A-Level Computer Science Project

Virtual-Life

Thomas Boxall

March 2022

BEXHILL COLLEGE: 56635
CANDIDATE NO: 5377

Contents

Chapter 1

Analysis

1.1 Introduction

My project will be a prototype life simulator game designed for Windows devices.
The project will be defined as a prototype as the scope of life simulator games is extremely broad; it will not be feasible to construct a well-rounded game in the time frame set out for this project. Due to the nature of a prototype, features may be incomplete or missing altogether. In my evaluation, I will discuss what features are incomplete or missing and ways in which they could be completed.
The idea for this project came from playing life simulator games and wanting to improve on them by adding a competitive aspect. This will be a single score which actions in the game will determine. The aim of the game will be to get the highest score possible. I want to design and implement this game for my project because I think it will be fun as well as a challenge; I am also interested in the balancing process which I will have to go through to make my game fair for players.

1.2 Computational methods my solution will contain

This game will be amenable to being solved using a computer program because games based off of random events are complex to develop on paper and often involve a second person to act as a 'game master', the use of a computer mitigates the need for a second person.

1.2.1 Abstraction

Real life is extremely complex and has lots of different aspects to it. This game will not have every aspect of real life programmed, so I will need to use abstraction to remove the inessential aspects of life. I will also need to use abstraction to hide some of the complex algorithms from the user, including the score calculation algorithms.
An area in which I will definitely be abstracting in, is in game items. The game will not have a shop or in game wardrobe, it will just be assumed that the character has clothes. A future iteration of the game may include this.

1.2.2 Decomposition

I will need to use decomposition to break down the elements of life which I have chosen to implement into small subroutines. This will maximise efficiency as I will be able to use the same well-designed subroutine many times, minimising the number of lines of code which I have to write. By using a number of subroutines, it will mean that my program can be developed further and maintained easier after I have completed this project.

1.2.3 Real time processing

Most of this game will be fundamentally based on real-time processing. Basically everything which happens will be run in real-time as the player clicks through the game. This will need to happen

quickly as it is unpleasant and off putting to use software or games where there are increased wait times for seemingly simple things.
The only exception to this might be the data storage, I'm not sure exactly how I will do this, it may be run parallel to the main game code.

1.2.4 Calculations

There will be a number of time calculations are used in the game. Primarily, they will be used to run the score system, calculating the scores in real time and modifying them as and when a user changes something in the game. Calculations will also be important in deciding probabilities for certain things to happen in the game; for example, there will be an algorithm which takes the character's age as an input and returns the chance of them dying. This chance will be based off of their age and a pre-determined modifier.

1.2.5 Data storage

The data generated from each round of the game will need to be able to be saved to an external (to the game) location; for example, the C drive of the computer which the game is being run on. This will enable the player to pause their game and resume at a later time or for them to share a life with their friends over a communication platform, then their friends can install the life and play through it. The game will need to have a way for the data to be re-loaded.

1.3 Stakeholders

My target audience for this game will be casual gamers, aged between 16 and 25. Generally, I have found that this is the target audience for similar games available for different platforms. This is a broad age range, covering the upper end of the teenage market and young adulthood. I have chosen this market because I think that the content of the game will be applicable and enjoyable to those who are starting out in life (teenagers) and those who have made a start in life but could stop and pick up a different track if they wanted to (young adults).
The game will be suitable to the lifestyles of teenagers as it will be designed to play in short bursts, perhaps whilst they are waiting to meet up with friends or having a break from school work. It will also be engaging for teenagers with the competitive aspect allowing them to play together to get the best score at the end of a life, sharing a screenshot from the end of the game with their peers over a communication method. There won't be any unlockable aspects, apart from character age related content in game, so if the app has to be deleted from a device, there will be no worry about losing progress after a life is completed.
The game will be suitable to young adults because it will be able to be played however they want, either making use of all the features, or playing through very simply and quickly, not making use of the features which are off the main path. This will be suitable because they might want to play the game more competitively than the teenage audience, working out the optimum way to live a life to get the best score possible, beating colleagues or friends.
All users will interact with the game in the same basic way. Primarily this will be through buttons on the screens, however there will be occasional keyboard input required. The use of buttons will mean this game is suitable for those using eye tracker software or those who use touch input rather than a keyboard. The game, however, will be primarily designed for interaction using a mouse and keyboard.

1.4 Research

I had initially planned to make my game for Android devices, which is no longer possible due to software restrictions. This is why my research is aimed primarily towards mobile apps. The majority of this research will translate between Android and Windows naturally, with only very minor differences explored in the design chapter.

To conduct my research for my project, I will be creating a questionnaire to send to people and researching different currently available options myself.

