Main code:

package com.example.demo1;  
  
import javafx.application.Application;  
import javafx.application.Platform;  
import javafx.event.ActionEvent;  
import javafx.event.EventHandler;  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Scene;  
import javafx.scene.control.\*;  
import javafx.scene.layout.GridPane;  
import javafx.scene.layout.Pane;  
import javafx.stage.Stage;  
  
import java.io.IOException;  
  
public class connect4 extends Application {  
 private Controller controller;  
 @Override  
 public void start(Stage stage) throws IOException {  
 FXMLLoader loader = new FXMLLoader(connect4.class.getResource("hello-view.fxml"));  
 GridPane rootGridPane = loader.load();  
 controller = loader.getController();  
 controller.createPlayground();  
 MenuBar menubar=createMenu();  
 menubar.prefWidthProperty().bind(stage.widthProperty());  
 Pane menuPane= (Pane) rootGridPane.getChildren().get(0);  
 menuPane.getChildren().add(menubar);  
 Scene scene=new Scene(rootGridPane);  
 stage.setScene(scene);  
 stage.setTitle("Connect Four");  
 stage.setResizable(false);  
 stage.show();  
  
 }  
 private MenuBar createMenu(){  
 Menu fileMenu=new Menu("File");  
 MenuItem newGame=new MenuItem("New Game");  
 newGame.setOnAction(actionEvent -> controller.resetGame());  
 MenuItem resetGame=new MenuItem("Reset Game");  
 resetGame.setOnAction(actionEvent -> controller.resetGame());  
 SeparatorMenuItem spm=new SeparatorMenuItem();  
 MenuItem exitGame=new MenuItem("Exit Game");  
 exitGame.setOnAction(actionEvent ->{  
 Platform.*exit*();  
 System.*exit*(0);  
 });  
 fileMenu.getItems().addAll(newGame,resetGame,spm,exitGame);  
 Menu helpMenu=new Menu("Help");  
 MenuItem aboutGame=new MenuItem("About Connect4");  
 aboutGame.setOnAction(actionEvent -> aboutConnect4());  
 SeparatorMenuItem s=new SeparatorMenuItem();  
 MenuItem aboutMe=new MenuItem("About Developer");  
 aboutMe.setOnAction(actionEvent -> aboutMe());  
 helpMenu.getItems().addAll(aboutGame,s,aboutMe);  
 MenuBar menuBar=new MenuBar();  
 menuBar.getMenus().addAll(fileMenu,helpMenu);  
 return menuBar;  
 }  
  
 private void aboutMe() {  
 Alert a = new Alert(Alert.AlertType.*INFORMATION*);  
 a.setTitle("About the Developer");  
 a.setHeaderText("Vedanth");  
 a.setContentText("I love games so I keep creating games. Connect 4 is one of it");  
 a.show();  
 }  
  
 private void aboutConnect4() {  
 Alert alert=new Alert(Alert.AlertType.*INFORMATION*);  
 alert.setTitle("About Connect4 Game");  
 alert.setHeaderText("How to play?");  
 alert.setContentText("Connect Four is a two-player connection game in which the players first choose a color and then take turns dropping colored discs from the top into a seven-column, six-row vertically suspended grid."+" The pieces fall straight down, occupying the next available space within the column."+" The objective of the game is to be the first to form a horizontal, vertical, or diagonal line of four of one's own discs."+" Connect Four is a solved game."+" The first player can always win by playing the right moves.");  
 alert.show();  
 }  
  
 public static void main(String[] args) {  
 *launch*();  
 }  
 }

Controller code:

package com.example.demo1;  
  
import javafx.animation.TranslateTransition;  
import javafx.application.Platform;  
import javafx.fxml.FXML;  
import javafx.fxml.Initializable;  
import javafx.geometry.Point2D;  
import javafx.scene.control.\*;  
import javafx.scene.layout.GridPane;  
import javafx.scene.layout.Pane;  
import javafx.scene.paint.Color;  
import javafx.scene.shape.Circle;  
import javafx.scene.shape.Rectangle;  
import javafx.scene.shape.Shape;  
import javafx.util.Duration;  
  
import java.net.URL;  
import java.util.ArrayList;  
import java.util.List;  
import java.util.Optional;  
import java.util.ResourceBundle;  
import java.util.stream.Collectors;  
import java.util.stream.IntStream;  
  
public class Controller implements Initializable {  
 private static final int *COLUMNS* = 7;  
 private static final int *ROWS* = 6;  
 private static final int *CIRCLE\_DIAMETER* = 80;  
 private static final String *discColor1* = "#24303E";  
 private static final String *discColor2* = "#4CAA88";  
 private boolean isp1 = true;  
 private Disc[][] array = new Disc[*ROWS*][*COLUMNS*];  
 @FXML  
 public TextField t1;  
 @FXML  
 public TextField t2;  
 @FXML  
 public Button b1;  
 @FXML  
 public GridPane rootGridPane;  
 @FXML  
 public Pane insertedDiscsPane;  
 @FXML  
 public Label playerNameLabel;  
 String p1;  
 String p2;  
 private double ro;  
 private boolean isAllowedto = true;  
  
