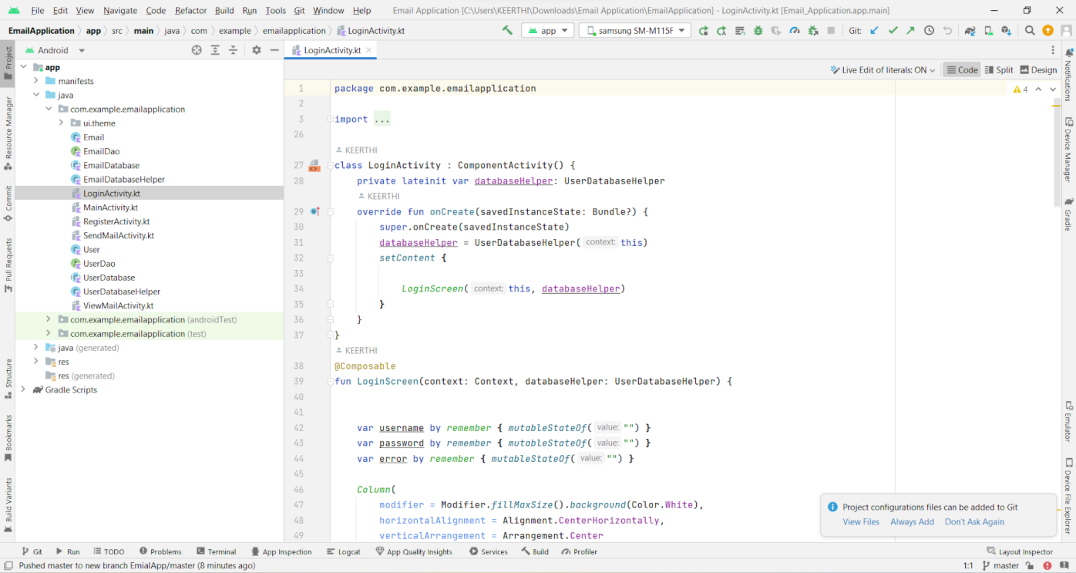
### Creating LoginActivity.kt with database

Creating LoginActivity.kt with database





Database connection in LoginActivity.kt



Complete code in below link:

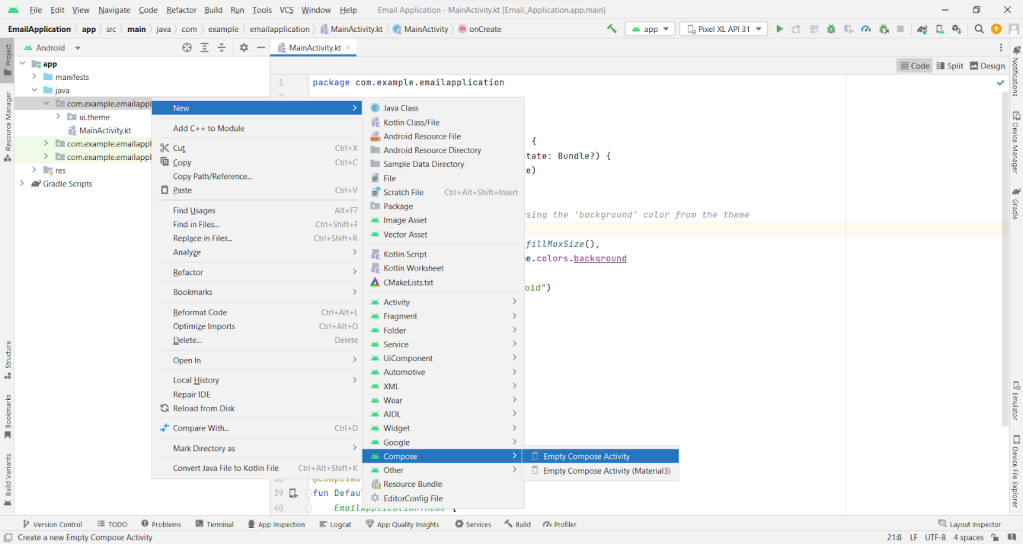
<https://github.com/smartinternz02/AdaptiveEmailApp/tree/main/app/src/main/java/com/example/emailapplication/LoginActivity.kt>

