Republic of the Philippines
RIZAL TECHNOLOGICAL UNIVERSITY
Cities of Mandaluyong and Pasig
COLLEGE OF ENGINEERING, ARCHITECHTURE AND TECHNOLOGY

COMPUTER PROGRAMMING 2

(ITP123L)

<ODYSSEY: NEW WORLD>

GRADE

Submitted by:

CORDERO, KYLE MATTHEW A.

BSIT

CEIT-37-202A

DE VERA, MARK LAURENCE.

BSIT

CEIT-37-202A

GUILLERMO, ROXANNE L.

BSIT

CEIT-37-202A

HO, LANCE STEPHEN

BSIT

CEIT-37-202A

INIGO, CYRIL ANNE A.
.

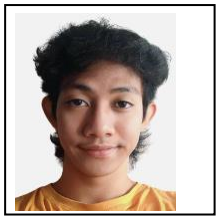
BSIT

CEIT-37-202A

ORTIZ, CRIS ZALDY T.

BSIT

CEIT-37-202A



RILLORTA, ANGELO L.

BSIT

CEIT-37-202A

TULIN, ROSELLE

BSIT

CEIT-37-202A

Submitted to:

MAY BARCELONA FIGUEROA

Professor

May 2023

Table of Contents

IX. INTRODUCTION

This interactive game invites players to embark on a captivating journey within a manageable playtime of 10-15 minutes. In the game's initial version, players were treated to a detailed story that vividly described different scenarios, resembling a captivating book. However, the latest version takes the experience to new heights by incorporating visually stunning images that depict each scene. Exploration lies at the core of the game, as players venture through various areas connected to the main storyline, with the narrative evolving as they progress. As players conquer specific challenges in each area, they unlock visually breathtaking levels and areas, serving as rewards for their achievements. Within the digital world of Odyssey: New World, an immersive and memorable adventure awaits, where engaging storytelling, interactive gameplay, and striking visuals merge to create an extraordinary gaming experience that will leave a lasting impression.

X. DESCRIPTION OF THE PROJECT

In "The Odyssey: New World," players are engrossed in a captivating plot that transports them to a mysterious town devoid of inhabitants. As the protagonist, their choices hold the key to the game's outcome, driving them to scavenge crucial clues and resources from locations like the Town's Police Station, Town Hall, Attic, Mini Market, Library, and Basement. In the latest version, decision-making is made more intuitive, enabling players to effortlessly click on their desired choices instead of relying on numbered options. Each decision is accompanied by vibrant prompts, offering vivid descriptions of the current situation and subtle hints for progression. Unlike the previous version, the updated release replaces the ASCII text art visuals with actual pictures generated by AI technology, keeping up with the latest trend. These actual pictures depict the player's current surroundings, enhancing the visual experience. With its captivating storyline, interactive decision-making mechanics, and visually immersive elements, "The Odyssey: New World" promises an enthralling adventure that will leave players eagerly exploring the enigmatic town and uncovering its well-guarded secrets.

XI. OBJECTIVES

Creating an interactive adventure game for players aged 13 and up requires compelling storytelling, immersive gameplay mechanics, and engaging characters. By developing a captivating narrative with mystery and twists, offering meaningful choices and consequences, and creating relatable characters, players become invested in the game. Visually stunning to enhance the experience. In summary, a well-crafted adventure game incorporates these elements to entertain and captivate a wide range of players.

XII. SIGNIFICANCE OF THE STUDY

Keeping up with the technological trend, this research aims to familiarize the following with the fundamentals of game development using the C++ programming language specifically to the following:

Gamers. The result of this program will be an advantage for players who will be playing the game since they will learn how to appreciate the very basic style of a game.

Future IT Researchers. The output of this study will benefit future researchers by giving them references and additional information on the creation of simple games using C++ programming language.

XIII. SCOPE AND DELIMITATIONS

Odyssey is an interactive adventure game, where the player explores a seemingly abandoned town with various levels and interacts with buttons where the choices are placed. The game acts like an interactive novel where it narrates and shows the player's location and the result of the player's actions.

