Unit Conversion App

Mohammad Sargazi

Colorado State University Global

CSC475: Platform-Based Development

Dr. Bari

Nov 3rd, 2024

1- Pseudocode

```
1. User inputs a value.
2. User selects 'from' and 'to' units.
3. On clicking convert:
- Use Converter class to compute the conversion.
- Display the converted result.
4. Converter class:
    - Takes value, 'from' unit, and 'to' unit.
- Applies appropriate conversion formula.
5. Display error if unsupported conversion.
6. Unit tests validate each conversion function.
```

2- Purpose and challenges

The purpose of this project was to create a user-friendly Android app that converts between various units, such as temperature, length, and weight. With options like Celsius to Fahrenheit or meters to feet, this app provides practical utility by allowing users to perform essential conversions directly from their mobile devices. The simplicity of the app's design was intentional, aiming to make it straightforward for users of all technical backgrounds to navigate and use without confusion. Building this app allowed me to explore Android development concepts in a real-world context, bridging coding with practical application.

During development, one of the main obstacles was ensuring the app could handle incompatible unit conversions gracefully, such as when a user mistakenly tried to convert temperature units (e.g., Fahrenheit) into length units (e.g., meters). Without proper error handling, these attempts would cause the app to crash. To solve this, I implemented a try-catch block to detect unsupported conversions and display a helpful error message, making the app more robust and user-friendly. Additionally, I had to be mindful of the latest Android SDK requirements, as dependencies required a higher SDK level than initially configured. This led to adjustments in the project's Gradle files, which taught me about managing dependencies and configuring SDK levels correctly.

Overall, this project helped me acquire several key skills in Android development, including creating user interfaces with Compose, handling user input, and implementing error handling for a smoother user experience. The experience deepened my understanding of Kotlin programming, Compose UI elements, and Android's testing framework. Testing the conversion functions and ensuring they returned accurate results was both challenging and rewarding, as it demonstrated the importance of rigorous testing in software development. Below are screenshots showing the app's interface and successful conversions, highlighting the project's usability and functionality.

3- SourceFiles

3.1- Converter.kt

```kotlin
package com.example.unitconverter

object Converter {

    /**
     * Converts a given value from one unit to another.
     *
     * @param value The numerical value to convert.
     * @param fromUnit The unit to convert from .
     * @param toUnit The unit to convert to.
     * @return The converted value as a Double.
     */
    fun convert(value: Double, fromUnit: String, toUnit: String): Double {
        return when (fromUnit to toUnit) {
            // Temperature conversions
            "Celsius" to "Fahrenheit" -> (value * 9 / 5) + 32
            "Fahrenheit" to "Celsius" -> (value - 32) * 5 / 9

            // Length conversions
            "Meters" to "Feet" -> value * 3.28084
            "Feet" to "Meters" -> value / 3.28084

            // Weight conversions
            "Kilograms" to "Pounds" -> value * 2.20462
            "Pounds" to "Kilograms" -> value / 2.20462

            // If conversion is unsupported, throw an error
            else -> throw IllegalArgumentException("Unsupported conversion
from $fromUnit to $toUnit")
        }
    }
}
```