1.4.1 Questionnaire

I will create a Google form to send out to friends and classmates who fit into my target audience and have played games like this in the past.

Questions

I have split my questions into two categories.

1. Previous experiences
   (a) What device(s) have you used to play Life Simulator games?
   (b) What life simulator games have you played?
   (c) Roughly, when did you last play this game?
   (d) On average, how long do you think you played these games for?
   (e) What made you stop playing these games?
   (f) When you have played Life simulator games, what did you like about the game? Any specific features?
   (g) When you have played Life simulator games, what did you not like about the game? Any specific features?
   (h) Were there any parts of the games which you found more interesting than others? If so, what were they?
2. User Interface
   (a) Were there any parts of the user interface which you found hard to use? What were they and what was hard to use?
   (b) Were there any parts of the user interface which you found easy/intuitive to use? What were they and what was hard to use?
   (c) Would you prefer a light or dark theme for the app?

Responses to Questionnaire

I sent the questionnaire out to the 1st year Computer Science cohort, all of whom fit into my target audience. Listed below are the results of the questionnaire.

1. (a)
   - 33.3% - Android phone
   - 66.7% - iPhone

   (b)
   - BitLife (4)
   - The Sims (2)
   - Unnamed (1)

   (c)
   - 6 months ago
   - 2 months ago
   - May 2021
   - Last year.
   - 3 years ago
   - Maybe three years ago, 2018

   (d)
   - A fortnight
   - A couple of months

- 20 mins per week
- 4 hours
- Not very much
- 30 mins at a time

(e)
- Lost interest as it started to get repetitive
- Glitches in software and ads
- Boredom/Repetitive
- I didn't find them very interesting.
- The time needed for waiting
- Boredom

(f)
- Entertaining dialogue with some of the random simulations
- I liked being able to personalise decisions to fit my life and using it to predict my future
- Customisation
- I liked customising characters in the sims.
- Different types of options available
- How it can be a different experience each time you play, the variation

(g)
- After a while it gets repetitive, also a lot of the the more interesting/fun features were locked behind a paywall
- I didn't like when they give you options and you have to pick one of them because not all of them apply so you can't always pick something you actually want to pick
- Repetitive
- The game itself doesn't really appeal to me because I didn't always have to interact with it so I spent a lot of time idle while waiting for something to happen.
- The camera [unsure what this is referencing]

(h)
- The minigames for things like escaping jail etc
- I liked being able to see that I had to work on relationships with people (maintain friendships etc).
- More detailed
- Character customisation.
- The building aspect
- Not really

2. (a)
- Not really
- I don't understand what user interface means
- None hard to use
- (Bearing in mind I haven't played in ages) Some of the icons weren't obvious to me so I had to click around a lot to find things.
- Hard - free look mode
- Not really

(b)
- Most of the UI was fairly obvious as to what it did.
- The Buttons were simple
- Icons that are easily recognisable (cog for settings, house for home/menu). The buttons had a HUD so they were easy to find.
- Easy - guiding character
- Some of the larger buttons that where perfect for the size of my phone

(c)
- 66.7% - Dark
- 33.3% - Light

Reviewing Responses To The Questionnaire

1d Time played massively varied, this could be because different people have different attention spans or they could have explored different parts of the game. The time I am attracting people to my game for is not a massive concern, however it might be an interesting to engage in a focus group with those who only played for a short amount of time to understand their answer further.

1e It seems these responses fit into three categories: bored of playing the game; bored of waiting for things to unlock (which will allow them to progress further) and software issues, whether this be an intentional pause, paywall or glitches which made users stop playing. Ads also made the game less appealing to players, however I won't be commercialising my game so this won't be a problem.

1f The majority of these responses suggested that the variety and randomness of the games made them appealing to play, this is something I intend on recreating in my game

1g These responses are split into two, some not liking the repetitiveness of games like this which comes naturally when you are playing lots of lives consecutively as there are only a few routes which you can go down in a virtual life; and the second responses generally dislike a paywall feature which I wouldn't be implementing anyway.

2a One person commented that they found the UI hard to use, I intend to use standard icons which are well used and recognised throughout the app market to counteract this, as well as use a simple icon theme to keep my app looking tidy.

2b Most people found the UI intuitive to use, I intend to replicate this simplicity in my own way as I think that is a very key element to having a successful app.

2c I will be making this app with a dark theme as a default.

1.4.2 Similar Games currently available

Before I begin to design my application, I need to look at a number of other games which are available on the app market. This will help me to understand further some of the feedback which I got from my questionnaire.

