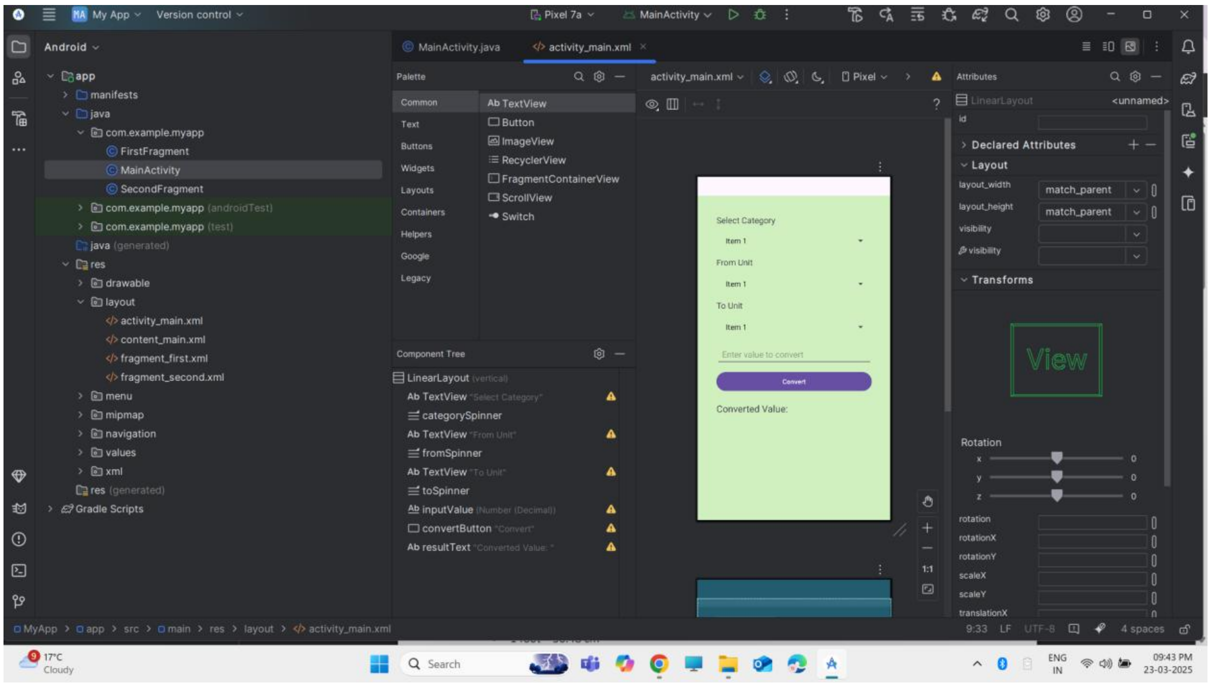
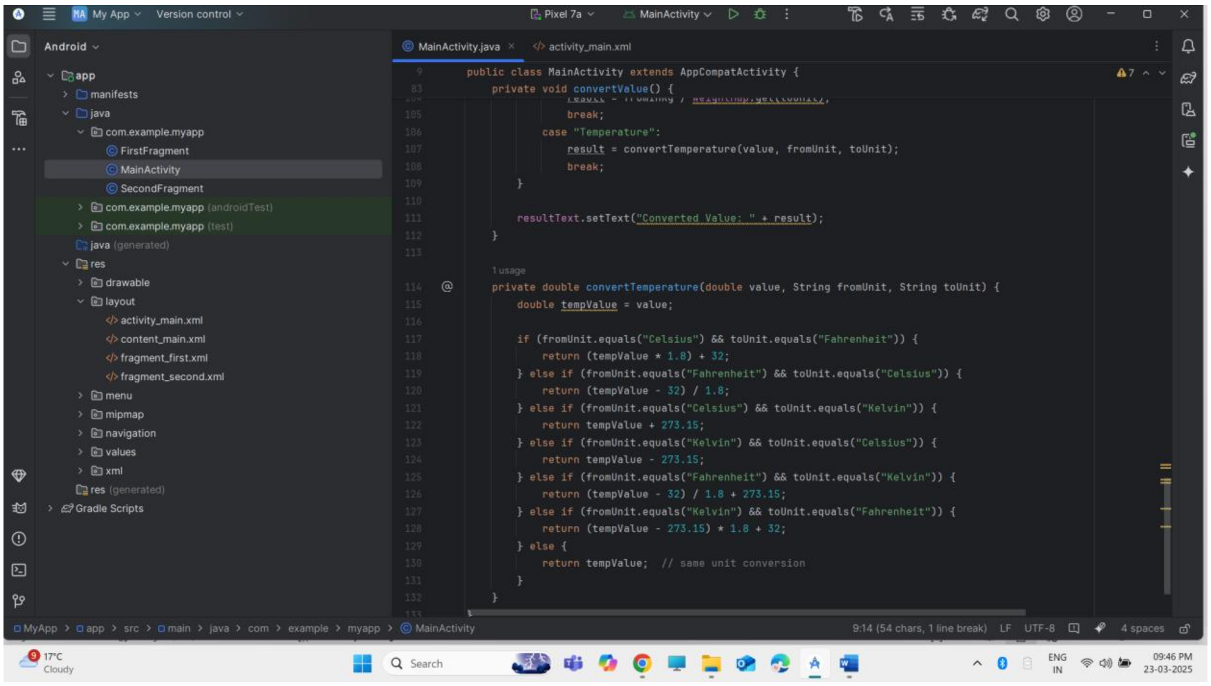
Git Repo: https://github.com/TOMINJOSE88/SIT708.git

