
Elephant's Memory Application

Trevor Kiggundu
December 2018

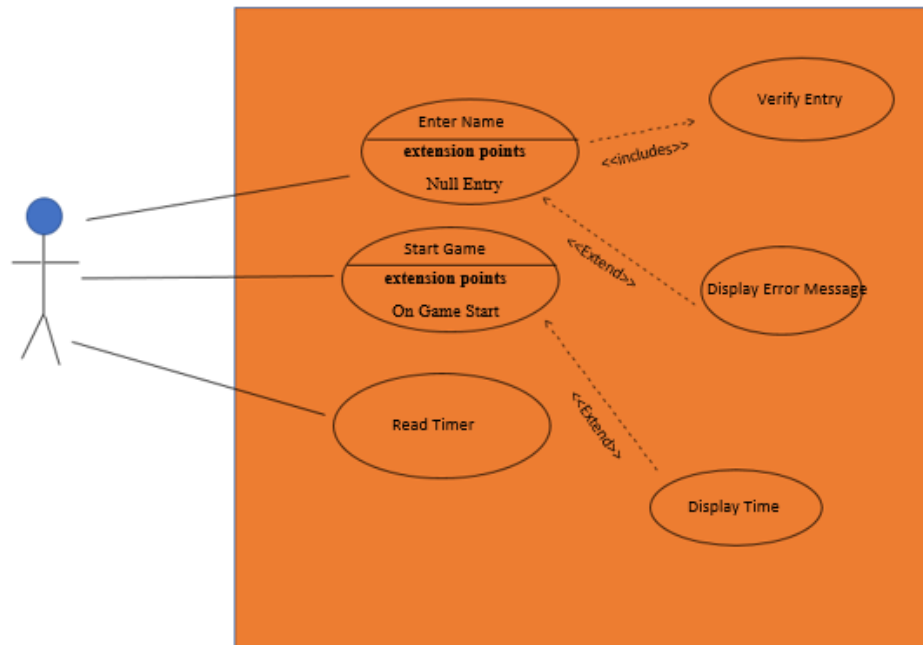
A report submitted in fulfilment of the requirements for the module Application Development,
Computing and Information Systems Department, University of Greenwich.

Table of Contents

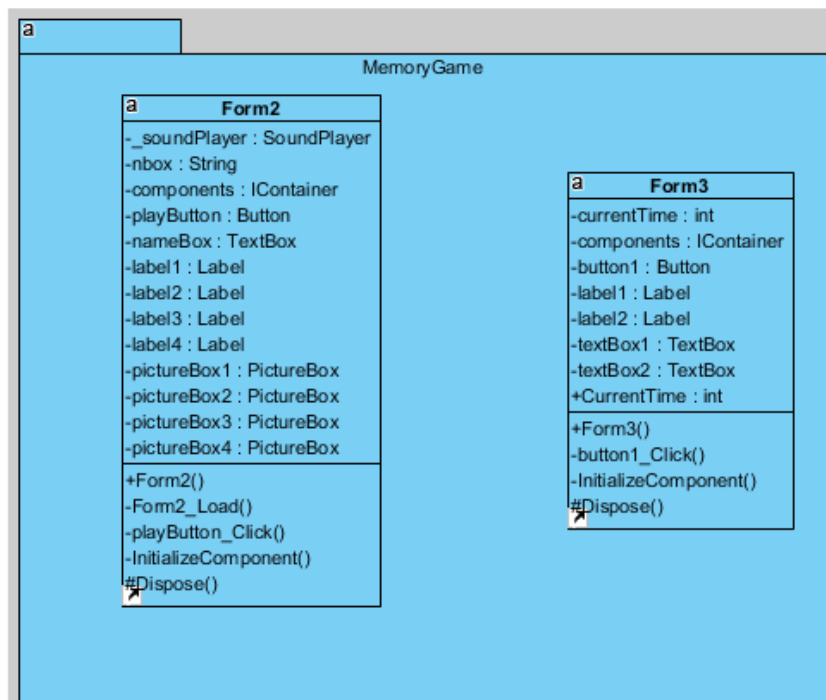
Technical documentation:	3
A list of bugs/weaknesses and/or strengths in your system:.....	7
Appendix: The source code written by you, including appropriate comments: ...	8
Simplified user documentation showing how the program works:	15
Completed self- assessment form:	19
References:.....	21