### Creating RegisterActivity.kt with database

Creating RegisterActivity.kt with database

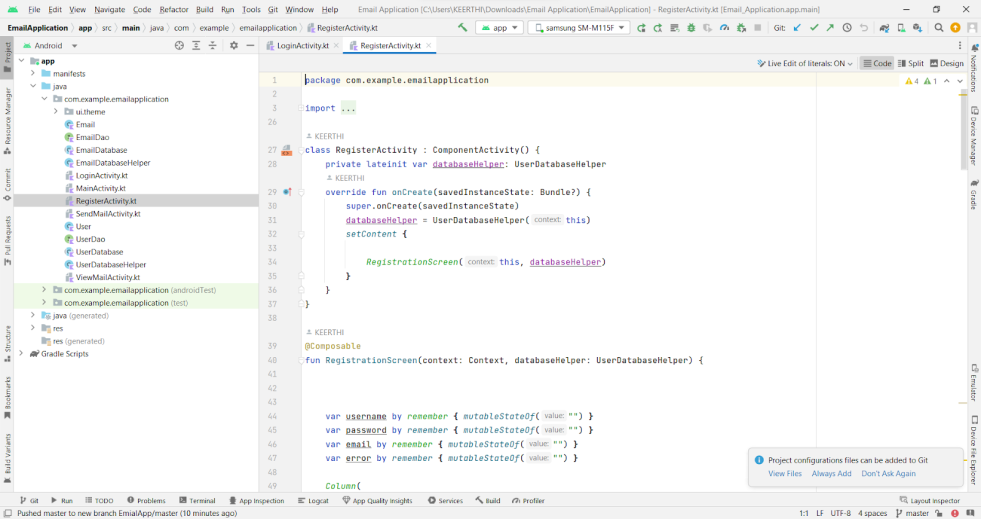








Database connection in RegisterActivity.kt



Complete code in below link:

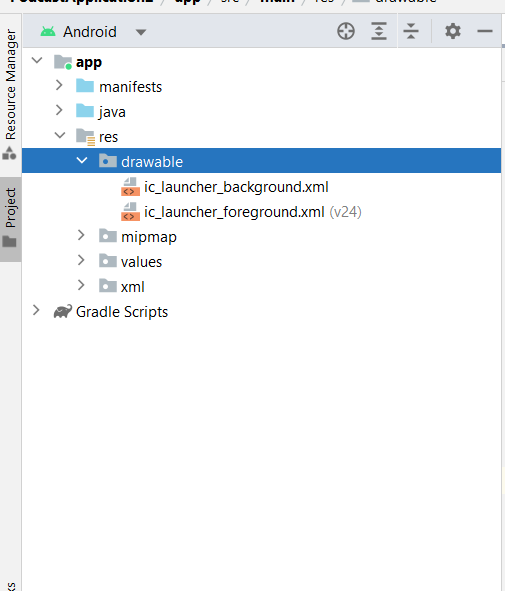
<https://github.com/smartinternz02/AdaptiveEmailApp/tree/main/app/src/main/java/com/example/emailapplication/RegisterActivity.kt>

**Creating MainActivity.kt file**

Creating MainActivity.kt file

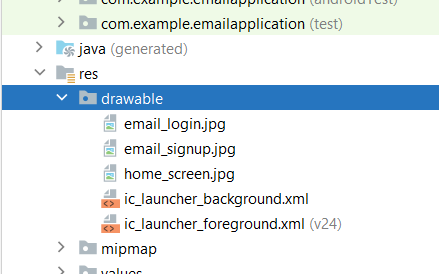
In MainActivity.kt file the main application is developed