Subtask 1: Design UI for the App



```
LinearLayout (vertical)
    Ab TextView "Select Category"
    categorySpinner
    Ab TextView "From Unit"
    fromSpinner
    Ab TextView "To Unit"
    toSpinner
    inputValue (Number (Decimal))
    convertButton "Convert"
    Ab resultText "Converted Value: "
```

Subtask 2: Implement the Conversion Logic



```java
public class MainActivity extends AppCompatActivity {
    private void convertValue() {
                break;
            case "Temperature":
                result = convertTemperature(value, fromUnit, toUnit);
                break;
        }

        resultText.setText("Converted Value: " + result);
    }

    1 usage
    private double convertTemperature(double value, String fromUnit, String toUnit) {
        double tempValue = value;

        if (fromUnit.equals("Celsius") && toUnit.equals("Fahrenheit")) {
            return (tempValue * 1.8) + 32;
        } else if (fromUnit.equals("Fahrenheit") && toUnit.equals("Celsius")) {
            return (tempValue - 32) / 1.8;
        } else if (fromUnit.equals("Celsius") && toUnit.equals("Kelvin")) {
            return tempValue + 273.15;
        } else if (fromUnit.equals("Kelvin") && toUnit.equals("Celsius")) {
            return tempValue - 273.15;
        } else if (fromUnit.equals("Fahrenheit") && toUnit.equals("Kelvin")) {
            return (tempValue - 32) / 1.8 + 273.15;
        } else if (fromUnit.equals("Kelvin") && toUnit.equals("Fahrenheit")) {
            return (tempValue - 273.15) * 1.8 + 32;
        } else {
            return tempValue;  // same unit conversion
        }
    }
```

Subtask 3: Research on Llama2

Llama 2 and Its Use in Android Mobile Applications

Llama 2 is a state-of-the-art large language model (LLM) developed by Meta (formerly Facebook). It is a successor to the original Llama model, offering better performance, larger training data, and a more advanced architecture. Llama 2 is open weight, meaning developers and researchers can use and adapt it for various applications, including mobile environments. The model comes in different sizes (7B, 13B, and 70B parameters), allowing flexibility for deployment based on resource availability. While larger models might require server-side processing, the smaller versions of Llama 2 can be integrated with edge devices and optimized for mobile usage.

The power of Llama 2 lies in its ability to understand and generate human-like text, follow instructions, summarize information, translate languages, and even answer questions with remarkable accuracy. Android app developers can harness this technology to bring intelligent features to mobile applications, improving user engagement and automation capabilities.

Below are five ideas on how Llama 2 can be used in Android mobile apps:

1. **Smart Virtual Assistant**
   Llama 2 can be used to build advanced virtual assistants inside mobile apps. Unlike traditional assistants that rely on pre-programmed commands, an LLM-based assistant can understand context, answer open-ended questions, schedule tasks, and even hold natural conversations. Such assistants can make apps more interactive and useful for productivity, customer support, and daily organization.

