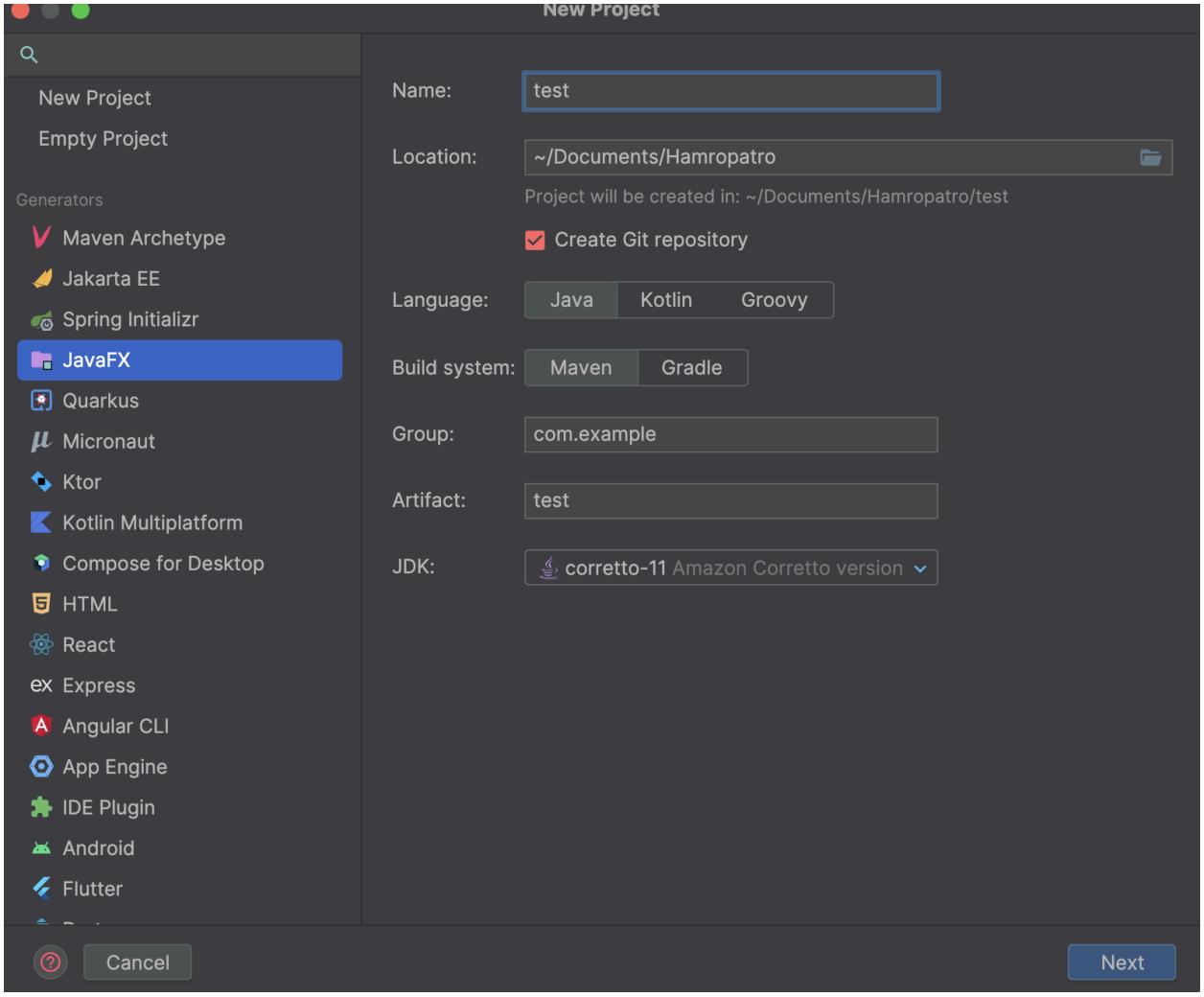
JavaFX Lab Setup Instructions

1. Download IntelliJ IDEA from the official website:
   https://www.jetbrains.com/idea/

2. Install IntelliJ IDEA by following the on-screen installation steps.

3. After installation, launch IntelliJ IDEA.

4. Create a new Java project and configure it to support JavaFX development.



Q. Write a JavaFX application with components, buttons, text fields, and labels, arranged in a VBox or HBox layout. (Look in teams)

```
package com.example.javafxdemo;

import javafx.application.Application;
import javafx.geometry.Insets;
import javafx.geometry.Pos;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.control.TextField;
import javafx.scene.layout.VBox;
import javafx.stage.Stage;

import java.io.IOException;

public class HelloApplication extends Application {
    @Override
    public void start(Stage stage) throws IOException {
        // 1. Create Individual UI Components (Nodes)
        Label welcomeLabel = new Label("This is a simple Label!");
        TextField inputField = new TextField();
        inputField.setPromptText("Just a text field for show..."); // Placeholder text

        Button displayButton = new Button("Look, a Button!");
        // We're not adding any action to this button for now, just displaying it.

        // You can add more components here:
        // CheckBox myCheckBox = new CheckBox("Check me out!");
        // Slider mySlider = new Slider(0, 100, 50); // Min, Max, Initial value

        // 2. Choose a Layout Container (the "root" of your Scene)
        // VBox is a common choice to arrange things vertically.
        VBox rootLayout = new VBox(15); // Spacing of 15 pixels between children

        // 3. Configure the Layout Container (optional styling/positioning)
        rootLayout.setPadding(new Insets(25)); // Padding around the edges of the VBox
        rootLayout.setAlignment(Pos.CENTER); // Center all children within the VBox horizontally

        // 4. Add the UI Components to the Layout Container's children list
        rootLayout.getChildren().addAll(welcomeLabel, inputField, displayButton);
        // If you added more components:
```

```
        // rootLayout.getChildren().addAll(welcomeLabel, inputField, displayButton, myCheckBox,
mySlider);

        // 5. Create the Scene, giving it the root layout and preferred dimensions
        Scene scene = new Scene(rootLayout, 400, 300); // Width, Height

        // 6. Set the Stage's properties
        stage.setTitle("Component Showcase"); // Title for the window
        stage.setScene(scene); // Assign the scene to the stage

        // 7. Show the Stage (make the window visible)
        stage.show();
    }

    public static void main(String[] args) {
        launch();
    }
}
```