Since the backbone of the game is built on an algorithm that relies on flow control statements such as if, else if and else, for, and while loops, it has limitations on other features that are commonly present on current games. The game does not utilize a file stream library which results in the game lacking a save and load feature. Since the game needs to be able to write and read a file for it to store necessary data, the absence of this results in the lack of such a feature. The game is also developed to terminate once a specific set of conditions are met:

- The player's age is 13 or below
- Reached one of the different endings

The game isn't designed to start over once those conditions are met. The player would have to re-launch the game to continue gameplay.

XIV. SCREEN OUTPUT

Figure 1. Main Menu Window

Figure 1 shows Odyssey's main menu window along with the various options the player can select from.

1. **Start** – this option will load the game's assets and will display a progress bar to indicate the loading process.
2. **About** – this option will provide the user a brief description of the game, along with the respective roles of each individual that contributed to the game's development.
3. **Quit –** immediate exit from the game.

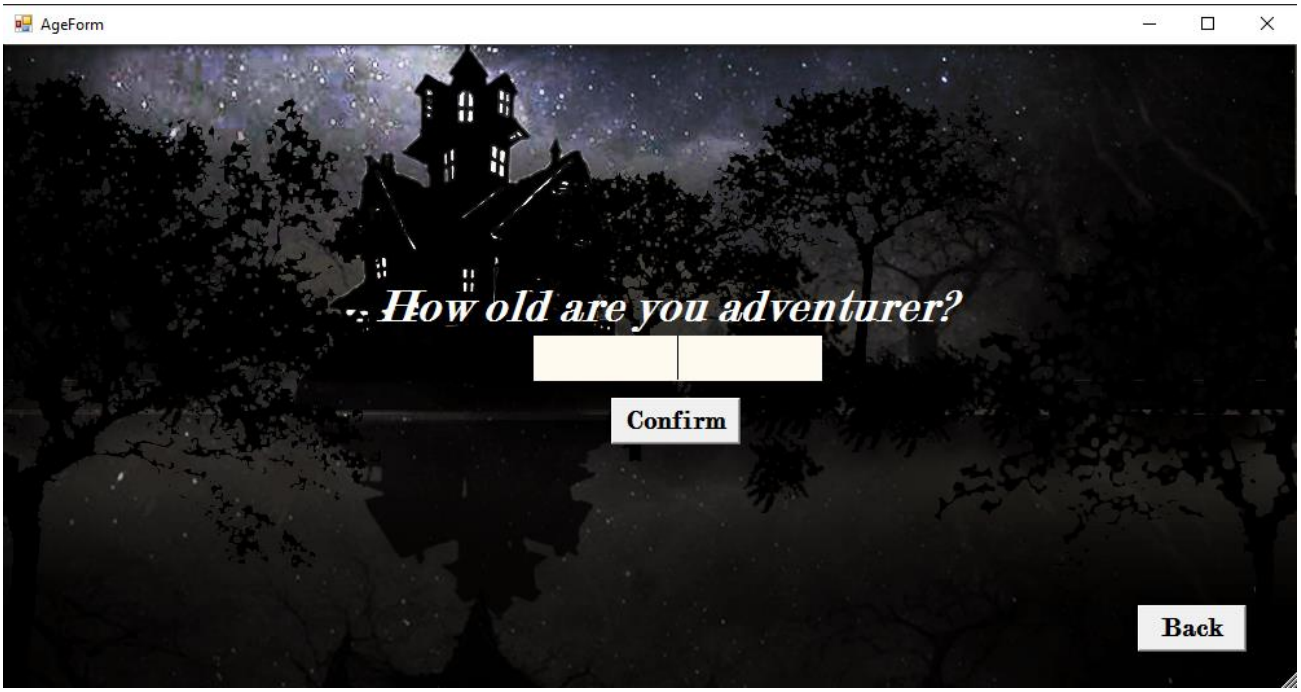


Figure 2. Age Restriction

Figure 2 shows the form where the player has to input their name because of the age restriction feature of the game.

