CSC 171: Final Project

Joe Droney & Isabella Franco

11/2/23

Farming Simulator User Manual:

The main objective of the Farming Simulator is to allow player(s) to experience the joys and challenges of running their own virtual farm. Within this simulator, users are given a short period of time to plant and harvest their crops. Once the player has successfully harvested all their crops (within the allotted time frame) the players score will increase. Failure to harvest all crops within the given time, will result in a decrease in the player's highscore. Players are allowed to reset the simulator and play until they reach the next level.

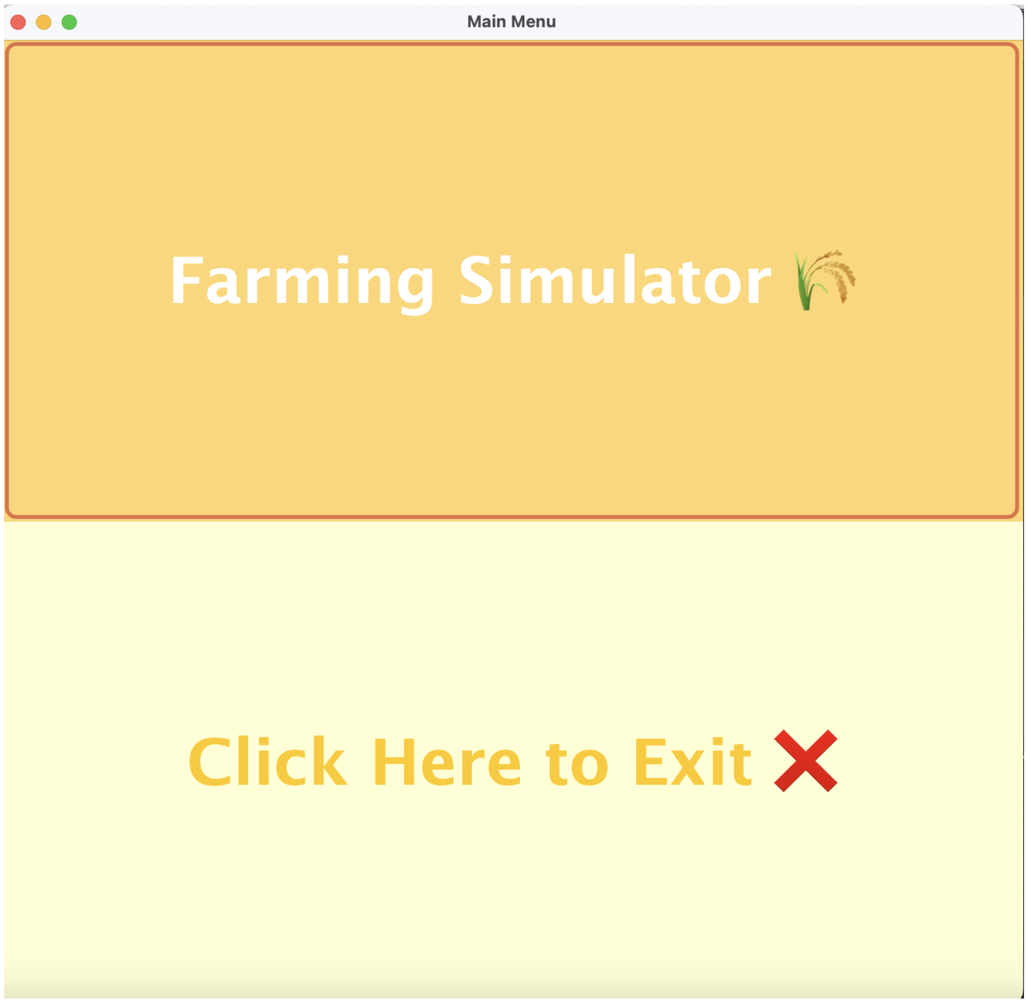
Starting the Farming Simulator

The Main Menu is the starting point of the application. It provides options to either start the Farming Simulator or exit the application.

Click on the "Farming Simulator 🌾" button to start the simulation. This will take you to the farming environment where you can interact with crops.