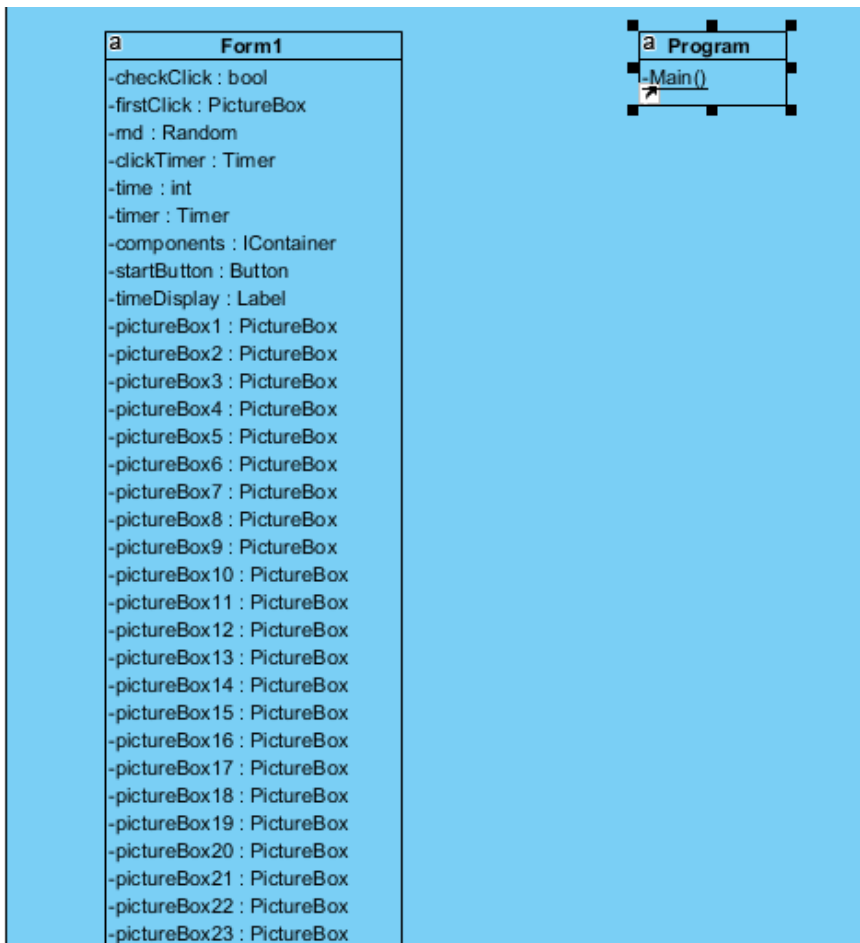
Technical documentation:

Use Case diagram:



Class Diagram:





The explanation of the design, GUI and model:

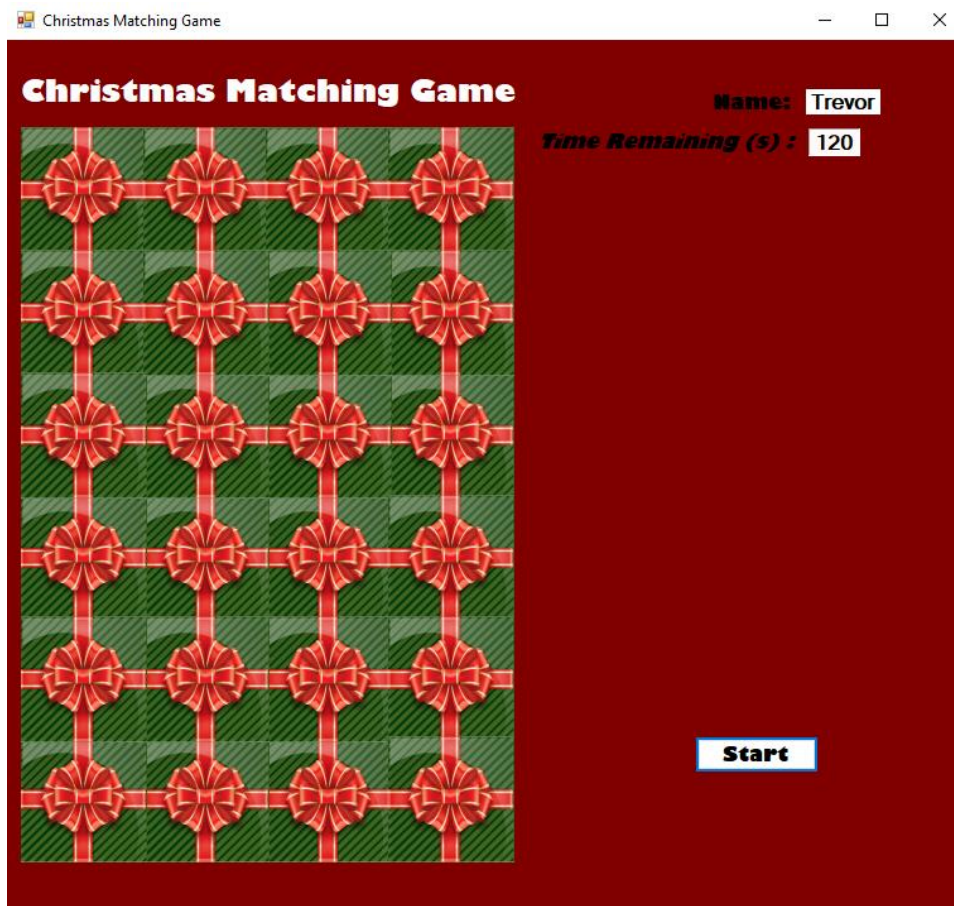
The application Visual Studio (2017) was used to create this project, with the windows forms app being the appropriate .NET selection. Different forms/tools are available to the user, making it the ideal object-oriented IDE to use as it is more hands-on than others such as Netbeans.