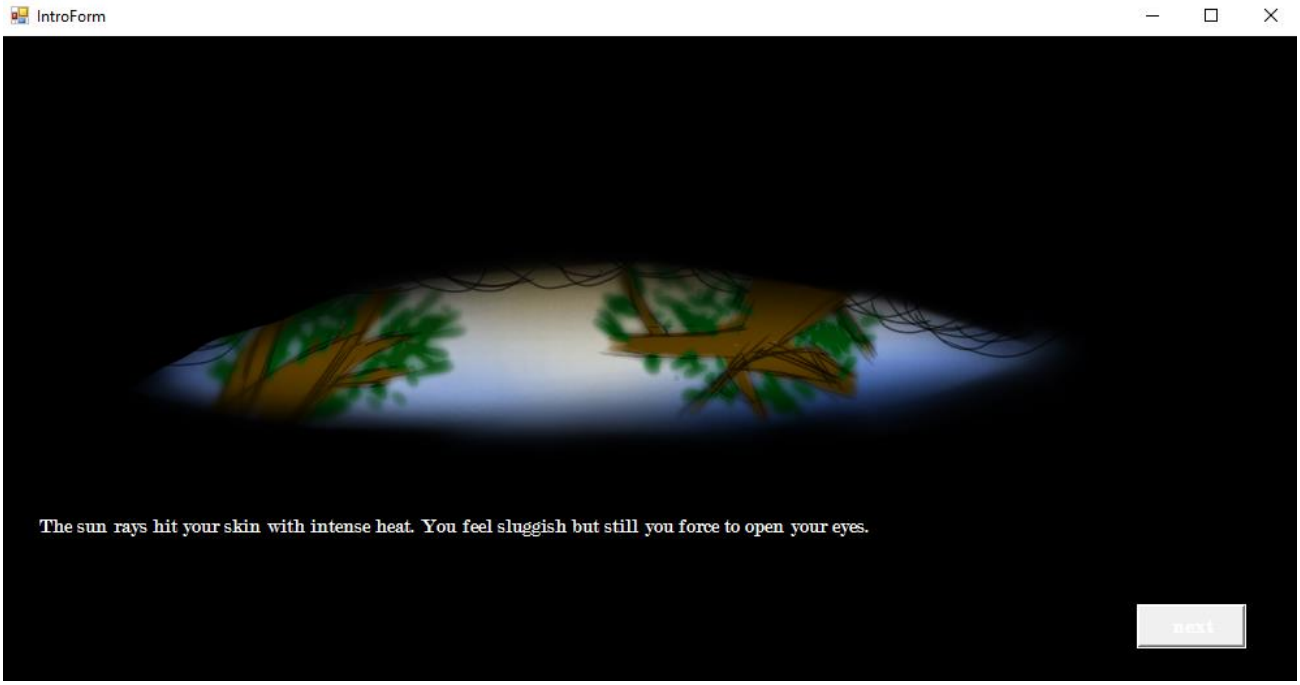


Figure 3. The Beginning

Figure 3 shows the beginning of the story in the game, and the option that players can choose from will be shown in the incoming forms. Those choices will affect the player's decision on how the story will proceed.

