

Milestone 7 – Mine Sweeper

Kelly Lamb | Professor James Shinevar M.S. | Grand Canyon University: CST-227 Enterprise Application Programming II

Problem / Question

Create a visually appealing, functionally superior Mine-Sweeper application for serious game play using C#.NET and Visual Studio

Hypothesis

- Create C# classes, interfaces, enumerations and forms to handle data storage, models, logic controllers and services to support all aspects of the Mine Sweeper game. Utilize advanced techniques in C# to store/retrieve persistent data; Create and utilize custom and existing libraries for visually appealing presentation.
- Design processes incrementally – confirming results in stages

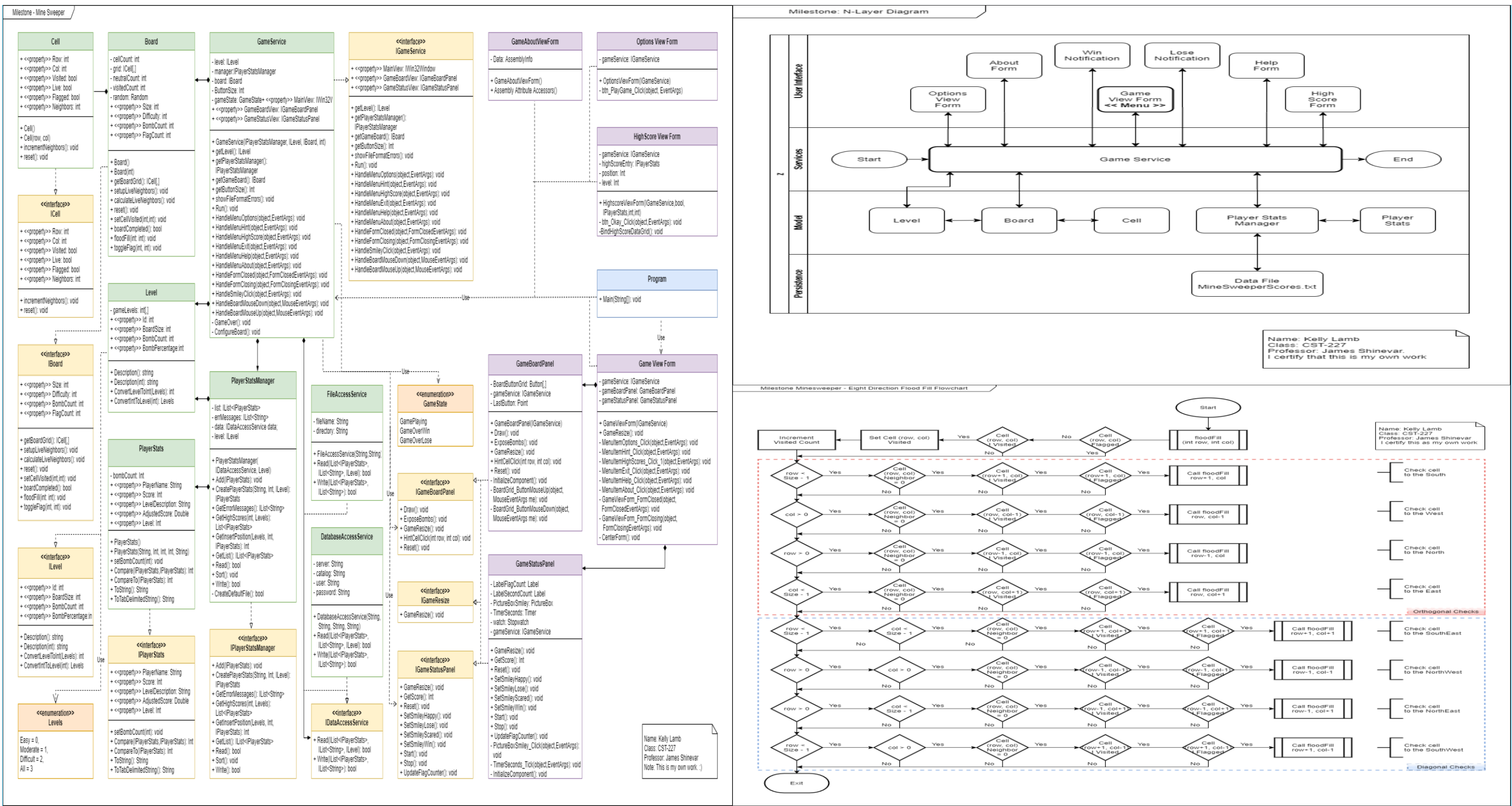
Project Overview

1. Develop application logic and flesh it out in a wireframe
2. Design Classes to model data and logic
3. Create UML documents and define dependencies / relationships
4. Create Flowcharts, initial logic; updated version with classes
5. Write C# code – confirm that it follows flowchart logic
6. Create videos to demonstrate effective progress in each step
7. Modify code to final stage to produce visually appealing product
8. Modify UML diagram to more accurately represent final product
9. Produce a poster that highlights the process

Classes / Research

Model Classes	Data Access Classes	Visual Classes