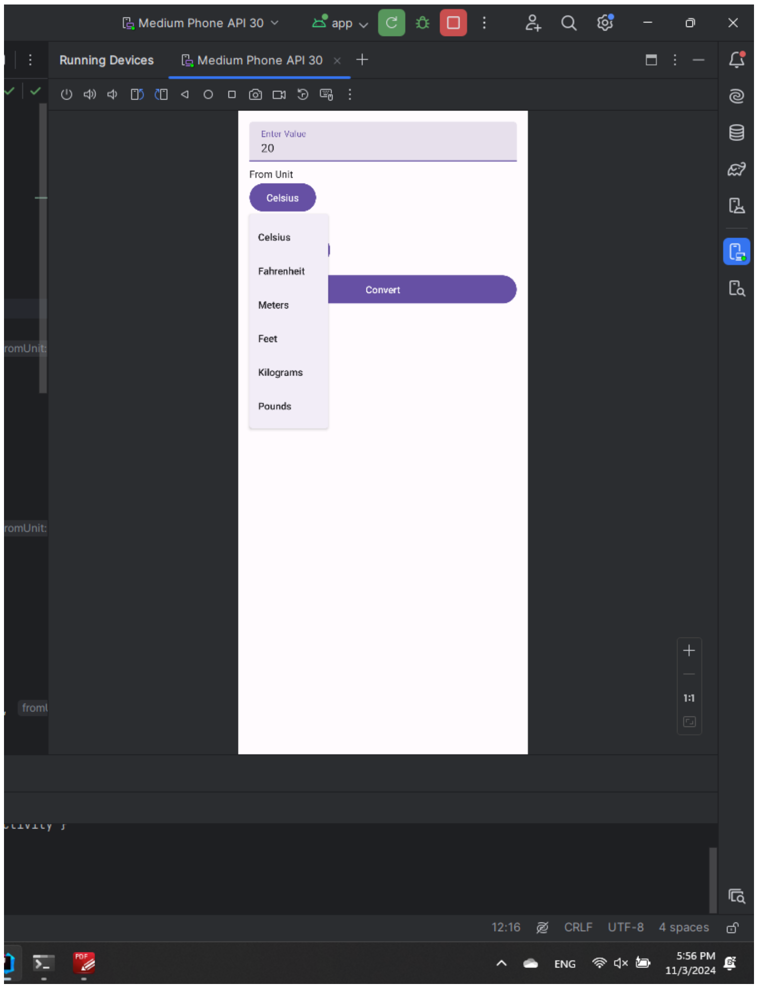
3.2- Mainactivity.kt

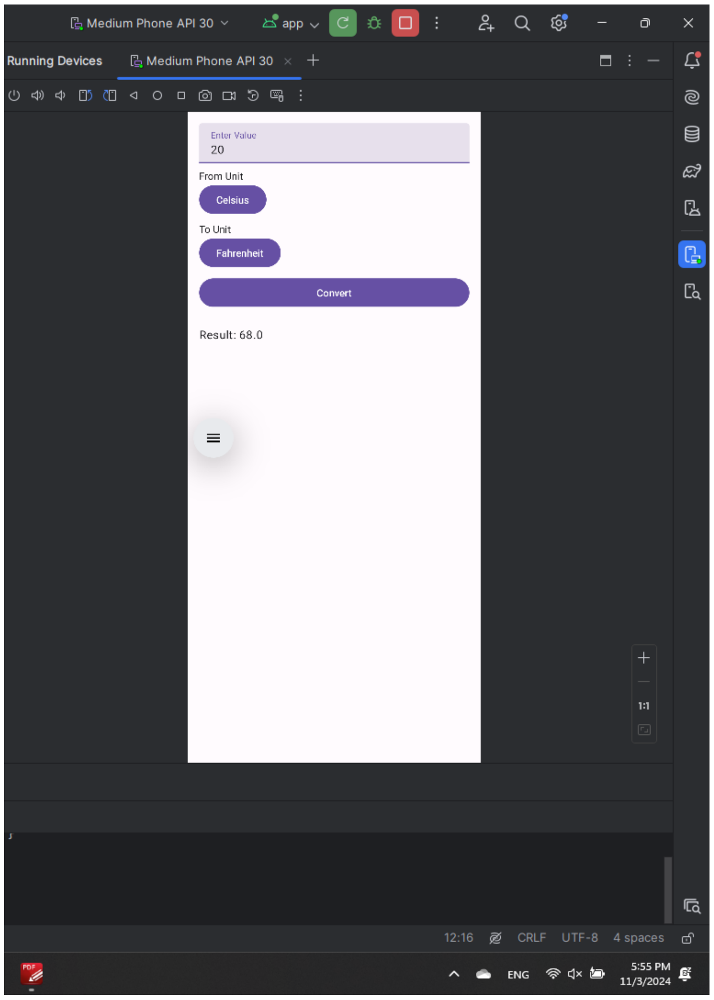
```kotlin
package com.example.unitconverter

import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.compose.foundation.layout.*
import androidx.compose.material3.*
import androidx.compose.runtime.*
import androidx.compose.ui.Modifier
import androidx.compose.ui.tooling.preview.Preview
import androidx.compose.ui.unit.dp
import com.example.unitconverter.ui.theme.UnitConverterTheme

class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            UnitConverterTheme {
                Surface(modifier = Modifier.fillMaxSize(), color =
MaterialTheme.colorScheme.background) {
                    ConversionScreen()
                }
            }
        }
    }
}

@Composable
fun ConversionScreen() {
    var input by remember { mutableStateOf("") }
    var fromUnit by remember { mutableStateOf("Celsius") }
    var toUnit by remember { mutableStateOf("Fahrenheit") }
    var result by remember { mutableStateOf("") }
    var errorMessage by remember { mutableStateOf("") }  // Track error
messages

    Column(
        modifier = Modifier
            .fillMaxSize()
            .padding(16.dp),
        verticalArrangement = Arrangement.spacedBy(8.dp)
    ) {
        TextField(
            value = input,
            onValueChange = { input = it },
            label = { Text("Enter Value") },
            modifier = Modifier.fillMaxWidth()
        )

        UnitDropdown(selectedUnit = fromUnit, onUnitSelected = { fromUnit =
it }, label = "From Unit")
        UnitDropdown(selectedUnit = toUnit, onUnitSelected = { toUnit = it },
label = "To Unit")

        Button(
            onClick = {
                // Reset error message
                errorMessage = ""
                result = ""

                val inputValue = input.toDoubleOrNull()
                if (inputValue != null) {
                    try {
                        // Attempt the conversion
                        result = Converter.convert(inputValue, fromUnit,
toUnit).toString()
                    } catch (e: IllegalArgumentException) {
                        // Display error message for unsupported conversions
                        errorMessage = e.message ?: "Conversion error"
                    }
                } else {
                    errorMessage = "Invalid input"
                }
            },
            modifier = Modifier.fillMaxWidth()
        ) {
            Text("Convert")
        }

        // Display result or error message based on the outcome
        if (errorMessage.isNotEmpty()) {
            Text(
                text = "Error: $errorMessage",
                color = MaterialTheme.colorScheme.error,
                style = MaterialTheme.typography.bodyLarge,
                modifier = Modifier.padding(top = 16.dp)
            )
        } else {
            Text(
                text = "Result: $result",
                style = MaterialTheme.typography.bodyLarge,
                modifier = Modifier.padding(top = 16.dp)
            )
        }
    }
}

@Composable
fun UnitDropdown(selectedUnit: String, onUnitSelected: (String) -> Unit,
label: String) {
    val units = listOf("Celsius", "Fahrenheit", "Meters", "Feet",
"Kilograms", "Pounds")
    var expanded by remember { mutableStateOf(false) }

    Column {
        Text(label, style = MaterialTheme.typography.bodyMedium)
        Button(onClick = { expanded = true }) {
            Text(selectedUnit)
        }
        DropdownMenu(
            expanded = expanded,
            onDismissRequest = { expanded = false }
        ) {
            units.forEach { unit ->
                DropdownMenuItem(
                    text = { Text(unit) },
                    onClick = {
                        onUnitSelected(unit)
                        expanded = false
                    }
                )
            }
        }
    }
}

@Preview(showBackground = true)
@Composable
fun ConversionScreenPreview() {
    UnitConverterTheme {
        ConversionScreen()
    }
}
```