Exiting the Application

Click on the "Click Here to Exit ❌" button to close the application. (Reference figure 1 down below)



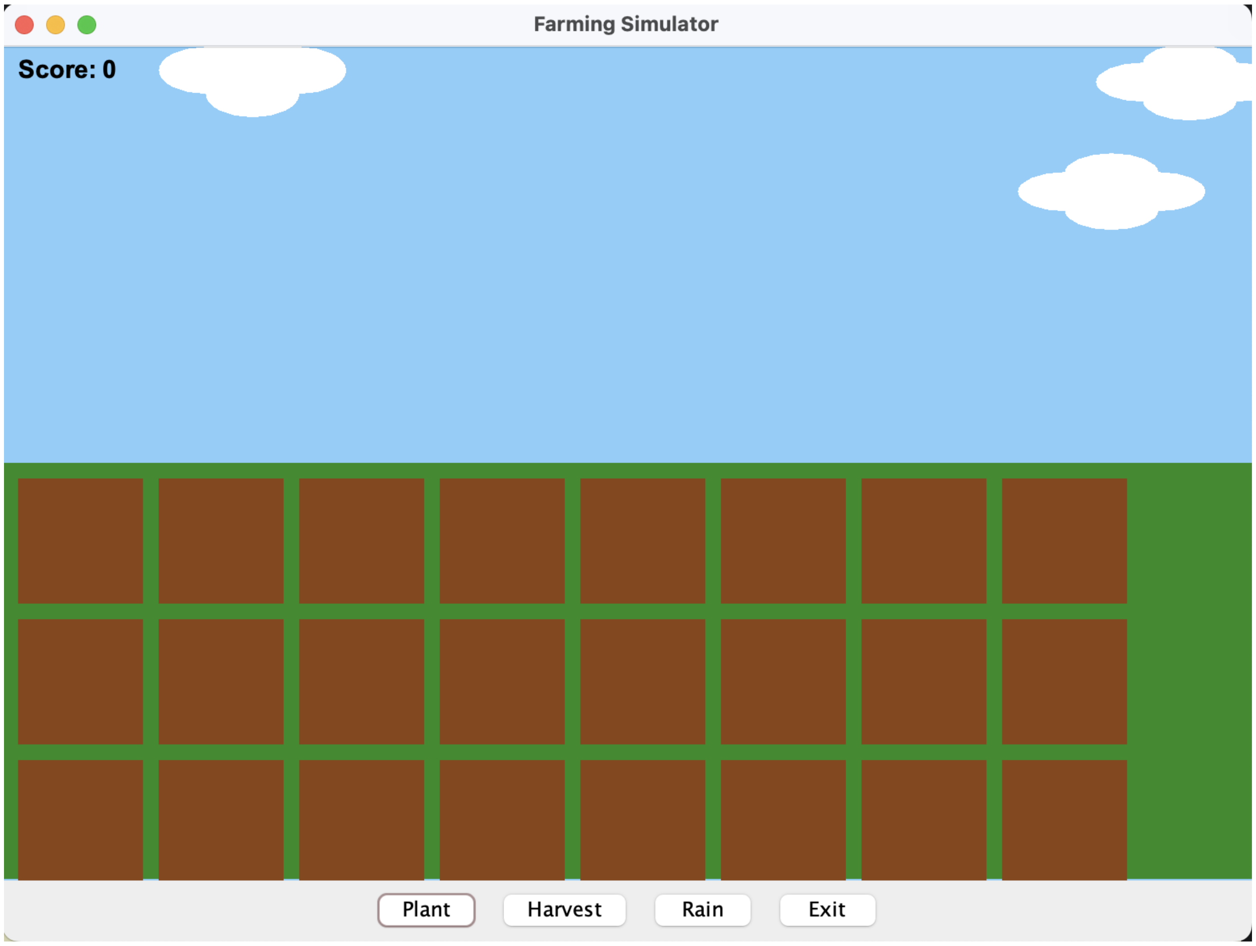
(Figure 1)

Interface

The Farming Simulator interface consists of a grassy area with crop plots. Additional buttons allow you to perform actions such as planting, harvesting, and triggering rain.