The goal was to make a visually pleasing yet simple GUI design. Across all forms, this consists of:

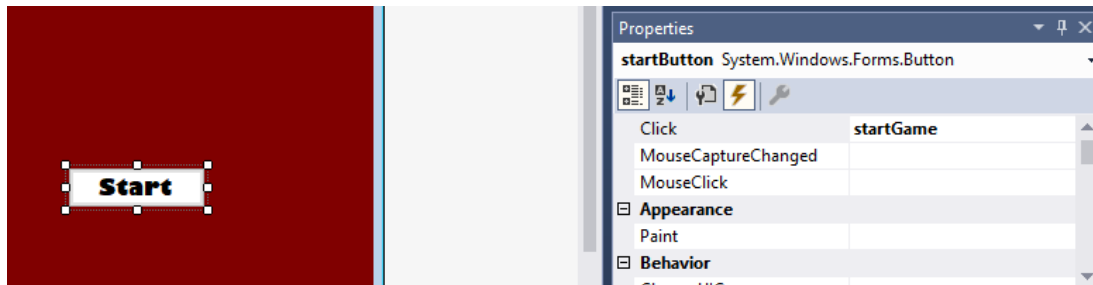
- 2 Working Forms
- 29 Picture Boxes
- 10 Labels
- 2 Message Boxes
- 3 Buttons
- 1 Text Box

Form 1:

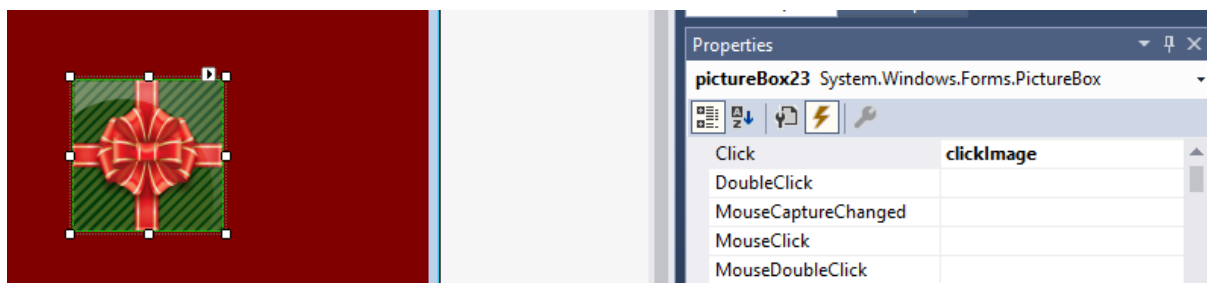
The Form is named 'Christmas Matching Game' and the background color is set to maroon.



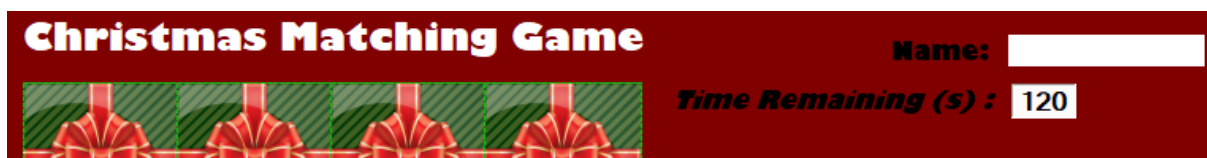
The button labeled 'start' is run by an eventhandler called 'startGame', and when invoked, shows the gift boxes on the picture boxes. However, if the background image of the individual picture boxes is set to 'giftBox.jpg', then the picture boxes will display gift boxes even before the startGame event handler is invoked.



Each picture box on this particular form is represented by a 'gift box' and these are set to run once the 'clickImage' event handler is invoked.