2. **Language Translation and Learning Apps**
   Mobile apps that focus on language learning or translation services can benefit greatly from Llama 2's multilingual abilities. The model can translate text between multiple languages, explain grammar rules, and provide usage examples in real-time. This can improve the quality of translation apps or educational tools, helping users learn new languages more effectively.

3. **Content Generation Tools**
   Llama 2 can power content creation apps that generate blog posts, product descriptions, captions for social media, or even poetry. Writers, marketers, and students can use mobile apps with integrated Llama 2 to brainstorm ideas, rewrite text, or create drafts quickly, saving time and effort.

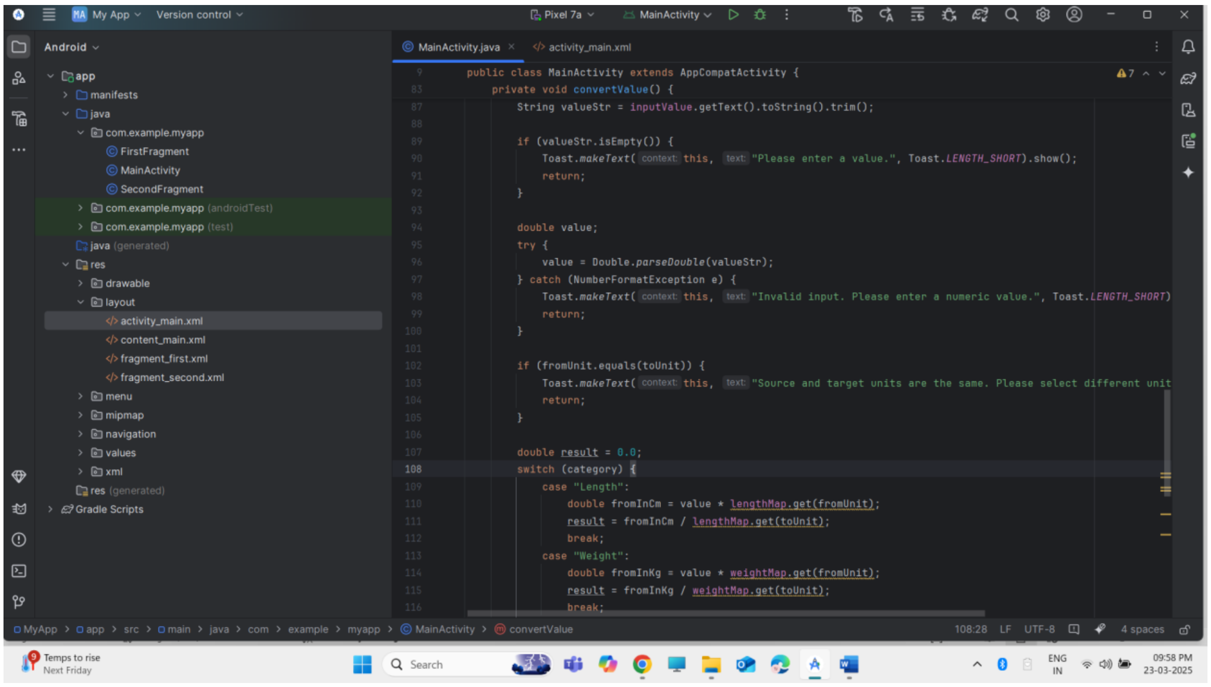
4. **Personalized Chatbots for E-commerce Apps**
   E-commerce apps can use Llama 2 to create smarter chatbots that assist

customers in finding products, answering questions, and providing recommendations. Unlike rule-based bots, a chatbot powered by Llama 2 can understand user intent better, handle complex queries, and offer a more human-like shopping assistant experience.

5. **Code Assistant for Mobile Developers**
   Another innovative use case is creating a coding assistant app for developers. With Llama 2's ability to generate and explain code snippets, Android developers can receive code suggestions, explanations, and debugging help directly on their phones. This can help both beginners and experienced programmers save time while learning or working on the go.