Crop Plots

Brown rectangles represent crop plots. Each plot can be occupied by a growing crop. (Reference figure 2)

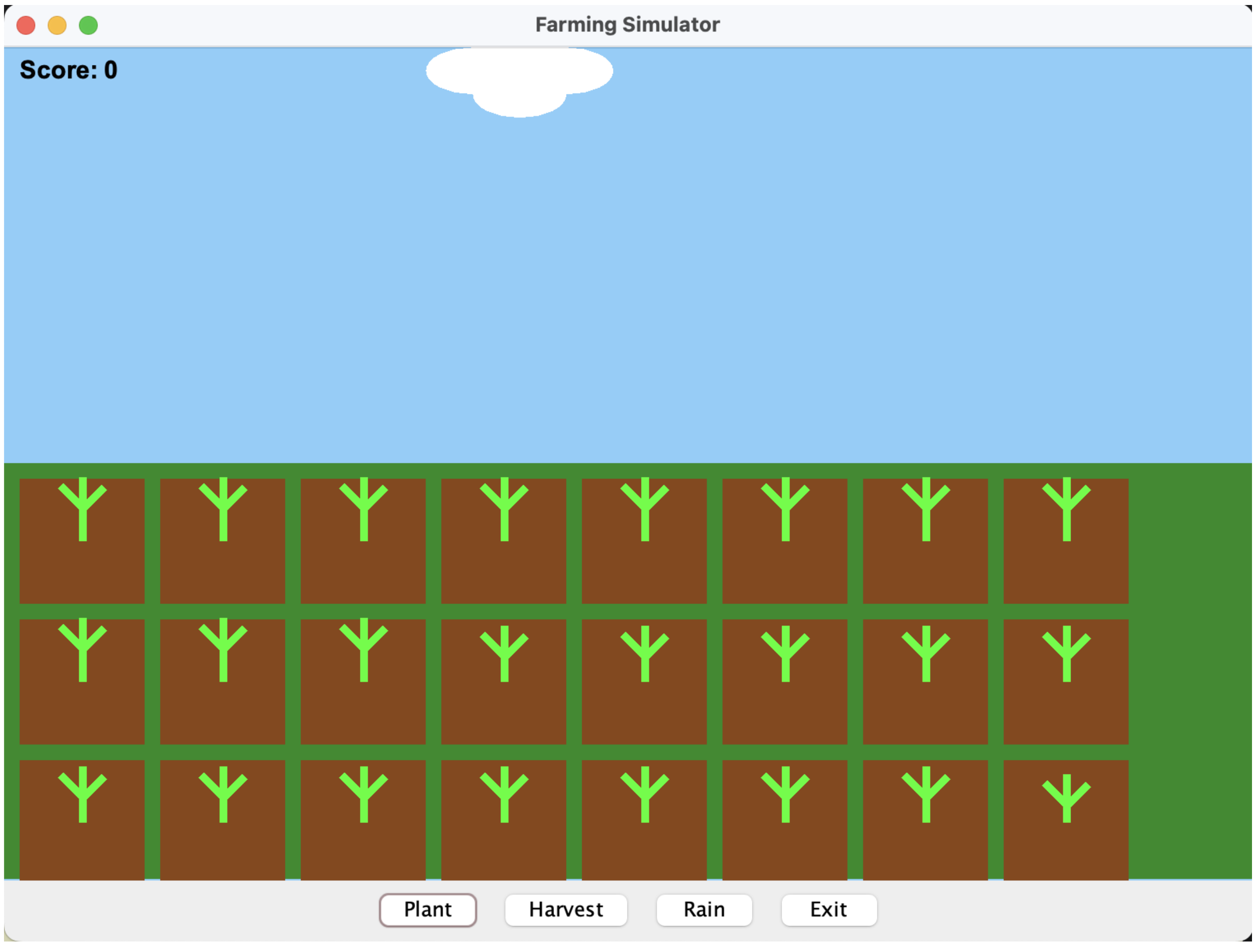


(Figure 2)

Planting Crops

Click on the “Plant” button to plant crops in available crop plots.