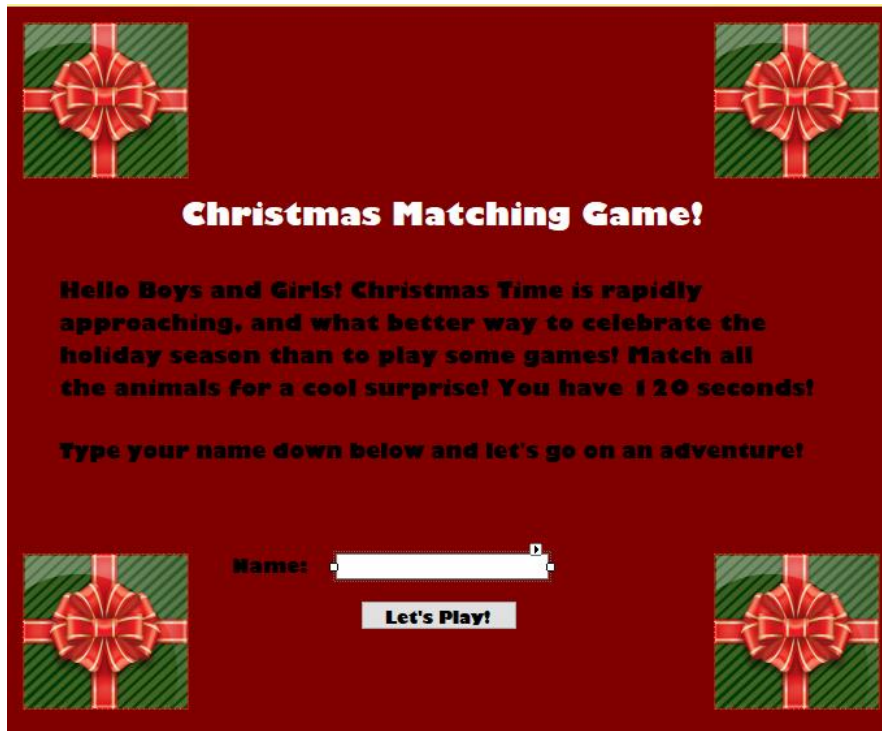


The 'Christmas Matching Game', 'Name' and 'Time Remaining (s)' only serve a purpose as labels for the user. However, their adjacent labels, which look like white text boxes, are initialized as strings and ints in Form1.cs and Form2.cs respectively.



Form 2:

The form is named 'Christmas Matching Game' and the background color is also set to maroon. The picture boxes with the 'gift box' image as the background image are merely for design and display. The 4 labels are also only for display, as they do not have any other function related to them.



Christmas Matching Game!

Hello Boys and Girls! Christmas Time is rapidly approaching, and what better way to celebrate the holiday season than to play some games! Match all the animals for a cool surprise! You have 120 seconds!

Type your name down below and let's go on an adventure!

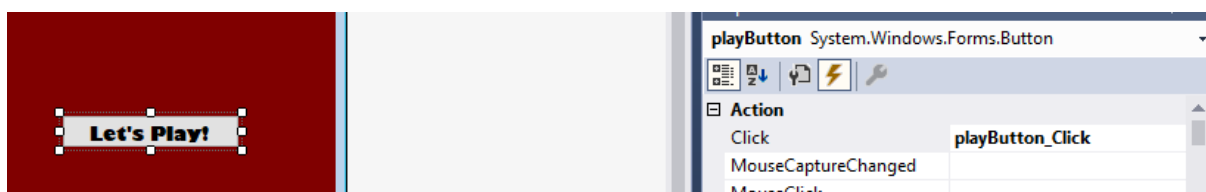
Name:

Let's Play!

The text box is linked to Form 2, and is crucial to linking the two forms together, as it is part of the invocation of the 'Let's Play Button!'.



The button labeled 'Let's Play' invokes the 'playButton_click' event handler, which converts the text in the text box into a label in Form 1. It is also the button and event handler that switches from Form 2 to Form 1 once the application is running.



A list of bugs/weaknesses and/or strengths in your system:

The design that I came up with does have its fair share of flaws. To begin with, it does not 100% comply with the application that the children's company was asking for. The application that I was able to come up with only matches images/images or words/words; never both to each other at the same time. This eliminates the educational element of the

program, as the game relies solely on the user's (child's) memory to win the game. Anybody with a decent memory can solve it, and even without that, trial and error will eventually lead to the user solving the puzzle. The application also fails to make use of a database. This is a very important feature within games development, as the user would typically want to know how well they performed the task; however, this is not the case with the matching game. Another weakness in the program is its inability to give the user a positive or negative score based on what two picture box images are selected. The only feature that does this in the game is the message box that displays after the user successfully completes the game. This is the only indicator of how efficient the user was in completing the task. The program is also lacking in adequate enough use of patterns, as well as extracurricular features, such as web services that would make the program more robust and marketable to the company. There are a lot of strengths about the application as well. The interface is very user-friendly and does take advantage of the fact that it is aimed at a younger audience. The game is Christmas themed, and through the use of threading, I am able to run Christmas music/animations in the background, keeping the user interested in the game. It also makes use of encapsulation of objects really well. This ensures that all the Visual Studio forms run smoothly and as intended, as none of the access modifiers are tampered with, something I really struggled with when I had just begun programming. A few bugs would be the music still running after the application has ended, something that I would definitely fix if given more time. I also believe that if I were to do this again, I would be much more comfortable with creating the game with all of the knowledge I have acquired from Visual Studio.