In conclusion, Llama 2 is an exciting tool for app developers looking to integrate natural language intelligence into mobile applications. Whether it's building conversational interfaces, automating content creation, or offering personalized experiences, Llama 2 has the potential to transform how users interact with Android apps. With further optimization, it could become a key part of AI-driven mobile experiences soon.

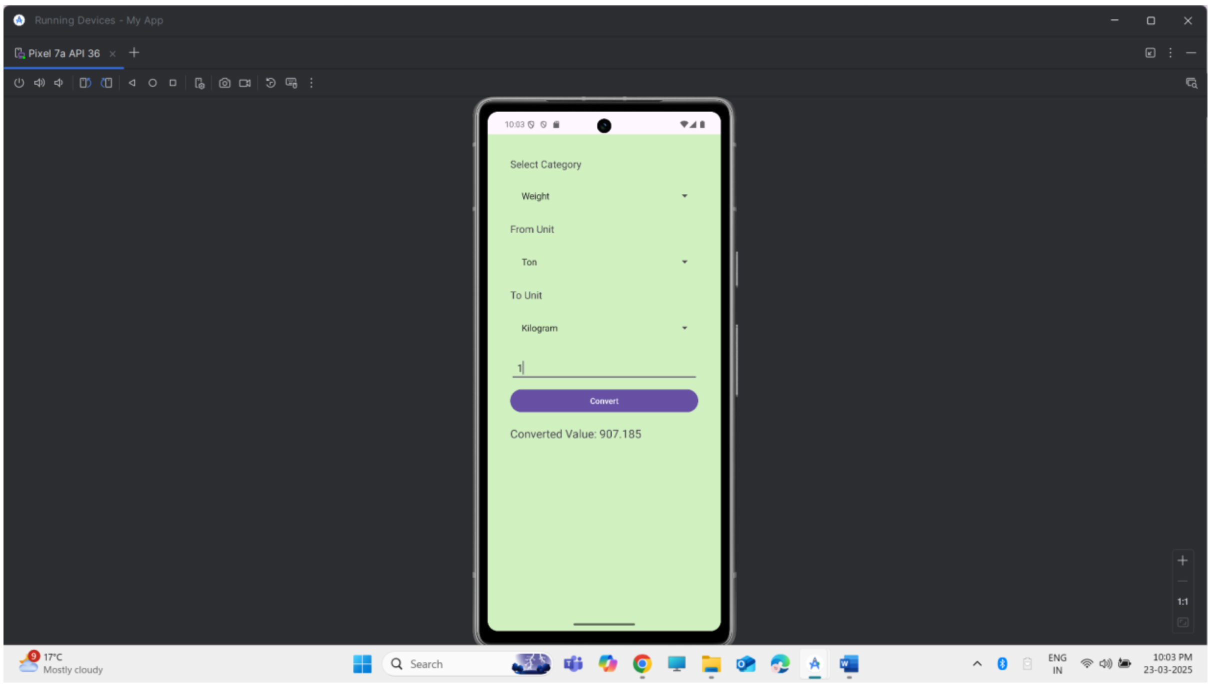
Subtask 4: Add Validation and Error Handling



```java
public class MainActivity extends AppCompatActivity {
    private void convertValue() {
        String valueStr = inputValue.getText().toString().trim();

        if (valueStr.isEmpty()) {
            Toast.makeText( context: this,  text: "Please enter a value.", Toast.LENGTH_SHORT).show();
            return;
        }

        double value;
        try {
            value = Double.parseDouble(valueStr);
        } catch (NumberFormatException e) {
            Toast.makeText( context: this,  text: "Invalid input. Please enter a numeric value.", Toast.LENGTH_SHORT)
            return;
        }

        if (fromUnit.equals(toUnit)) {
            Toast.makeText( context: this,  text: "Source and target units are the same. Please select different unit
            return;
        }

        double result = 0.0;
        switch (category) {
            case "Length":
                double fromInCm = value * lengthMap.get(fromUnit);
                result = fromInCm / lengthMap.get(toUnit);
                break;
            case "Weight":
                double fromInKg = value * weightMap.get(fromUnit);
                result = fromInKg / weightMap.get(toUnit);
                break;
```

validation and error handling were added to the convertValue() method to make the app more user-friendly and robust.

App Screen:



Select Category

Weight

From Unit

Ton

To Unit

Kilogram

1

Convert

Converted Value: 907.185