Crops will grow over time, simulating their life cycle. (Reference figure 3)



(Figure 3)

Harvesting Crops

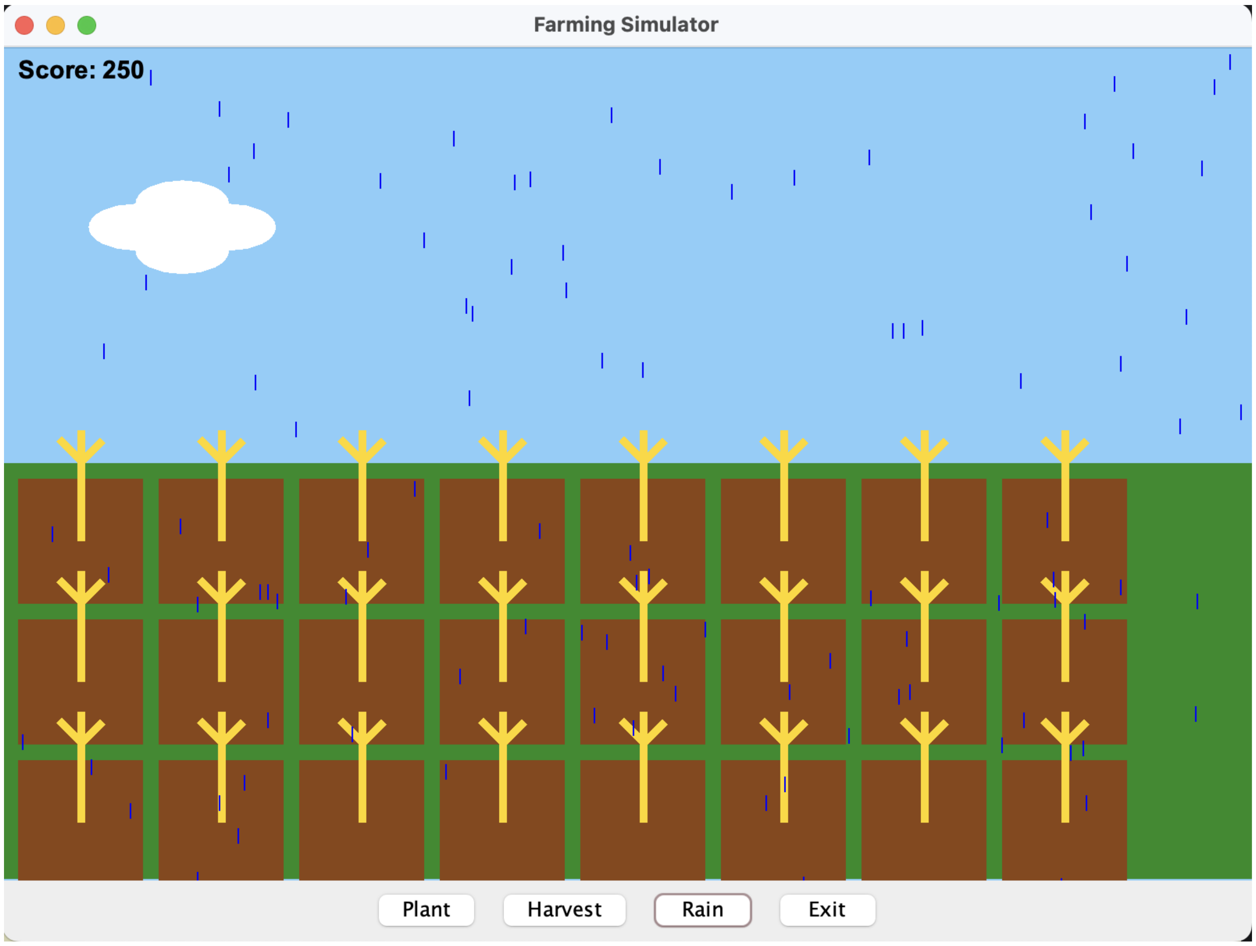
Click on the “Harvest” button to harvest fully grown crops.

Accumulate points for each successful harvest.

Rain Effect

Click on the “Rain” button to trigger rain.

Rain accelerates crop growth. (Reference figure 4)



(Figure 4)

Scoring

The score is displayed at the top-left corner of the interface. Harvesting crops adds points to your score.

Exiting the Simulator

Click on the “Exit” button at the top to exit the Farming Simulator and return to the Main Menu

Farming Simulator Design Document

Overview:

The Farming Simulator application is designed to provide users with a virtual farming experience. Users can interact with a simulated environment, plant crops, harvest them, and experience dynamic weather effects like rain. The application is structured into several classes to manage different aspects of the simulation.