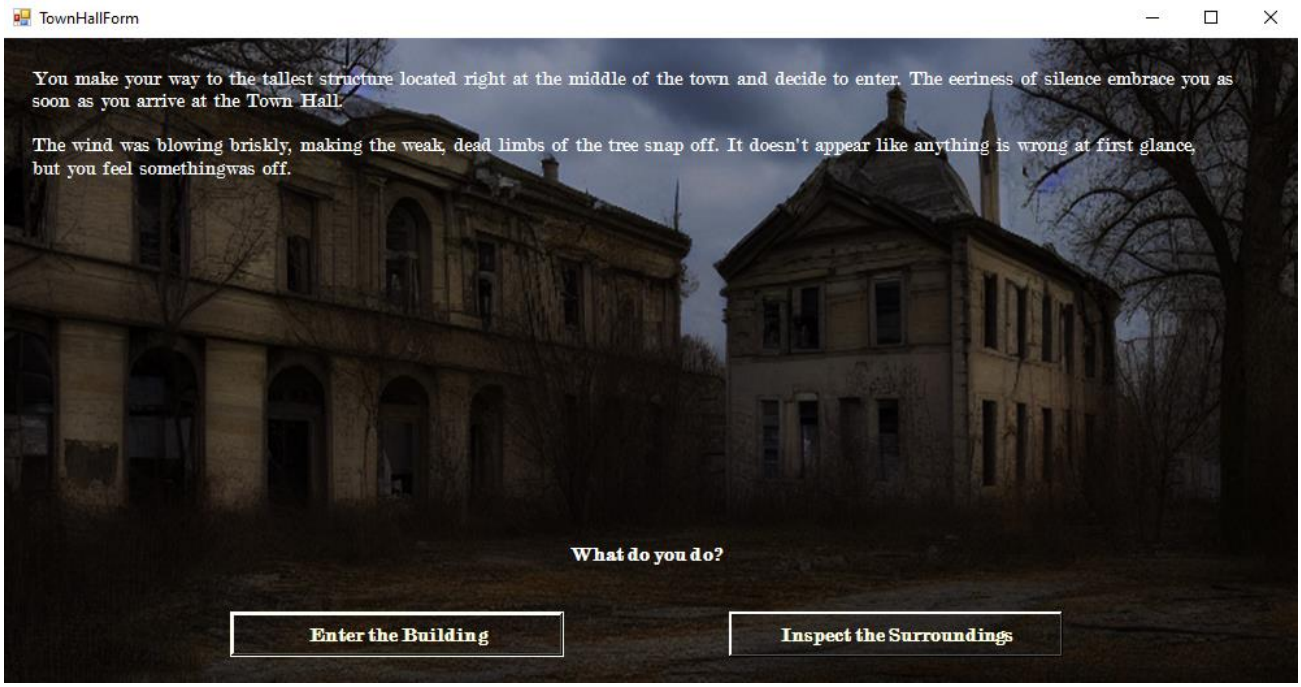


Figure 4. Town Hall

Figure 4 shows when the player was done with the mission on the first level.