Appendix:

The source code written by you, including appropriate comments:

Form 1:

```
namespace MemoryGame
{
    public partial class Form1 : Form //method          //Global Variables declared //Reference
    the timer code bro!!!
    {
        bool checkClick = false;
        PictureBox firstClick;          //object: variaable named firstClick, instance of
        PictureBox
        // PictureBox secondClcik;

        Random rnd = new Random();      // object: instance of Random number
        generator class
        Timer clickTimer = new Timer();  // //Class Timer
        int time = 120;                  // Set value of time to 120 seconds
        Timer timer = new Timer { Interval = 1000 }; // 1000 milliseconds = 1 second
```



```

public Form1(String playerName)      // Code to link Form 1 to Form 2
{
    InitializeComponent();
    nameDisplay.Text = playerName;    // Displays the player name in Form 1
    winnerDisplay.Text = "Merry Christmas " + playerName + "!"; // Displays "Merry
Christmas" + the player name if the user wins the game
}

private PictureBox[] pictureBoxes
{
    get { return Controls.OfType<PictureBox>().ToArray(); }
}

private static IEnumerable<Image> images //Creates Array of picture Resouces
{
    get
    {
        return new Image[]
        {
            Properties.Resources.bull,
            Properties.Resources.chicken,
            Properties.Resources.cow,
            Properties.Resources.dog,
            Properties.Resources.donkey,
            Properties.Resources.duck,
            Properties.Resources.goat,
            Properties.Resources.horse,
            Properties.Resources.mouse,
            Properties.Resources.pig,
            Properties.Resources.rabbit,
            Properties.Resources.sheep,

            //Properties.Resources.bull1,
            //Properties.Resources.chicken1,
            //Properties.Resources.cow1,
            //Properties.Resources.dog1,
            //Properties.Resources.donkey1,
            //Properties.Resources.duck1,
            //Properties.Resources.goat1,
            //Properties.Resources.horse1,
            //Properties.Resources.mouse1,

```

```
//Properties.Resources.pig1,  
//Properties.Resources.rabbit1,  
//Properties.Resources.sheep1  
  
//Properties.Resources.bull,  
//Properties.Resources.bull1,  
  
//Properties.Resources.chicken,  
//Properties.Resources.chicken1,  
  
//Properties.Resources.cow,  
//Properties.Resources.cow1,  
  
//Properties.Resources.dog,  
//Properties.Resources.dog1,  
  
//Properties.Resources.donkey,  
//Properties.Resources.donkey1,  
  
//Properties.Resources.duck,  
//Properties.Resources.duck1,  
  
//Properties.Resources.goat,  
//Properties.Resources.goat1,  
  
//Properties.Resources.horse,  
//Properties.Resources.horse1,  
  
//Properties.Resources.mouse,  
//Properties.Resources.mouse1,  
  
//Properties.Resources.pig,  
//Properties.Resources.pig1,  
  
//Properties.Resources.rabbit,  
//Properties.Resources.rabbit1,  
  
//Properties.Resources.sheep,  
//Properties.Resources.sheep1,  
  
};  
}  
}
```

```

private void startTimer()                                //Reference the Timer Code bro!!
{
    timer.Start();
    timer.Tick += delegate
    {
        time--;
        if (time < 0)
        {
            timer.Stop();
            MessageBox.Show("Sorry, You're out of Time :( Try Again?");
            BlankImages();
        }
        var ssTime = TimeSpan.FromSeconds(time);
        timeDisplay.Text = time.ToString();    // "+s" if I want to show it in seconds
    };
}

private void CoverImages() //GiftBox
{
    foreach (var pic in pictureBoxes)
    {
        pic.Image = Properties.Resources.giftBox; //Makes the cover image the giftbox
    }
}

private PictureBox assignAnimal()
{
    int num; //declares integer num
    do //loops through all picture boxes
    {
        num = rnd.Next(0, pictureBoxes.Count()); // Randomizes number of images
        between 0-24
    }
    while (pictureBoxes[num].Tag != null);
    return pictureBoxes[num]; // checks if null, then returns # of free boxes
}

private void setAnimalImages()
{
    foreach (var image in images)
    {
        assignAnimal().Tag = image;                //Assigns random animal image to the
        first 12 boxes
    }
}