* Before creating UI we need to add some images in drawables which are in res

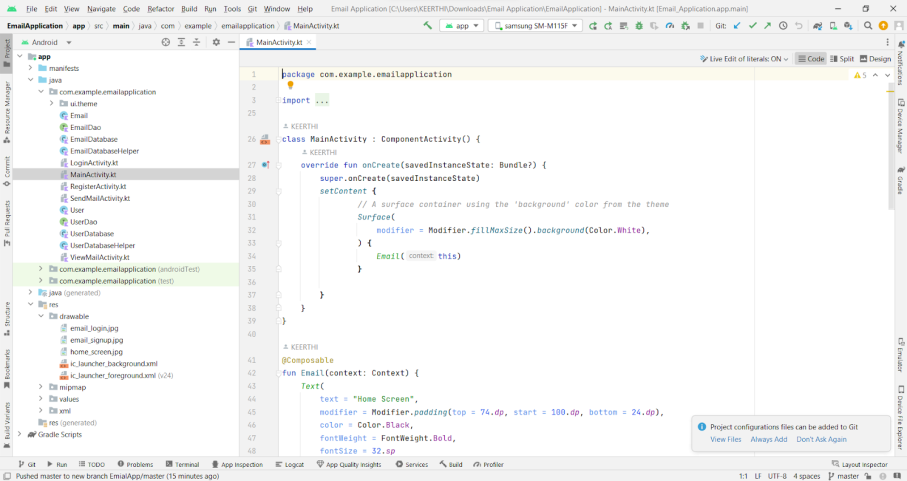


Download the required drawable from the code:

<https://github.com/smartinternz02/AdaptiveEmailApp/tree/master/app/src/main/res/drawable>