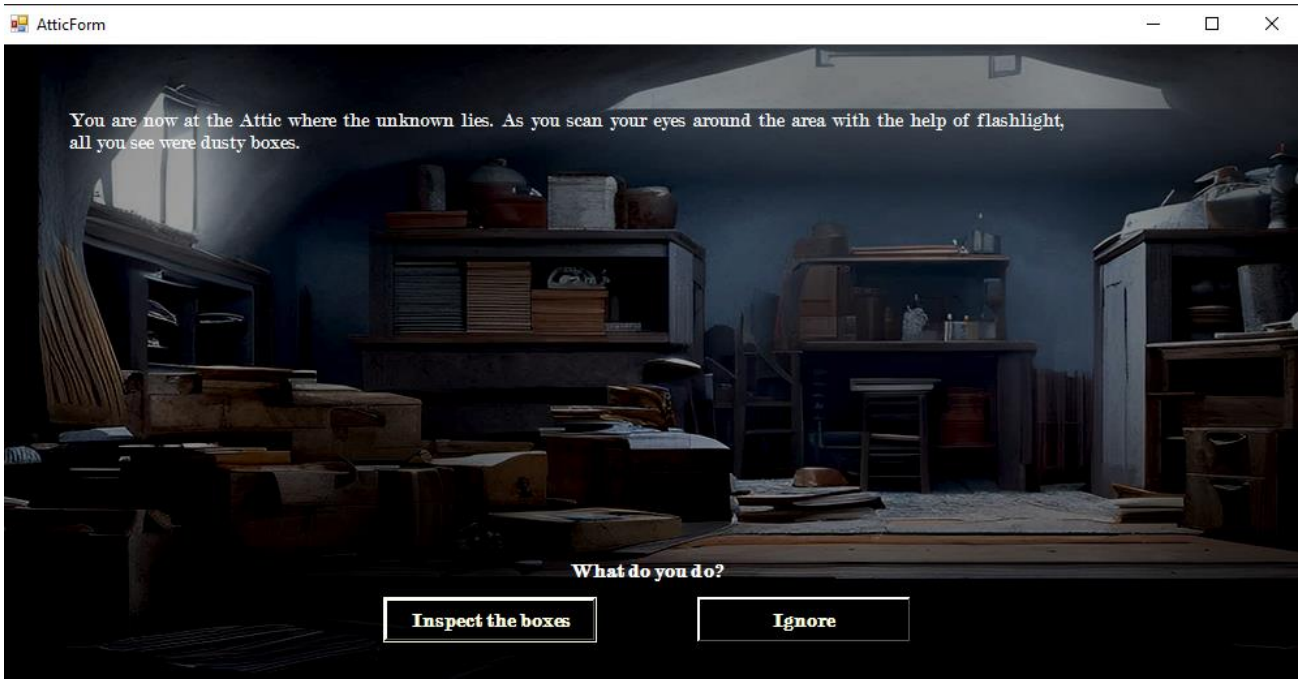


Figure 5. Attic

Figure 5 shows when the player was done with the mission on the second level.