Board.cs Cell.cs GameService.cs IBoard.cs ICell.cs IGameResize.cs IGameService.cs ILevel.cs Level.cs IPlayerStats.cs IPlayerStatsManager.cs PlayerStats.cs PlayerStatsManager.cs	DatabaseAccessService.cs FileAccessService.cs IDataAccessService.cs IPlayerStats.cs IPlayerStatsManager.cs PlayerStats.cs PlayerStatsManager.cs	AboutViewForm.cs GameBoardPanel.cs GameStatusPanel.cs GameViewForm.cs HighScoreViewForm.cs IGameBoardPanel.cs IGameStatusPanel.cs OptionsViewForm.cs

Flow Chart / UML



Procedure - Milestones

Step 1

Design initial logic flowchart, mockups, wireframes

Step 2

Design Data Model - ADT, UML, Wireframe, N-Layer

Step 3

Incremental Design, Write, Test, Research C#.NET

Step 4

Create Visual classes and merge with data model

Base Code Snippets

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Minesweeper
{
    public class GameService : IGameService
    {
        private readonly IBoard board;
        private readonly ILevel level;
        private readonly IPlayerStatsManager statsManager;
        private readonly IGameBoardPanel gameBoardPanel;
        private readonly IGameStatusPanel gameStatusPanel;
        private readonly IOptionsViewForm optionsViewForm;

        public GameService(IBoard board, ILevel level, IPlayerStatsManager statsManager, IGameBoardPanel gameBoardPanel, IGameStatusPanel gameStatusPanel, IOptionsViewForm optionsViewForm)
        {
            this.board = board;
            this.level = level;
            this.statsManager = statsManager;
            this.gameBoardPanel = gameBoardPanel;
            this.gameStatusPanel = gameStatusPanel;
            this.optionsViewForm = optionsViewForm;
        }