MainActivity.kt

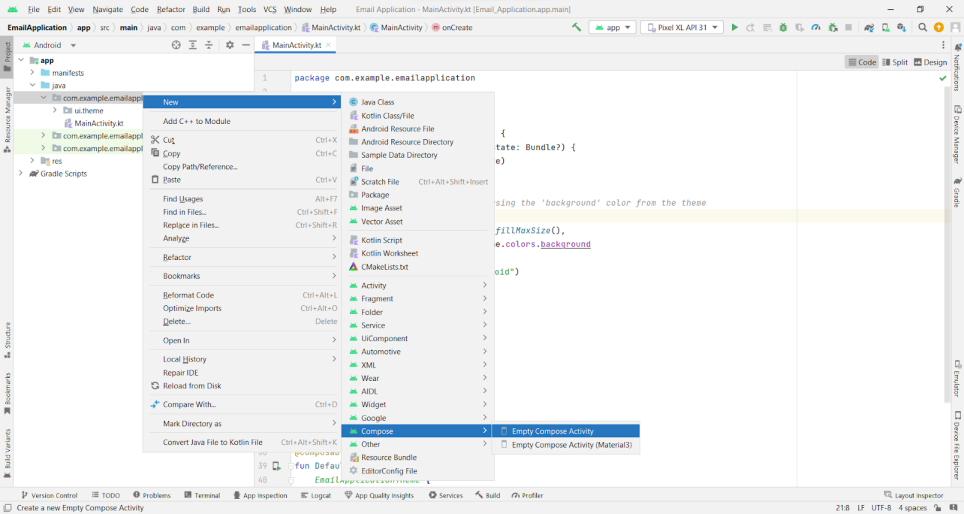


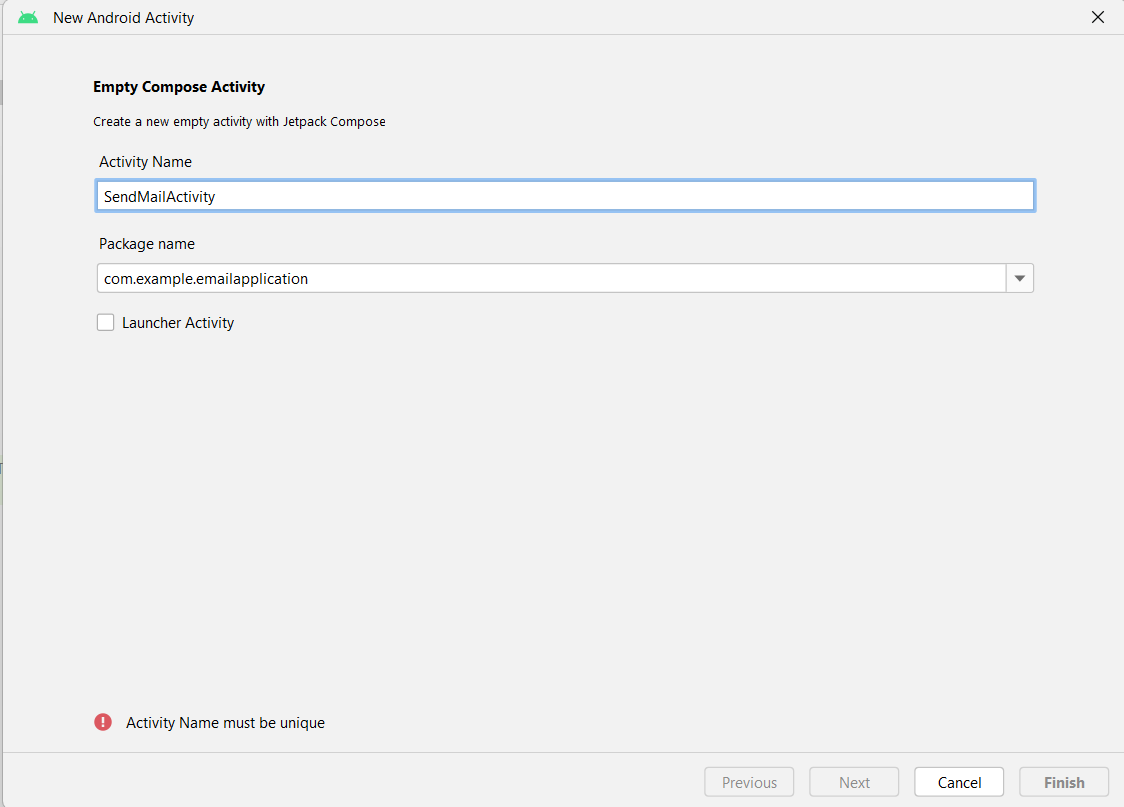
Complete code in below link:

<https://github.com/smartinternz02/AdaptiveEmailApp/tree/main/app/src/main/java/com/example/emailapplication/MainActivity.kt>