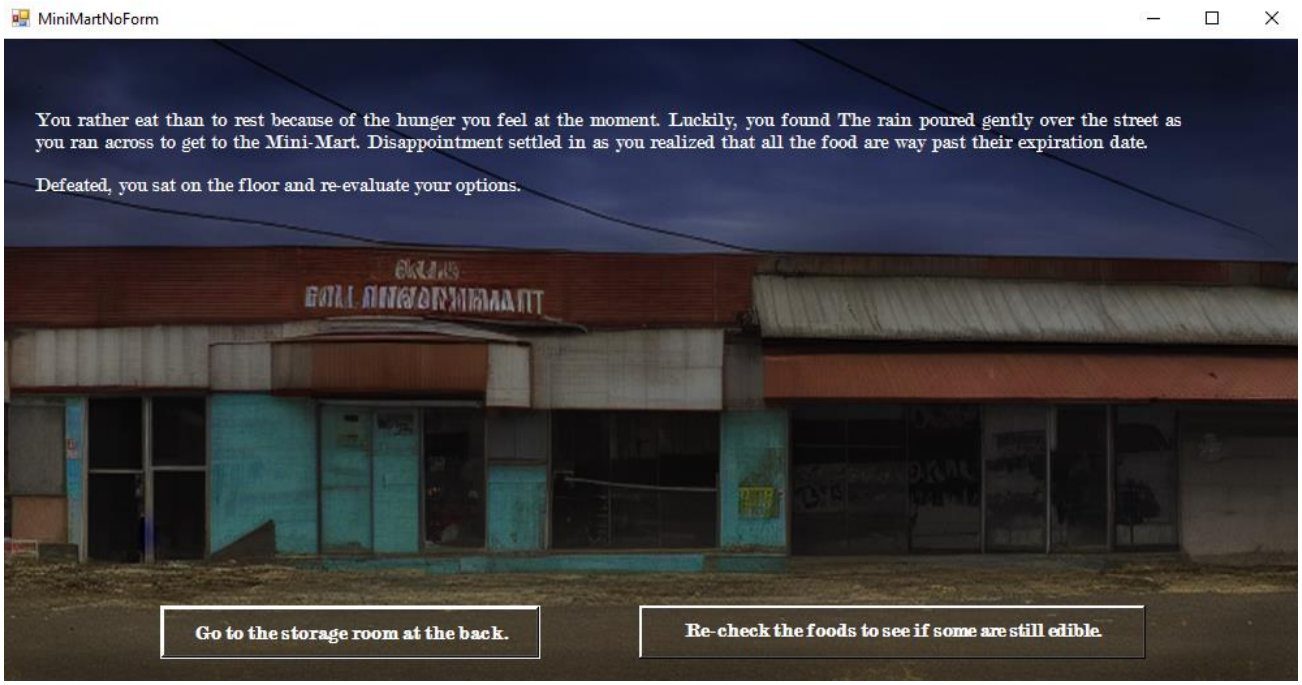


Figure 6. Mini Mart

Figure 6 shows when the player was done with the mission on the third level.

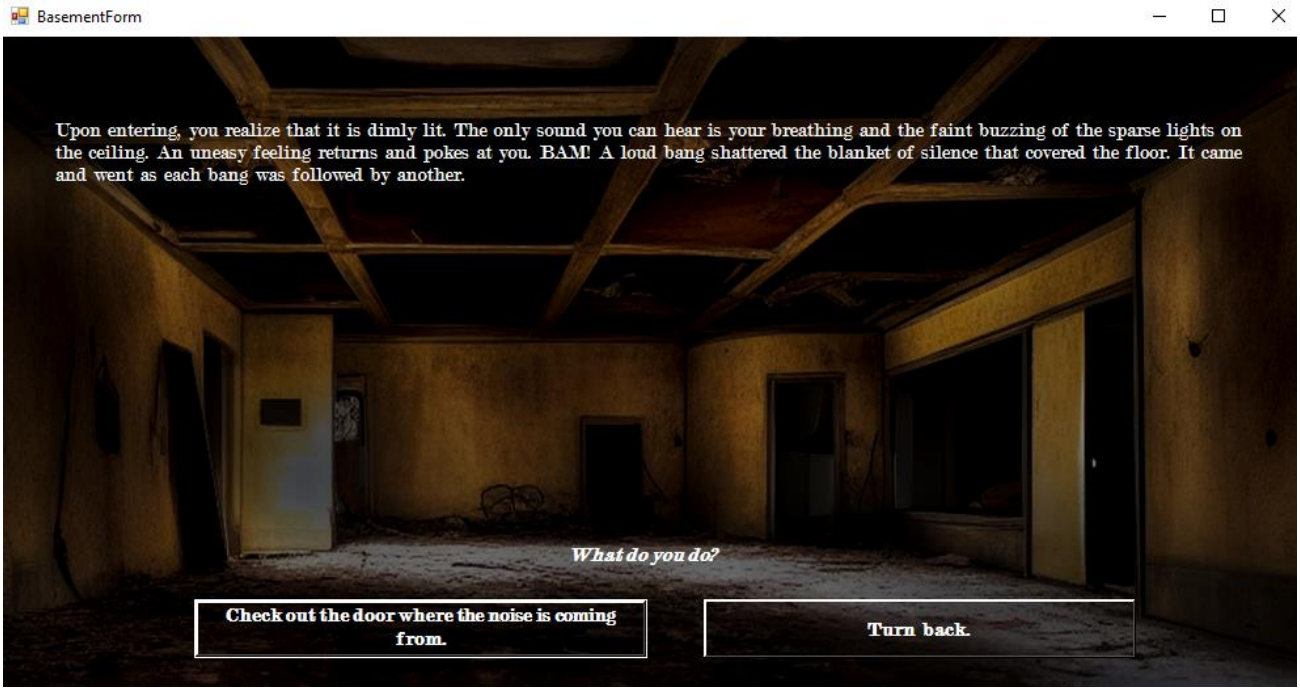


Figure 7. Basement

Figure 7 shows when the player was done with the mission on the fourth level. This will also serve as the last level of the game.