```

```
        assignAnimal().Tag = image;           // Assigns random corresponding animal
images to second set of 12 boxes
```

```
    }
}
```

```
private void BlankImages() //Sets the picture boxes back to "giftBox" when user
restarts the game
```

```
{
    foreach (var pic in pictureBoxes)
    {
        pic.Tag = null;
        pic.Visible = true;
    }
    CoverImages();
    setAnimalImages();
    time = 120;
    timer.Start();
}
```

```
private void beginTimer (object sender, EventArgs e)
```

```
{
    CoverImages(); //shows giftbox if background image isn't set giftbox
    checkClick = true;
    clickTimer.Stop(); //freezes timer
}
```

```
private void clickImage(object sender, EventArgs e) //overloaded method
```

```
{
    if (!checkClick) return; //if nothing is clicked, dont do anything
    var pic = (PictureBox)sender;

    if (firstClick == null) //clicks on first image
    {
        firstClick = pic;
        pic.Image = (Image)pic.Tag; //Turn Tag back into an image

        return;
    }
}
```

```
pic.Image = (Image)pic.Tag; //matches tag and image
```

```

        if (pic.Image == firstClick.Image && pic != firstClick) // If images match when
clicked...
        {

            {
                pic.Visible = firstClick.Visible = false; //both picture boxes become invisible
                {
                    firstClick = pic;
                }
                CoverImages();

            }
        }

        else
        {
            checkClick = false;
            clickTimer.Start(); //keep running timer
        }

        firstClick = null;
        if (pictureBoxes.Any(p => p.Visible)) return; //If no more pictureboxes are visible....
        timer.Stop(); //stop time
        MessageBox.Show("You Win! You did it with: " + timeDisplay.Text + " seconds left.
Now Try Again!"); //display
        BlankImages();

        //this.Close();
        //Form3 f3 = new Form3();
        //f3.Show();
        //f3.CurrentTime = nameDisplay;

    }

    private void startGame(object sender, EventArgs e)
    {
        checkClick = true;
        setAnimalImages(); // runs random images
        CoverImages(); // covers random images with "giftBox"
        startTime(); //timer starter
        clickTimer.Interval = 1000; // sets timer interval to 1000 milliseconds
    }

```

```

        clickTimer.Tick += beginTimer; // begins timer when "startButton" is
pressed //Reference this
        startButton.Enabled = false;
    }

}
}

```

Form 2:

```

namespace MemoryGame
{
    public partial class Form2 : Form
    {
        private SoundPlayer _soundPlayer;
        String nbox;

        public Form2()
        {
            InitializeComponent();
            _soundPlayer = new SoundPlayer("christmasmusic.wav"); //plays specific audio file
that is asked for

        }

        private void Form2_Load(object sender, EventArgs e)
        {
            _soundPlayer.Play();           // starts audio

        }

        private void playButton_Click(object sender, EventArgs e)
        {
            nbox = nameBox.Text;
            Form1 newFormTwo = new Form1(nbox); //After nbox is clciked in Form 2, Form 1
starts up
            newFormTwo.Show();
            this.Hide();

            if (nameBox == null) // Makes sure a user name is actually entered

```

```

{
    MessageBox.Show("Please enter your name");

    return;
}
}
}
}

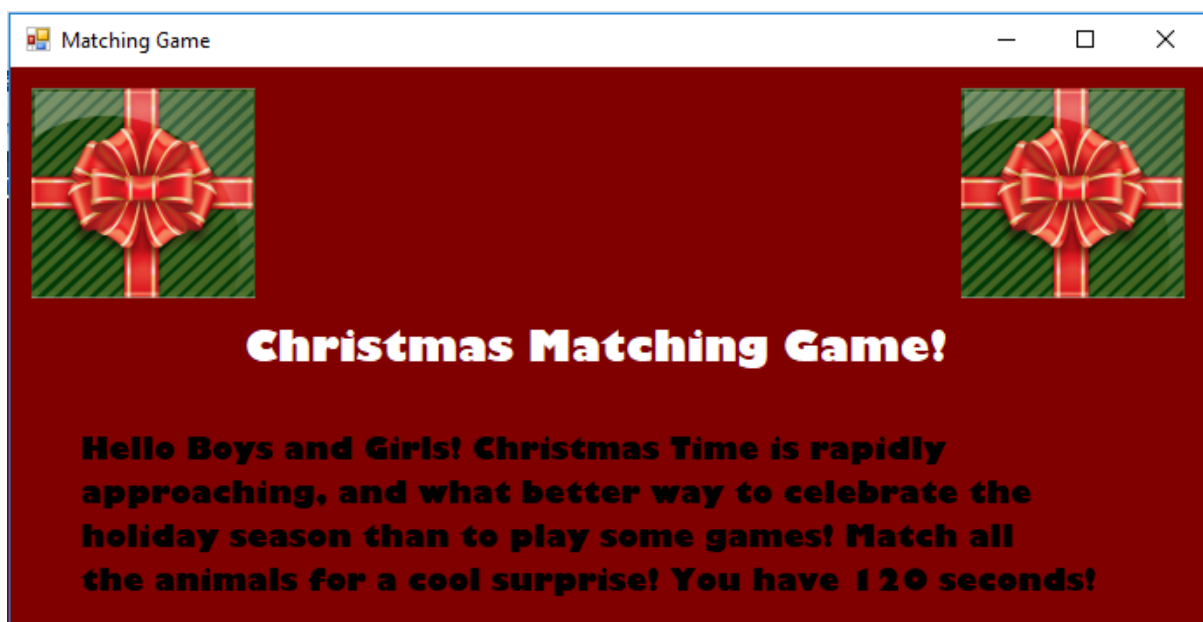
```

Form 3: Incomplete, Database creation attempt.

