**Lab Exercise 2- Exploring the Text Component in Jetpack Compose**

**Objective:**

Understand and experiment with the Text component in Jetpack Compose by learning how to:

* Display text.
* Customize text appearance (size, color, style).
* Apply text alignment, font weights, and font styles.

**1. Display Simple Text**

**Task**: Create a composable that displays a simple text.

@Composable

fun SimpleText() {

Text(text = "Hello, Jetpack Compose!")

}

**Instructions**:

* Display a static text "Hello, Jetpack Compose!" in the middle of the screen.

**2. Customizing Text Appearance**

**Task**: Modify the text's font size, color, and style.

import androidx.compose.ui.graphics.Color

import androidx.compose.ui.unit.sp

@Composable

fun CustomText() {

Text(

text = "Customized Text!",

fontSize = 24.sp,

color = Color.Blue

)

}

**Instructions**:

* Change the text color to blue.
* Set the font size to 24sp.

**3. Font Weight and Style**

**Task**: Experiment with different font weights and styles.

import androidx.compose.ui.text.font.FontWeight

import androidx.compose.ui.text.font.FontStyle

@Composable

fun StyledText() {

Text(

text = "Bold and Italic Text",

fontWeight = FontWeight.Bold,

fontStyle = FontStyle.Italic

)

}

**Instructions**:

* Apply **bold** and **italic** styles to the text.

**4. Text Alignment**

**Task**: Center-align the text within a Box.

import androidx.compose.foundation.layout.Box

import androidx.compose.foundation.layout.fillMaxSize

import androidx.compose.ui.Alignment

@Composable

fun CenterAlignedText() {

Box(modifier = Modifier.fillMaxSize(), contentAlignment = Alignment.Center) {

Text(text = "Centered Text")

}

}

**Instructions**:

* Use a Box to center-align the text in the middle of the screen both vertically and horizontally.

**5. Multiple Text Styles with AnnotatedString**

**Task**: Use AnnotatedString to style different parts of the text.

import androidx.compose.ui.text.AnnotatedString

import androidx.compose.ui.text.SpanStyle

import androidx.compose.ui.text.buildAnnotatedString

import androidx.compose.ui.text.withStyle

@Composable

fun MultiStyledText() {

Text(buildAnnotatedString {

append("This is ")

withStyle(style = SpanStyle(fontWeight = FontWeight.Bold)) {

append("bold ")

}

append("and this is ")

withStyle(style = SpanStyle(fontStyle = FontStyle.Italic)) {

append("italic.")

}

})

}

**Instructions**:

* Display text where certain words are **bold** and others are **italic**.

**6. Text Overflow Handling**

**Task**: Handle cases where the text is too long and overflows its container.

import androidx.compose.ui.text.style.TextOverflow

@Composable

fun OverflowText() {

Text(

text = "This is a very long text that might not fit in a single line.",

maxLines = 1,

overflow = TextOverflow.Ellipsis

)

}

**Instructions**:

* Set a limit of 1 line for the text.
* Use TextOverflow.Ellipsis to display an ellipsis (...) if the text exceeds one line.

**7. Interactive Text with Mutable State**

**Task**: Change the text when a button is clicked.

import androidx.compose.runtime.\*

import androidx.compose.material3.Button

@Composable

fun InteractiveText() {

var text by remember { mutableStateOf("Click the button to change text!") }

Column {

Text(text = text)

Button(onClick = { text = "Text changed!" }) {

Text("Change Text")

}

}

}

**Instructions**:

* Display a button below the text.
* Change the text when the button is clicked.

**8. Exercise Summary**

You have now learned the following:

* Basic usage of the Text composable.
* Customizing text appearance with size, color, weight, and style.
* Using AnnotatedString for multiple styles within a single text.
* Handling text overflow.
* Creating interactive text that updates based on user input.