Class Details:

Then MainMenu.java class represents the main-menu window for the application:

```
public class MainMenu extends JFrame {
    public FarmingSimulator farmingSimulator;

    public MainMenu() {
        // Constructor for the main menu window
        // Sets up the main menu window with buttons for starting the simulator and exiting the application
    }

    public void FarmingSimulator() {
        // Method to transition to the FarmingSimulator window
    }

    public static void main(String[] args) {
        // Main method to launch the application
    }
}
```

[Methods]

MainMenu(): Constructor to initialize the main menu window

FarmingSimulator(): Method to transition to the Farming Simulator window

[Relationships] Allows FarmingSimulator to facilitate smooth transitions between the main menu and the simulator.

FarmingSimulator Class:

Represents the main window for the farming simulation and manages the farming environment, crop plots, user interactions, and weather effects.

The FarmingSimualtor class represents the farming simulator window:

```
public class FarmingSimulator extends JFrame {

    public MainMenu mainMenu;

    public FarmingSimulator() {

        // Constructor for the FarmingSimulator window

        // Sets up the farming simulator window with buttons for planting, harvesting, and triggering rain

        // Initializes the background panel and crop plots

    }

    // Other methods for managing crop plots, wheat growth, and rain effects

}
```

[Methods]

FarmingSimulator(): Constructor to initialize the simulator window
initializeCropPlots(): Method to create and initialize crop plots

plantCrop(): Method to plant crops in available crop plots

harvestCrop(): Method to harvest fully grown crops

riggerRain(): Method to initiate the rain effect

[Relationships]

Allows MainMenu to allow users to return to the main menu

Contains BackgroundPanel to manage the rendering of the background and crop plots

BackgroundPanel Class:

Custom JPanel for rendering the background of the simulator and manages crop plots, wheat growth, rain effects, and user interactions

```
private static class BackgroundPanel extends JPanel {
    private List<Wheat> wheatList;
    private List<CropPlot> cropPlots;
    private int score;
    private int cloud1X = 100;
    private int cloud2X = 300;
    // Other fields for managing clouds, rain, etc.

    public BackgroundPanel() {
        // Constructor for the background panel
        // Initializes wheat list, crop plots, and sets up a timer for wheat growth
        // Manages mouse events for triggering rain
    }

    // Other methods for initializing crop plots, planting crops, harvesting crops, and handling rain
effects
}
```

[Methods]

BackgroundPanel(): Constructor to initialize the background panel

initializeCropPlots(): Method to create and initialize crop plots within the grass area

plantCrop(): Method to plant crops in available crop plots

harvestCrop(): Method to harvest fully grown crops

triggerRain(): Method to initiate the rain effect

[Relationships]

Allows Wheat and CropPlot to represent the wheat entities and crop plot entities, respectively.

Wheat Class:

Represents an individual wheat entity in the simulator and manages wheat growth stages and visualization

```
private static class Wheat {

    private int x;

    private int y;

    private int height;

    private int maxGrowth = 50;

    private Color stemColor;

    public Wheat(int x, int y) {

        // Constructor for the Wheat class

    }

    // Methods for drawing, growing, and checking if fully grown

}
```

[Methods]

Wheat(int x, int y): Constructor to initialize a wheat structure

draw(Graphics g): Method to render the wheat structure

grow(): Method to simulate wheat growth

isFullyGrown(): Method to check if the wheat is fully grown

[Relationships]

Used by BackgroundPanel to visualize wheat growth

CropPlot Class:

Represents an individual crop plot in the simulator and manages whether a plot is occupied by a crop

```
private static class CropPlot {

    private int x;

    private int y;

    private int width;

    private int height;

    private boolean occupied;

    public CropPlot(int x, int y, int width, int height) {

        // Constructor for the CropPlot class

    }

    // Method for checking if a point is within the crop plot

}
```

[Methods]

CropPlot(int x, int y, int width, int height): Constructor to initialize a crop plot

containsPoint(int x, int y): Method to check if a point is within the crop plot

[Relationships]

Used by BackgroundPanel to represent and manage crop plots

Class Relationships:

- ★ MainMenu and FarmingSimulator are connected, allowing transitions between the main menu and the farming simulator.
- ★ FarmingSimulator contains a BackgroundPanel for managing the rendering of the background and crop plots.
- ★ BackgroundPanel aggregates Wheat and CropPlot classes to represent wheat entities and crop plot entities, respectively.