Code is referenced and will not be included in source code.

Simplified user documentation showing how the program works:

Upon first opening the program, the user will find a maroon window decorated with 4 gift boxes, titled 'Christmas Matching Game!'. The game instructions are below the title as well. The user will also hear Christmas music when they open the program.



The instructions will also prompt the user to enter their name into the text box provided and press enter. For the sake of this demonstration, the name 'Trevor' has been entered.

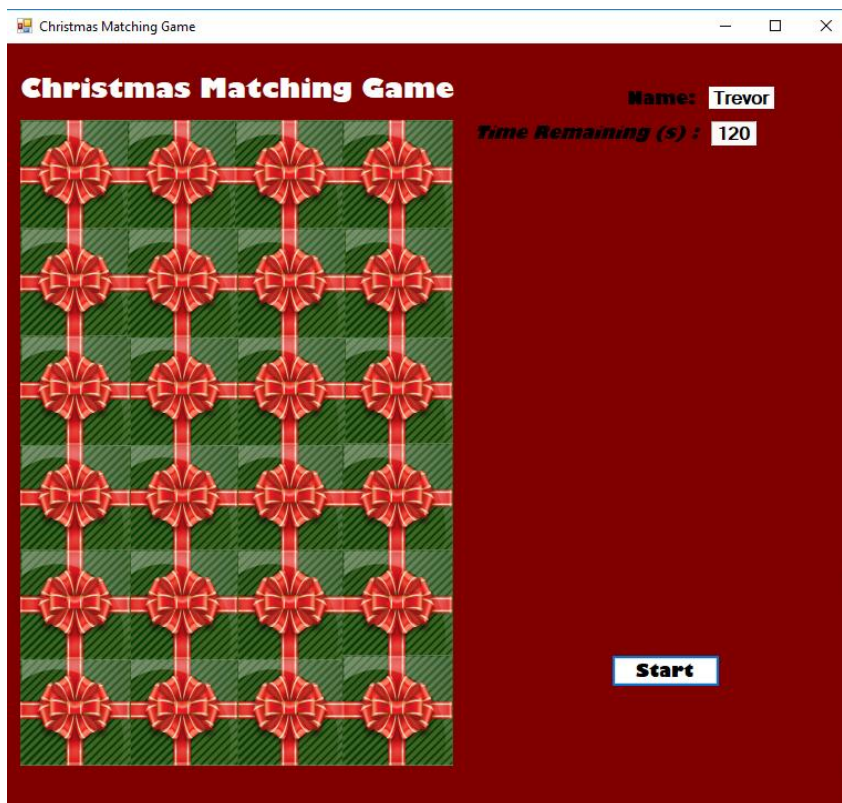
Type your name down below and let's go on an adventure!

**Name:**

Let's Play!



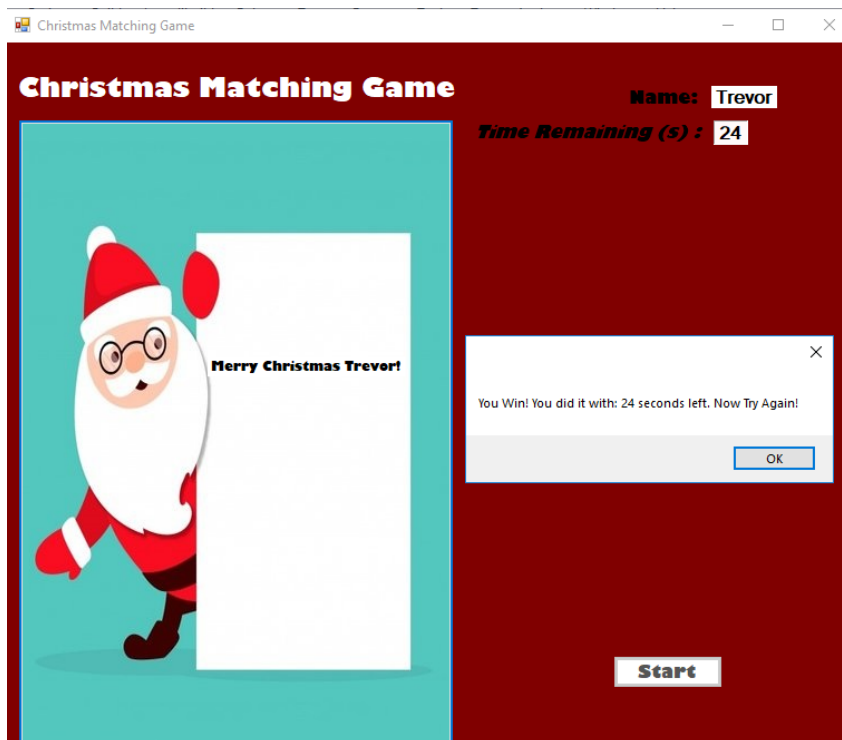
When the user clicks the 'Let's Play' button, they will be taken to the second screen, where the game actually runs. Notice that user's name is shown in the top right corner. The amount of time left is also shown in that corner.