BitLife

BitLife is currently available on the Google Play Store.
The description of the app on the Google Play store outlines some features which the app has, in a handwavey way. From playing the game, I have learnt that it allows the user to age up. When the user does this it tells them about something which has happened, for example they started at a school. Then it might prompt the user as the game goes on to pick different aspects of the life, or make decisions for the character, for example, if to kiss someone or not, if to go on a date or not. I have also learnt that many of the features of the game are only accessible after the user ages up a certain amount, this is beneficial as it keeps more to the real world as a 10 year old can't buy a house. Furthermore, there are many other features which are accessible via a menu. There is an element of morality to the game, for example when I was playing it, I was trying to make my character go on a date with many different people even though they were married, the game prevented this every time. This game also allows you to carry on playing as your child when your character dies. Throughout the life of the character, there are events which happen - for example, a family member falls ill or dies. Another feature of the game which I like is the finance management system; part of the game is having a job which earns the character money. With that money, you are then able to purchase things - for example a house or speedboat.
Reading the reviews of the game suggests to me that many people are happy with the game, however some suggest that there are elements which need to be improved on for the game to be more fun, for example, someone would like there to be a one month age up option so the life of the virtual character

can be explored more in depth. The app contains in-game purchases which allow ads to be removed entirely from the game.
The user interface of the game is relatively messy, with adverts covering parts of the game on my phone as many buttons are placed along the bottom edge of the screen. Also, lots of the menus are very small and difficult to touch accurately

Figure 1.1: Screenshot of BitLife promotional images on the Google Play Store

Life Simulator 3

Life Simulator 3 is currently available on the Google Play store.
The description of the app on the Google play store details the many elements which the game has. The main difference between this game and BitLife, is that this game allows users to create a virtual avatar whereas BitLife doesn't. Reading the reviews, I have learnt that there are many programmatical errors which make the game hard to play, including: when a house is sold you become "homeless" for a short while until a new house is bought, a user found that they died each time they became homeless as their health deteriorated too quickly. Judging by the images on the Google Play store page, the user interface looks very touchscreen friendly, with big buttons and large menus where the entirety of a box is touch friendly. The images look as though the app defaults to a dark theme.

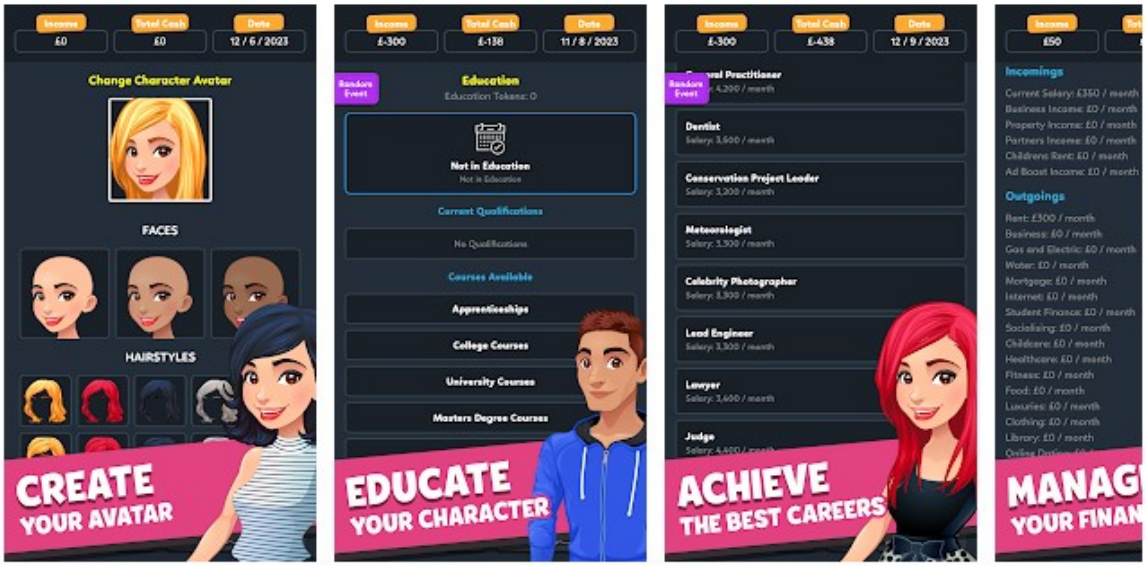


Figure 1.2: Screenshot of Life Simulator 3 promotional images on the Google Play Store