XV. SOURCE CODE

```
//HomeForm header

#pragma once

#include "AgeForm.h"
#include "AboutForm.h"

// for the buttons

#pragma endregion
	private: System::Void HomeForm_Load(System::Object^ sender,
System::EventArgs^ e) {
	}

	private: System::Void btnAbout_Click(System::Object^ sender, System::EventArgs^
e) {
		this->Hide();
		AboutForm^ aaw = gcnew AboutForm();

		aaw->Show();
```

```
        }
private: System::Void btnStart_Click(System::Object^ sender, System::EventArgs^ e) {

        AgeForm^ age = gcnew AgeForm(this);
        this->Hide();

        age->ShowDialog();
}
private: System::Void btnQuit_Click(System::Object^ sender, System::EventArgs^ e) {
        this->Close();
}
};

// For the header in AgeForm where the user will input their age

#pragma once

#include "LoadingForm.h"

// the buttons

#pragma endregion
        private: System::Void AgeForm_Load(System::Object^ sender, System::EventArgs^
e) {
        }
        private: System::Void btnBack_Click(System::Object^ sender, System::EventArgs^
e) {
                this->Hide();
                home->ShowDialog();
        }
        private: System::Void btnConfirm_Click(System::Object^ sender,
System::EventArgs^ e) {

                int age = Int32::Parse(txtAge->Text);

                if (age >= 13) {

                        this->Hide();
                        LoadingForm^ load = gcnew LoadingForm();

                        load->Show();


                }
                else {
                        MessageBox::Show("This  is not for you to play kid, quit now.", "AGE
RESTRICTION", MessageBoxButtons::OK, MessageBoxIcon::Stop);
                        this->Close();
```

```
		}
	}
};

//for the LoadingForm header

#pragma once

#include "IntroForm.h"

// for the buttons and timer

#pragma endregion
	private: System::Void LoadingForm_Load(System::Object^ sender,
System::EventArgs^ e) {

		timerLoad->Interval = 500;
		timerLoad->Start();
	}

	private: System::Void timerLoad_Tick(System::Object^ sender, System::EventArgs^
e) {
		if (loadProgBar->Value < loadProgBar->Maximum) {
			loadProgBar->PerformStep();
	}
		else {
			timerLoad->Stop();
			IntroForm^ intro = gcnew IntroForm();
			intro->Show();
			this->Hide();
		}
	}
	private: System::Void loadProgBar_Click(System::Object^ sender,
System::EventArgs^ e) {

	}

};

//for the IntroForm where the story starts (header)

#pragma once

#include "Intro1Form.h"

// for the buttons

#pragma endregion
	private: System::Void IntroForm_Load(System::Object^ sender, System::EventArgs^
```

```
e) {
        }

private: System::Void pcbTxtIntro_Click(System::Object^ sender, System::EventArgs^ e)
{
}
private: System::Void label1_Click(System::Object^ sender, System::EventArgs^ e) {
}
private: System::Void btnIntroNext_Click(System::Object^ sender, System::EventArgs^ e)
{
        this->Hide();
        Intro1Form^ next = gcnew Intro1Form();

        next->ShowDialog();
}
};

// for the PoliceStation (header)

#pragma once

#include "PSLeaveForm.h"
#include "PSLookForm.h"

// for the buttons

#pragma endregion
        private: System::Void PoliceStationForm_Load(System::Object^ sender,
System::EventArgs^ e) {
        }
        private: System::Void button2_Click(System::Object^ sender, System::EventArgs^ e)
{
                this->Hide();
                PSLeaveForm^ leave = gcnew PSLeaveForm(this);

                leave->Show();
        }
private: System::Void button1_Click(System::Object^ sender, System::EventArgs^ e) {
        this->Hide();
        PSLookForm^ look = gcnew PSLookForm();

        look->Show();

}
};

//for the TownHall (header)

#pragma once
```

```
#include "PHHallForm.h"
#include "THSurroundingForm.h"

// for the buttons

#pragma endregion
	private: System::Void TownHallForm_Load(System::Object^ sender,
System::EventArgs^ e) {
	}
	private: System::Void button1_Click(System::Object^ sender, System::EventArgs^ e)
{
		this->Hide();
		PHHallForm^ hall = gcnew PHHallForm();

		hall->Show();
	}
private: System::Void button2_Click(System::Object^ sender, System::EventArgs^ e) {
	this->Hide();
	THSurroundingForm^ sur = gcnew THSurroundingForm();

	sur->Show();
}
};

//for the Attic (header)

#pragma once

#include "ASmallBForm.h"
#include "AIgnoreForm.h"

//for the buttons

#pragma endregion
	private: System::Void AtticForm_Load(System::Object^ sender, System::EventArgs^
e) {
	}
	private: System::Void button1_Click(System::Object^ sender, System::EventArgs^ e)
{
		this->Hide();
		ASmallBForm^ smol = gcnew ASmallBForm();

		smol->Show();
	}
private: System::Void button2_Click(System::Object^ sender, System::EventArgs^ e) {
	this->Hide();
	AIgnoreForm^ aii = gcnew AIgnoreForm();
```

```
        aii->Show();
}
};

// for MiniMart (header)

#pragma once

#include "MMStorageForm.h"
#include "MMRecheckForm.h"

// for buttons

#pragma endregion
        private: System::Void MiniMartNoForm_Load(System::Object^ sender,
System::EventArgs^ e) {
        }
        private: System::Void button1_Click(System::Object^ sender, System::EventArgs^ e)
{
                this->Hide();
                MMStorageForm^ stor = gcnew MMStorageForm();

                stor->Show();

        }
private: System::Void button2_Click(System::Object^ sender, System::EventArgs^ e) {
        this->Hide();
        MMRecheckForm^ chec = gcnew MMRecheckForm();

        chec->Show();

}
};

//for Basement (header)

#pragma once

#include "BRoomForm.h"
#include "BLeaveForm.h"

//for buttons

#pragma endregion
        private: System::Void BasementForm_Load(System::Object^ sender,
System::EventArgs^ e) {
        }
        private: System::Void button1_Click(System::Object^ sender, System::EventArgs^ e)
{
```

```
            this->Hide();
            BRoomForm^ brr = gcnew BRoomForm();

            brr->Show();

        }
    private: System::Void button2_Click(System::Object^ sender, System::EventArgs^ e) {
        this->Hide();
        BLeaveForm^ leav = gcnew BLeaveForm();

        leav->Show();
    }
    };
```