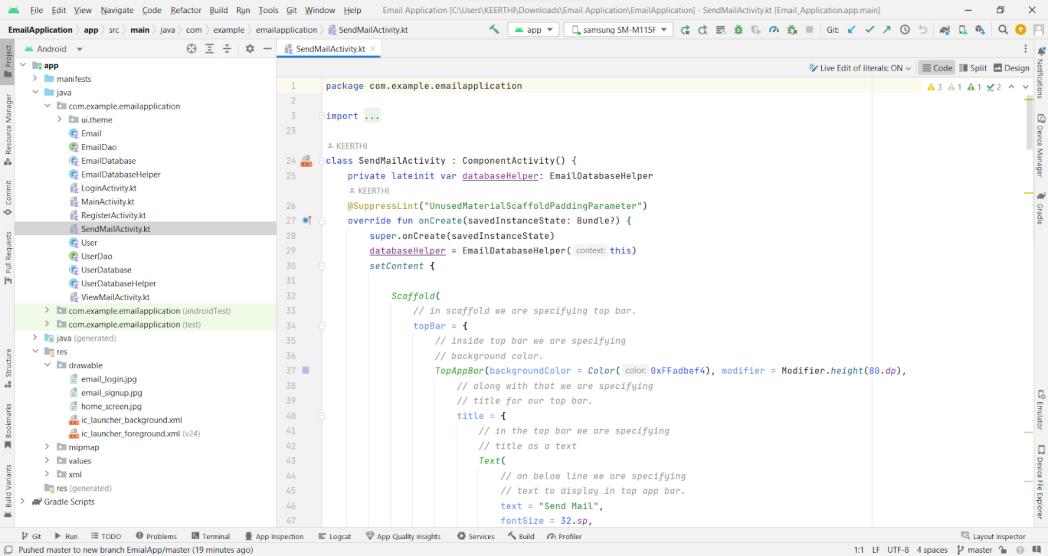
### Creating SendMailActivity.kt file

Creating SendMailActivity.kt file





SendMailActivity.kt

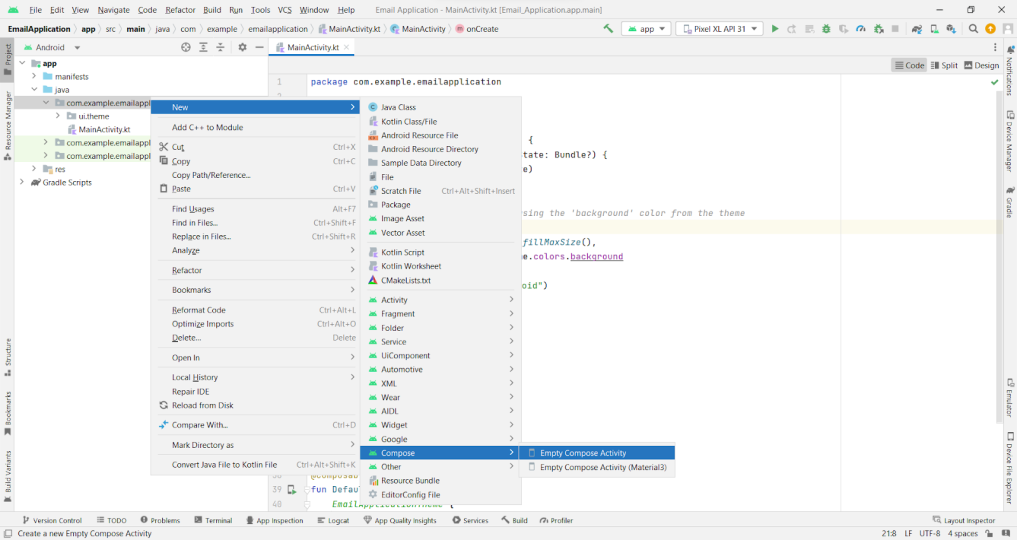


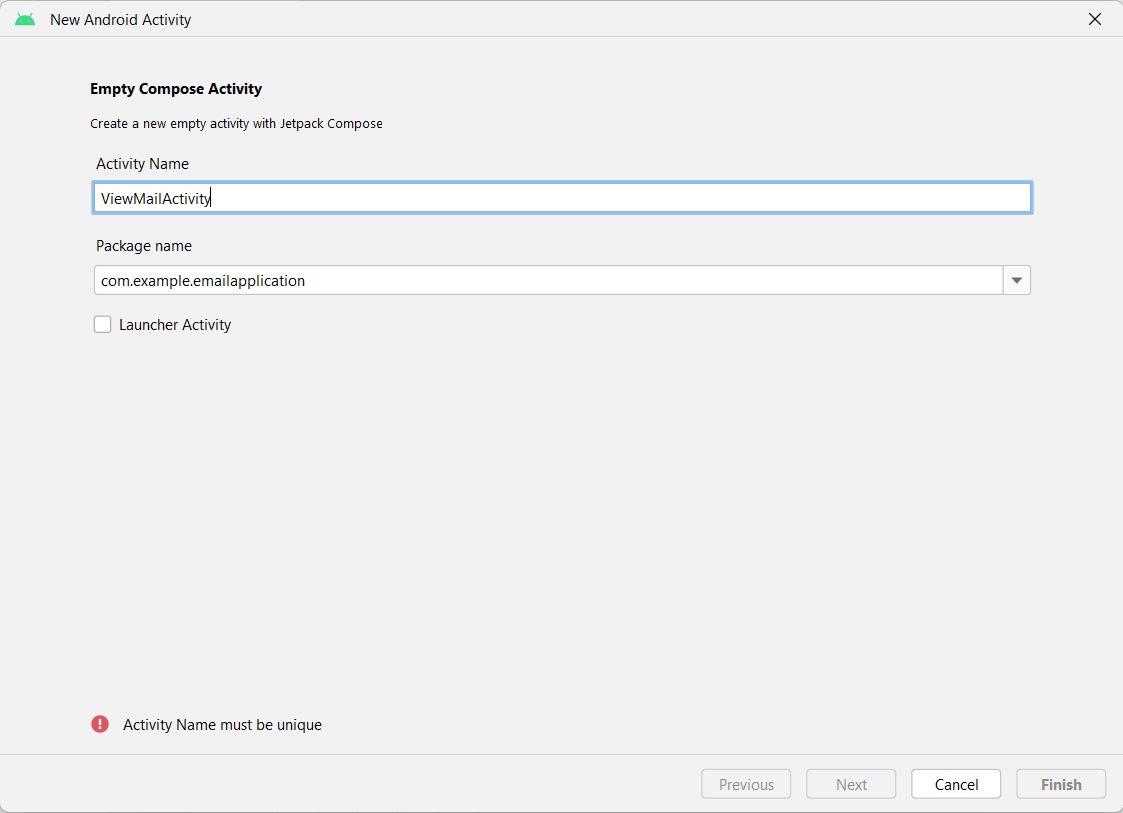
Complete code in below link:

<https://github.com/smartinternz02/AdaptiveEmailApp/tree/main/app/src/main/java/com/example/emailapplication/SendMailActivity.kt>

### Creating ViewMailActivity.kt file

Creating ViewMailActivity.kt file



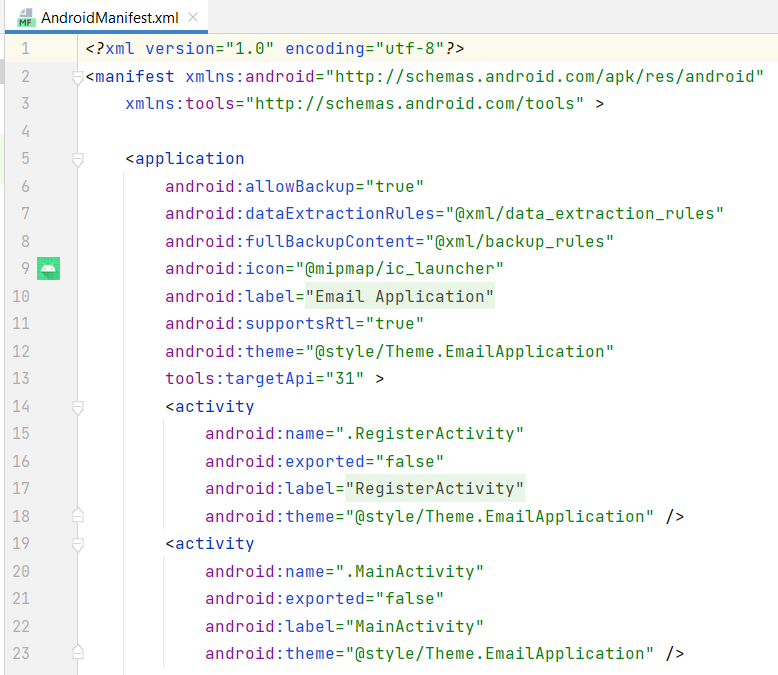


Complete code in below link:

<https://github.com/smartinternz02/AdaptiveEmailApp/tree/main/app/src/main/java/com/example/emailapplication/ViewMailActivity.kt>

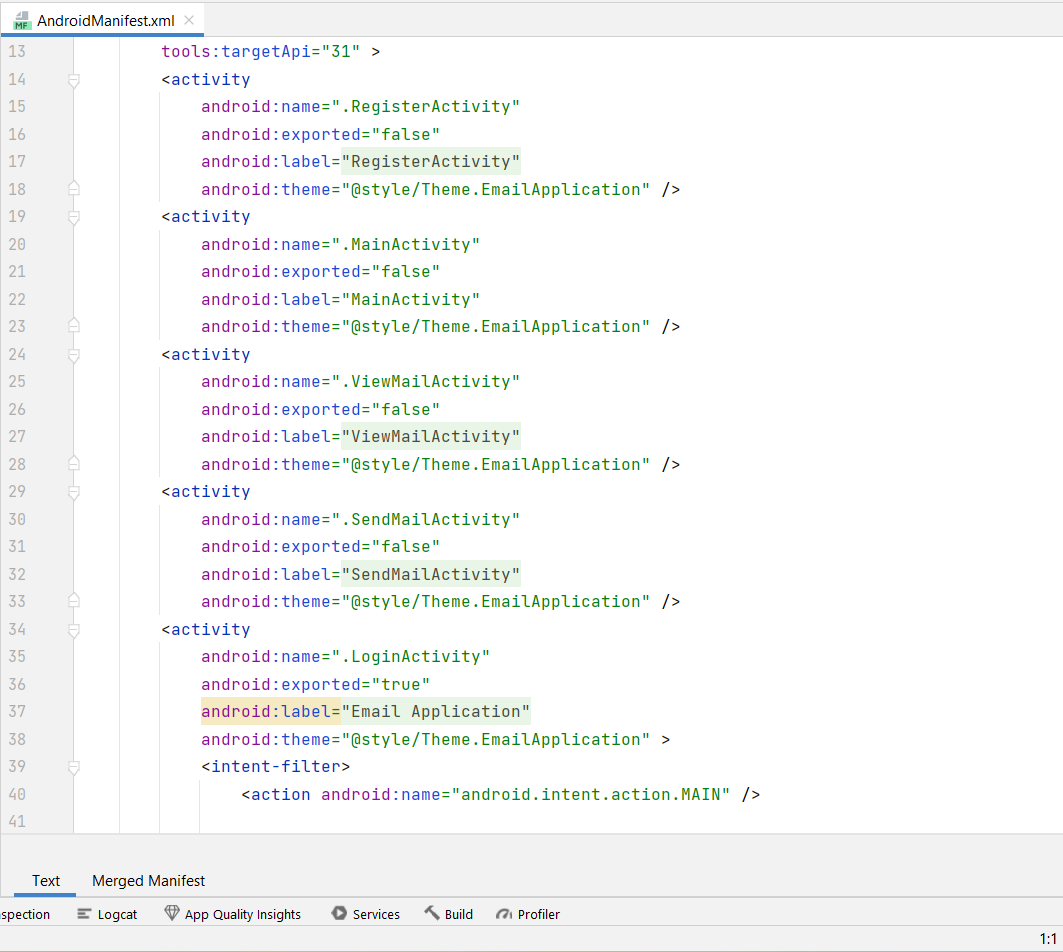
### Modifying AndroidManifest.xml

Modifying AndroidManifest.xml



When we run the app we will get the MainActivity.kt file as our first screen , but we want LoginActivity.kt , So we need to change in AndroidManifest.xml.