3.3- ConverterTest.kt

```
package com.example.unitconverter

import org.junit.Test
import org.junit.Assert.assertEquals

class ConverterTest {

    /**
     * Test Celsius to Fahrenheit conversion.
     */
    @Test
    fun testCelsiusToFahrenheit() {
        // 0°C should be 32°F
        assertEquals(32.0, Converter.convert(0.0, "Celsius", "Fahrenheit"),
0.001)
    }

    /**
     * Test Fahrenheit to Celsius conversion.
     */
    @Test
    fun testFahrenheitToCelsius() {
        // 32°F should be 0°C
        assertEquals(0.0, Converter.convert(32.0, "Fahrenheit", "Celsius"),
0.001)
    }

    /**
     * Test Meters to Feet conversion.
     */
    @Test
    fun testMetersToFeet() {
        // 1 meter should be approximately 3.28084 feet
        assertEquals(3.28084, Converter.convert(1.0, "Meters", "Feet"),
0.001)
    }

    /**
     * Test Feet to Meters conversion.
     */
    @Test
    fun testFeetToMeters() {
        // 3.28084 feet should be approximately 1 meter
        assertEquals(1.0, Converter.convert(3.28084, "Feet", "Meters"),
0.001)
    }

    /**
     * Test Kilograms to Pounds conversion.
     */
    @Test
    fun testKilogramsToPounds() {
        // 1 kilogram should be approximately 2.20462 pounds
        assertEquals(2.20462, Converter.convert(1.0, "Kilograms", "Pounds"),
0.001)
    }

    /**
     * Test Pounds to Kilograms conversion.
     */
    @Test
    fun testPoundsToKilograms() {
        // 2.20462 pounds should be approximately 1 kilogram
        assertEquals(1.0, Converter.convert(2.20462, "Pounds", "Kilograms"),
0.001)
    }

    /**
     * Test for unsupported conversion case.
     */
    @Test(expected = IllegalArgumentException::class)
    fun testUnsupportedConversion() {
        // Attempting an unsupported conversion should throw an error
        Converter.convert(1.0, "Liters", "Gallons")
    }
}
```