XVI. CONCLUSION

Developing an interactive game using Visual Studio Windows Forms offers the developers a multitude of learning opportunities and experiences. Throughout the process, they can gain valuable insights into game development concepts, such as game loops, input handling, collision detection, and rendering. They will also learn how to design user interfaces that enhance the game experience by utilizing intuitive controls and components. Implementing game mechanics like character movement, item interactions, and puzzle-solving enables developers to understand the underlying logic and algorithms involved. Additionally, they will sharpen their problem-solving and debugging skills as they encounter and address various bugs and issues. Collaboration and teamwork become essential when working with members to coordinate efforts and integrate different components seamlessly. Thorough testing and debugging help developers refine the game, ensuring functionality, performance, and usability. Finally, game development fosters creativity and innovation, allowing developers to experiment with different mechanics, and storytelling techniques elements to make their game truly stand out. Through these experiences, developers can grow their skillset and apply their newfound knowledge to future projects, contributing to their professional growth in the field of software development.

To PROFESSOR:

Final Project Specification:

-Choose any system/game you want to create. Examples: Banking System, Grading System, Library System, etc.

-The program should include all the topics in ITC111/ITC11L.

-The system will be created by group (min of 3 and max of 4 members).

-Deadline will be during the Final Exam Week.

-Submit the Final Project files to this GDrive link: (Note to Prof: Insert Gdrive link here)

-In the GDrive, create a project folder and name it by the surnames of your group members. Your project folder should contain the: (a) Screen record/video of the running program showing the different features of your program (b) Project Documentation

Note: **The template for your project documentation is attached here.**

Note: **Only the group leader will upload the files to the project folder**