Changed AndroidManifest.xml



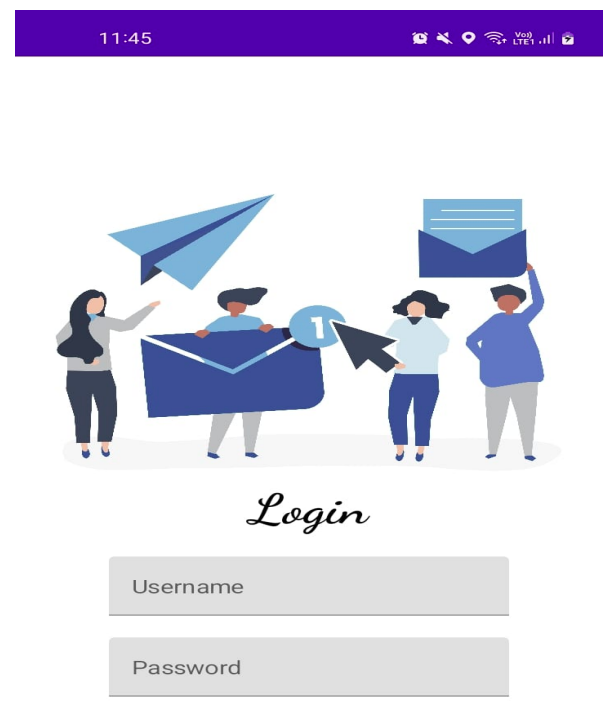
Complete AndroidManifest.xml code:

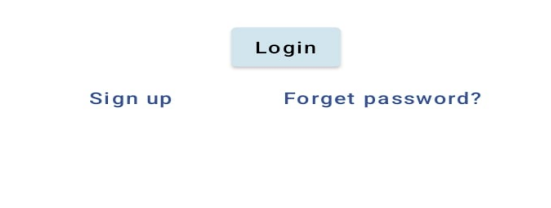
<https://github.com/smartinternz02/AdaptiveEmailApp/blob/master/app/src/main/AndroidManifest.xml>

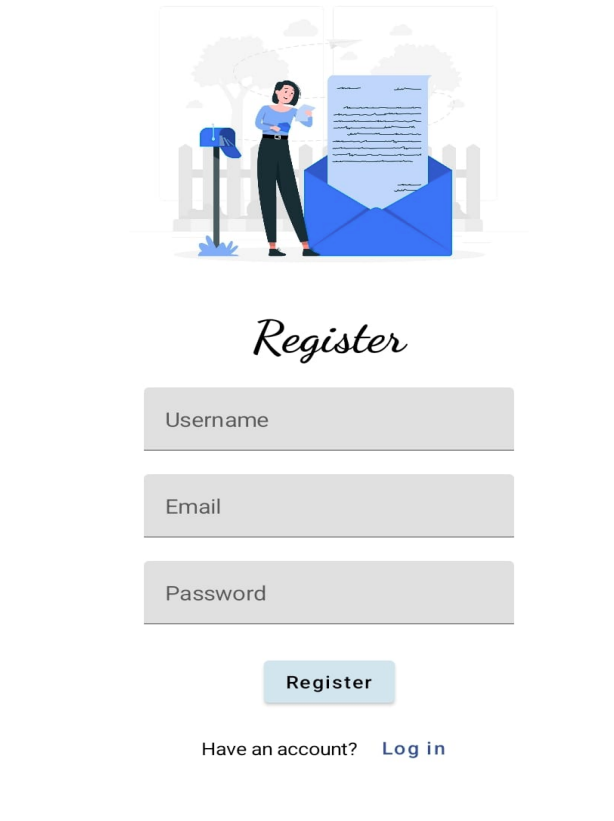
### Running the application

This milestone explains about running the application.

**RESULT**

****





\*\*\*THANK YOU\*\*\*