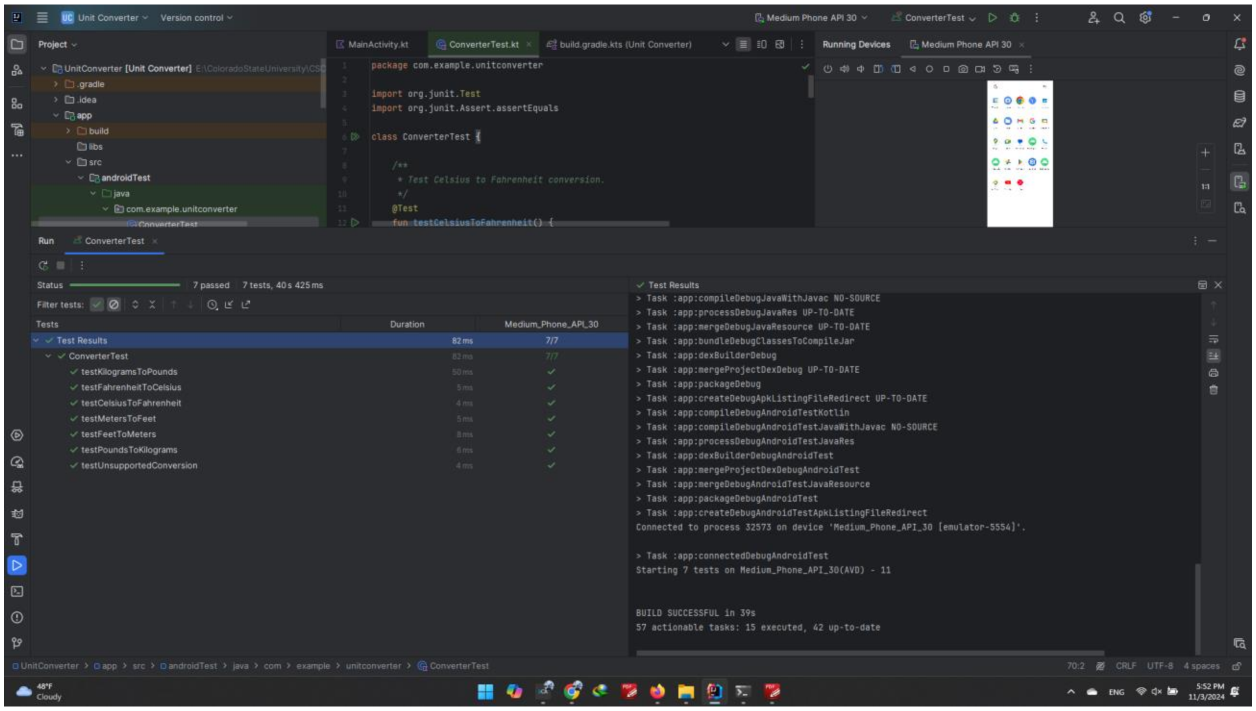
4- App Screenshots:





Result: 68.0

5- Test screenshots



```kotlin
package com.example.unitconverter

import org.junit.Test
import org.junit.Assert.assertEquals

class ConverterTest {

    /**
     * Test Celsius to Fahrenheit conversion.
     */
    @Test
    fun testCelsiusToFahrenheit() {
```

Status: 7 passed | 7 tests, 40 s 425 ms

Tests	Duration	Medium_Phone_API_30
Test Results	82 ms	7/7
ConverterTest	82 ms	7/7
testKilogramsToPounds	50 ms	✓
testFahrenheitToCelsius	5 ms	✓
testCelsiusToFahrenheit	4 ms	✓
testMetersToFeet	5 ms	✓
testFeetToMeters	8 ms	✓
testPoundsToKilograms	6 ms	✓
testUnsupportedConversion	4 ms	✓

```
> Task :app:compileDebugJavaWithJavac NO-SOURCE
> Task :app:processDebugJavaRes UP-TO-DATE
> Task :app:mergeDebugJavaResource UP-TO-DATE
> Task :app:bundleDebugClassesToCompileJar
> Task :app:dexBuilderDebug
> Task :app:mergeProjectDexDebug UP-TO-DATE
> Task :app:packageDebug
> Task :app:createDebugApkListingFileRedirect UP-TO-DATE
> Task :app:compileDebugAndroidTestKotlin
> Task :app:compileDebugAndroidTestJavaWithJavac NO-SOURCE
> Task :app:processDebugAndroidTestJavaRes
> Task :app:dexBuilderDebugAndroidTest
> Task :app:mergeProjectDexDebugAndroidTest
> Task :app:mergeDebugAndroidTestJavaResource
> Task :app:packageDebugAndroidTest
> Task :app:createDebugAndroidTestApkListingFileRedirect
Connected to process 32573 on device 'Medium_Phone_API_30 [emulator-5554]'.

> Task :app:connectedDebugAndroidTest
Starting 7 tests on Medium_Phone_API_30(AVD) - 11


BUILD SUCCESSFUL in 39s
57 actionable tasks: 15 executed, 42 up-to-date
```

6- GitHub:

https://github.com/Sargazi77/CSC470_module7

References