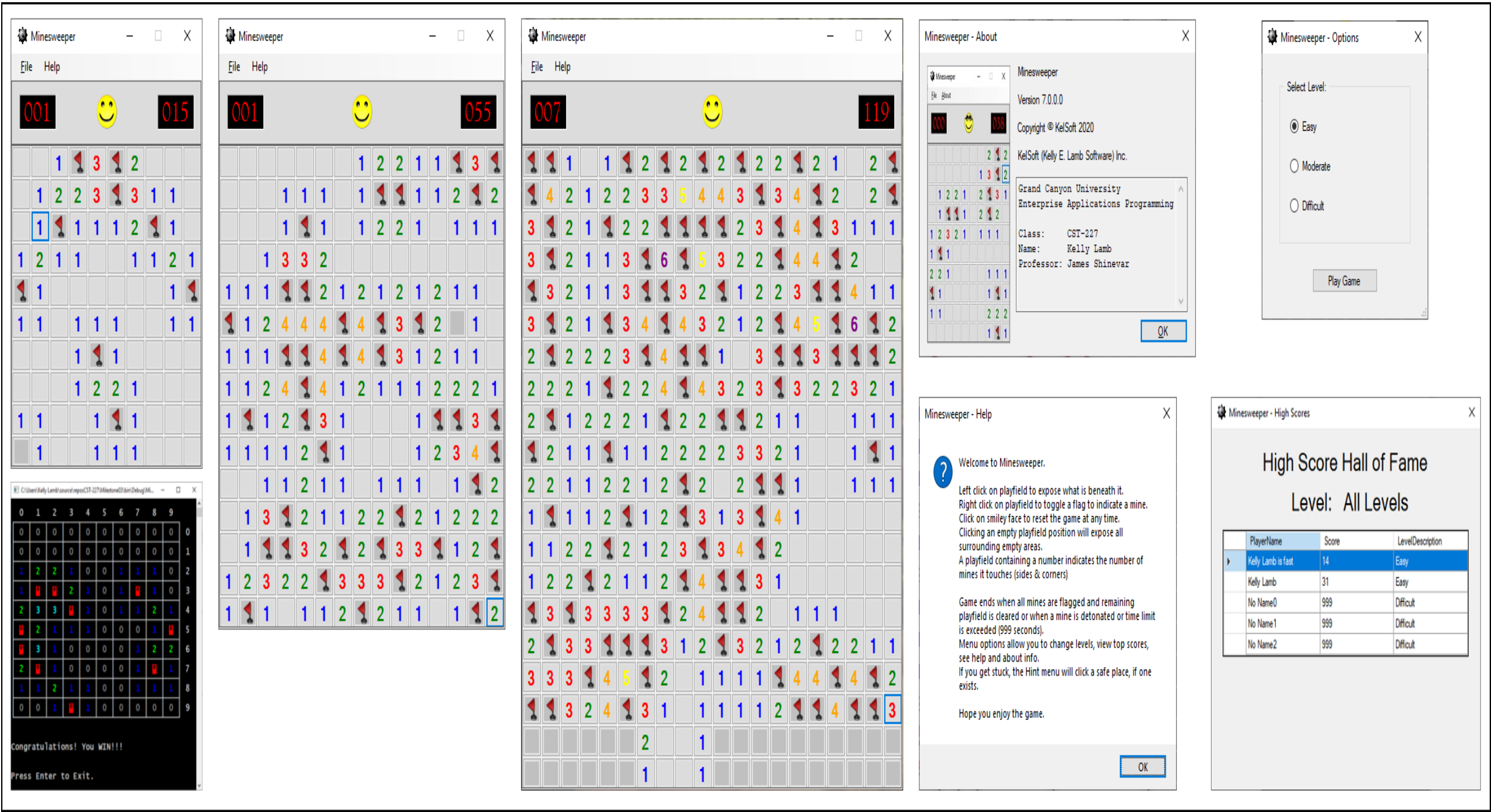
        public void StartGame()
        {
            // Start the game
        }

        public void HandleClick(int row, int col)
        {
            // Handle a click on the board
        }

        public void ShowOptions()
        {
            // Show the options dialog
        }

        public void ShowHighScore()
        {
            // Show the high score list
        }
    }
}
```

Results - Screenshots



- Application initializes, loads data, handles all process requirements
- Screenshots: Represents all implemented functionality
- Each GUI Level, Console Version (Easy), About, Options, Help, and High score forms
- Number of classes: 16 (excluding form designer)
- Number of form layouts: 8
- Number of lines: C# code: 4,934

Conclusion

- Minesweeper – Surprisingly fun to design and code but provided strong logic requirements, decisions, and solid OOP analysis and code constructs.
- Flowchart, UML, Wireframe, and N-Layer Diagrams are great tools to help in the design process
- Visual Studio offers a robust graphics programming platform but required significant research and understanding – many tutorials online.
- Very satisfying outcome watching everything work together.

Works Cited

Gittleman, A. (2012). Computing with C# and the .NET Framework docs.microsoft.com

Nagel, C. (2018). Professional C# 7 and .Net Core 2.0 (7th ed.)

Troelsen, A., & Japikse, P. (2017). Pro C# 7: With .Net and .Net Core (8th ed.)

docs.microsoft.com/en-us/dotnet/csharp

www.geeksforgeeks.org

csharp.net-tutorials.com

www.tutorialspoint.com

www.c-sharpcorner.com

www.youtube.com (IAmTimCorey, ShadSluiter, Several other tutorials)