 public void createPlayground() {  
 Shape r = new Rectangle((*COLUMNS* + 1) \* *CIRCLE\_DIAMETER*, (*ROWS* + 1) \* *CIRCLE\_DIAMETER*);  
 for (int row = 0; row < *ROWS*; row++) {  
 for (int col = 0; col < *COLUMNS*; col++) {  
 Circle c = new Circle();  
 c.setRadius(*CIRCLE\_DIAMETER* / 2);  
 c.setCenterX(*CIRCLE\_DIAMETER* / 2);  
 c.setCenterY(*CIRCLE\_DIAMETER* / 2);  
 c.setSmooth(true);  
 c.setTranslateX(col \* (*CIRCLE\_DIAMETER* + 5) + *CIRCLE\_DIAMETER* / 4);  
 c.setTranslateY(row \* (*CIRCLE\_DIAMETER* + 5) + *CIRCLE\_DIAMETER* / 4);  
 r = Shape.*subtract*(r, c);  
 }  
 }  
 r.setFill(Color.*WHITE*);  
 rootGridPane.add(r, 0, 1);  
 List<Rectangle> rectangleList = createClickableColumns();  
 for (Rectangle rectangle : rectangleList) {  
 rootGridPane.add(rectangle, 0, 1);  
 }  
 b1.setOnAction(event -> {  
 p1 = t1.getText();  
 p2 = t2.getText();  
 playerNameLabel.setText(isp1 ? p1 : p2);  
 });  
 }  
  
 private List<Rectangle> createClickableColumns() {  
 List<Rectangle> rectangleList = new ArrayList<>();  
 for (int col = 0; col < *COLUMNS*; col++) {  
  
 Rectangle rect = new Rectangle(*CIRCLE\_DIAMETER*, (*ROWS* + 1) \* *CIRCLE\_DIAMETER*);  
 rect.setFill(Color.*TRANSPARENT*);  
 rect.setTranslateX(col \* (*CIRCLE\_DIAMETER* + 5) + (*CIRCLE\_DIAMETER* / 4));  
 rect.setOnMouseEntered(event -> rect.setFill(Color.*valueOf*("#eeeeee26")));  
 rect.setOnMouseExited(event -> rect.setFill(Color.*TRANSPARENT*));  
 final int column = col;  
 rect.setOnMouseClicked(event -> {  
 if (isAllowedto) {  
 isAllowedto = false;  
 insertDisc(new Disc(isp1), column);  
 }  
 });  
 rectangleList.add(rect);  
 }  
 return rectangleList;  
 }  
  
 private void insertDisc(Disc disc, int column) {  
 int row = *ROWS* - 1;  
 while (row >= 0) {  
 if (getDiscIfPresent(row, column) == null)  
 break;  
 row--;  
 }  
 if (row < 0)  
 return;  
 array[row][column] = disc;  
 insertedDiscsPane.getChildren().add(disc);  
 disc.setTranslateX(column \* (*CIRCLE\_DIAMETER* + 5) + *CIRCLE\_DIAMETER* / 4);  
 TranslateTransition t = new TranslateTransition(Duration.*seconds*(0.5), disc);  
 t.setToY(row \* (*CIRCLE\_DIAMETER* + 5) + *CIRCLE\_DIAMETER* / 4);  
 int currentRow = row;  
 t.setOnFinished(event -> {  
 isAllowedto = true;  
 if (gameEnded(currentRow, column)) {  
 gameOver();  
 return;  
 }  
 isp1 = !isp1;  
 playerNameLabel.setText(isp1 ? p1 : p2);  
 });  
 t.play();  
 }  
  
 private void gameOver() {  
 String winner = isp1 ? p1 : p2;  
 Alert a = new Alert(Alert.AlertType.*INFORMATION*);  
 a.setTitle("Connect 4");  
 a.setHeaderText("Thw winner is " + winner);  
 a.setContentText("Want to play again?");  
 ButtonType y = new ButtonType("Yes");  
 ButtonType n = new ButtonType("No, Exit");  
 a.getButtonTypes().setAll(y, n);  
 Platform.*runLater*(() -> {  
 Optional<ButtonType> bc = a.showAndWait();  
 if (bc.isPresent() && bc.get() == y) {  
 resetGame();  
 } else {  
 Platform.*exit*();  
 System.*exit*(0);  
 }  
 });  
 }  
  
 public void resetGame() {  
 insertedDiscsPane.getChildren().clear();  
 for (int row = 0; row < array.length; row++) {  
 for (int col = 0; col < array[row].length; col++) {  
 array[row][col] = null;  
 }  
 }  
 isp1 = true;  
 createPlayground();  
 }  
  
 private static class Disc extends Circle {  
 private final boolean isP1M;  
  