Clicking the 'Start' button will start the game and decrement the timer every second. Successfully matching a pair of images removes those images from the grid.



Successfully completing the game results in the user seeing the 'surprise' that was promised earlier, a Christmas message from Santa himself. A message box also pops up informing the user of how much time it took to complete the game.



Failing to finish the game will result in a pop-up box letting the user know that they have run out of time.



Completed self- assessment form:

Based on the criteria provided, I have decided to award myself 65/100 marks. Shown below is the completed self-assessment form.

NOTE: You must include comments for each section of the self-assessment sheet

Student Name: Trevor Kiggundu

	%	P	A	S	G	E	O	Comments
Object Oriented Design (including UML diagrams and how the design is reflected into code and any other technical documentation included in the report)	15				A			Done well, and referenced in the report, could be better though. 10 marks
Use of desired features and concepts								
• Well named variables and, methods. Decisions, Iteration. Overloading	10				A			Variables , methods and decisions are all defined clearly. Iterate statements are easy to understand, and overloading is used perfectly. 8 marks
• Objects and Classes. Collections	10				A			Done very well. PictureBox and random are objects are instances of their own classes. Other classes include the timer and random number generator. 10 marks
• Events controls. Validation	10					A		Implemented into program well, event handlers are well defined and validated. ex: -startGame: linked to start button -clickImage: linked to picture boxes -beginTimer: Linked to start button and timer. 8 marks
• Threads. Animations.	10				A			Asynchronous tasks running at the same time as the selected form. ex: Christmas music running in the background while the game is being played. 7 marks

<ul style="list-style-type: none"> Inheritance. Interfaces 	10		A						Use of encapsulation, making sure objects in the program are not misused, such as the access modifiers: public and private. could be better. 3 marks
<ul style="list-style-type: none"> Persistence – storing and reading data from a db 	10		A						Attempted, but not adequate enough. 2 marks
<ul style="list-style-type: none"> Patterns 	5		A						No attempt
<ul style="list-style-type: none"> Extra features (such using web services) 	5		A						No attempt
Acceptance Testing (includes how well the app is presented and the ability to answer technical questions)	10						A		Apart for some flaws regarding the specification, the program runs well, the user is also able to answer questions about the system. 10 marks
Accurate Self- Assessment	5							A	Accurate. 3 Marks
Possible AO?									

References:

Anon (n.d.) Want a button to disable for 30 second after click and enable it automatically, *Stack Overflow*, [online] Available at: <https://stackoverflow.com/questions/20717982/want-a-button-to-disable-for-30-second-after-click-and-enable-it-automatically> (Accessed December 9, 2018).

dotnet-bot (n.d.) Timer.Enabled Property (System.Windows.Forms), *Microsoft Docs*, Microsoft, [online] Available at: <https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.timer.enabled?view=netframework-4.7.2> (Accessed December 9, 2018).

HD, iG. (2014) How to add background music to C# Form, *YouTube*, YouTube, [online] Available at: https://www.youtube.com/watch?v=pRKie_jqfBA (Accessed December 9, 2018).

Macrovector (n.d.) *Farm animals livestock and pets icons set isolated vector illustration*, *Freepik.com*, Freepik, [online] Available at: https://www.freepik.com/free-vector/farm-animals-livestock-and-pets-icons-set-isolated-vector-illustration_1159300.htm.

Rpetrusha (n.d.) Timer Class (System.Timers), *Microsoft Docs*, Microsoft, [online] Available at: <https://docs.microsoft.com/en-us/dotnet/api/system.timers.timer?view=netframework-4.7.2> (Accessed December 9, 2018).

Veaceslav Draganov_ Background music for video (2017) Christmas Background Music Instrumental for Videos / Slideshow | Royalty Free, *YouTube*, YouTube, [online] Available at: <https://www.youtube.com/watch?v=k6nZxLvEcdw> (Accessed December 9, 2018).

Zasimova, A. (n.d.) *Gift Vectors, Royalty Free Vectors*, Vector Stock, [online] Available at: <https://www.vectorstock.com/royalty-free-vector/green-gift-box-with-a-red-bow-top-view-vector-20286304> (Accessed December 9, 2018).