 public Disc(boolean isP1M) {  
 this.isP1M = isP1M;  
 setRadius(*CIRCLE\_DIAMETER* / 2);  
 setFill(isP1M ? Color.*valueOf*(*discColor1*) : Color.*valueOf*(*discColor2*));  
 setCenterX(*CIRCLE\_DIAMETER* / 2);  
 setCenterY(*CIRCLE\_DIAMETER* / 2);  
 }  
 }  
  
 private boolean gameEnded(int row, int column) {  
 List<Point2D> verticalPoints = IntStream.*rangeClosed*(row - 3, row + 3)  
 .mapToObj(ro -> new Point2D(ro, column))  
 .collect(Collectors.*toList*());  
 List<Point2D> horizontalPoints = IntStream.*rangeClosed*(column - 3, column + 3)  
 .mapToObj(col -> new Point2D(ro, col))  
 .collect(Collectors.*toList*());  
 Point2D startPoint1 = new Point2D(row - 3, column + 3);  
 List<Point2D> diag1Points = IntStream.*rangeClosed*(0, 6)  
 .mapToObj(i -> startPoint1.add(i, -i))  
 .collect(Collectors.*toList*());  
 Point2D startPoint2 = new Point2D(row - 3, column - 3);  
 List<Point2D> diag2Points = IntStream.*rangeClosed*(0, 6)  
 .mapToObj(i -> startPoint2.add(i, i))  
 .collect(Collectors.*toList*());  
 boolean isEnded = checkCombination(verticalPoints) || checkCombination(horizontalPoints) || checkCombination(diag1Points) || checkCombination(diag2Points);  
 return isEnded;  
 }  
  
 private boolean checkCombination(List<Point2D> points) {  
 int chain = 0;  
 for (Point2D point : points) {  
 int rowIndexForArray = (int) point.getX();  
 int columnIndexForArray = (int) point.getY();  
 Disc disc = getDiscIfPresent(rowIndexForArray, columnIndexForArray);  
 if (disc != null && disc.isP1M == isp1) {  
 chain++;  
 if (chain == 4) {  
 return true;  
 }  
 } else {  
 chain = 0;  
 }  
 }  
 return false;  
 }  
  
 private Disc getDiscIfPresent(int row, int column) {  
 if (row >= *ROWS* || row < 0 || column >= *COLUMNS* || column < 0)  
 return null;  
 return array[row][column];  
 }  
  
 public void initialize(URL url, ResourceBundle resourceBundle) {  
  
 }  
}

FXML code:

<?xml version="1.0" encoding="UTF-8"?>  
  
<?import javafx.scene.control.\*?>  
<?import javafx.scene.layout.\*?>  
<?import javafx.scene.text.\*?>  
  
<GridPane fx:id="rootGridPane" style="-fx-background-color: #DDF7F0;" xmlns="http://javafx.com/javafx/16" xmlns:fx="http://javafx.com/fxml/1" fx:controller="com.example.demo1.Controller">  
 <columnConstraints>  
 <ColumnConstraints />  
 <ColumnConstraints maxWidth="340.4" minWidth="179.4" prefWidth="196.0" />  
 </columnConstraints>  
 <rowConstraints>  
 <RowConstraints minHeight="10.0" prefHeight="25.0" />  
 <RowConstraints />  
 </rowConstraints>  
 <children>  
 <Pane fx:id="myPane" GridPane.columnSpan="2" />  
 <Pane fx:id="insertedDiscsPane" prefHeight="400.0" prefWidth="200.0" GridPane.rowIndex="1" />  
 <VBox style="-fx-background-color: #2B3B4C;" GridPane.columnIndex="1" GridPane.rowIndex="1">  
 <children>  
 <Pane prefHeight="288.0" prefWidth="199.0">  
 <children>  
 <TextField fx:id="t1" layoutX="20.0" layoutY="36.0" promptText="Enter Player One Name" />  
 <Label fx:id="playerNameLabel" alignment="CENTER" layoutY="144.0" prefHeight="36.0" prefWidth="180.0" text="Player One" textFill="#faf6f6">  
 <font>  
 <Font name="System Bold" size="25.0" />  
 </font>  
 </Label>  
 <Label alignment="CENTER" layoutX="1.0" layoutY="187.0" prefHeight="36.0" prefWidth="180.0" text="Turn" textFill="#f4f3f3">  
 <font>  
 <Font size="25.0" />  
 </font>  
 </Label>  
 <TextField fx:id="t2" layoutX="20.0" layoutY="70.0" promptText="Enter Player Two Name" />  
 <Button fx:id="b1" layoutX="18.0" layoutY="106.0" mnemonicParsing="false" prefHeight="25.0" prefWidth="149.0" text="Set Names" />  
 </children></Pane>  
 <Region prefHeight="69.0" prefWidth="346.0" />  
 </children>  
 </VBox>  
 </children>  
</GridPane>