Life Simulator - Realistic Life Simulation Game

Life Simulator - Realistic Life Simulation Game is currently available on the Google Play Store.
The description of the app on the Google Play Store lists many elements which are available within the app. This app seems more modern and 'with the current times' than the others which I have looked at, containing features such as having a side hustle and a main job. It also has scenarios which set the player up to follow a specific track and achievements which can be unlocked to allow the player to do more within the app.
From playing this game, I have learnt that a tutorial or a clear how to play button is beneficial, which this game lacks. I spent the first 5 minutes or so trying to work out how to play the game, before stumbling across the button which makes the character older (it's a work button, not a dedicated age up button). When you do press this button, it ages you up weekly, this is shown by a tiny little dot at the top of the screen, which is not very accessible for those with a visual impairment. Also in this status bar are 3 attributes, which you have to maintain at a good level to stay alive. This is relatively challenging because to increase them, you have to spend money and when you work for money, they decrease. This is a good thing in my opinion. An improvement to this attribute display would be a percent indicator because each action has a different effect and it is hard to judge what the values in the bar are, making it hard to judge what modifier you need to apply to raise it just enough without exceeding its max value, thus wasting money. Quite a nice feature of this game, is that it doesn't force you to follow a certain plot, even though the free version of the game forces you to pick a scenario to start the game in.
Reading the reviews, it seems that generally, users are happy with the game, however there are issues with the balancing of the pricing within the game.
The user interface of the game looks clean, with a consistent colour theme throughout the pages and greying out & displaying a locked padlock icon for invalid buttons. It looks as though large buttons are used.

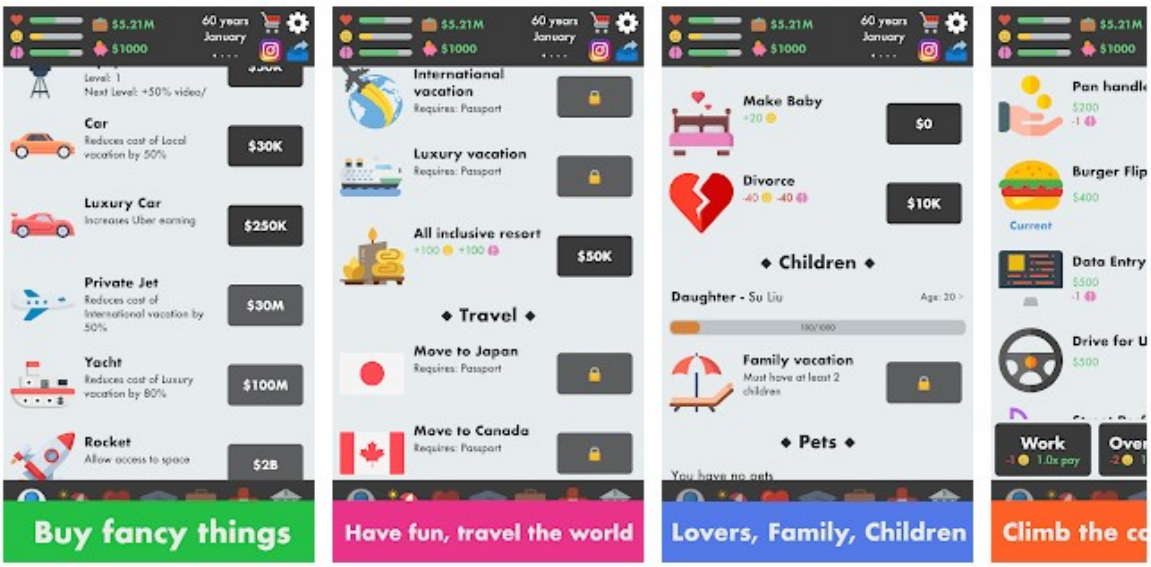


Figure 1.3: Screenshot of Life Simulator - Realistic Life Simulation Game promotional images on the Google Play Store

Reviewing Findings from Different Games

Reviewing these similar games, it is clear that there are a few basic themes which are a constant throughout this genre of game:

- Being able to make the character older
- Partners & family
- Jobs
- Finance management
- Education

Through reviewing the games, it is clear to me there isn't a competitive aspect to any of them. I plan on adding on to the game, through introducing a "life score"; the aim of the game is to get this score as high as possible. There will be various factors which will impact on the score.
I will strive to include all of these in my final program, as well as the following which were less common amongst my research, but sound like very good ideas:

- Food and diet - allowing the user to set what diet they choose for their character to dine on, this may have a financial implication if you choose for your character to eat at a hotel every day.
- Side Hustle - a second job which over time could progress to being the characters full time job
- Travelling and holidays - being able to use money earned from working to go on holiday, which improves characters attributes and opens up opportunities of finding partners with different nationalities.

There are also some elements which I really liked the look of, but will most definitely not be able to program in the time allocated for this project.

1.5 Essential Features Of The Game

1.5.1 Success Criteria

1. Ageing Up

   (a) The game has a button which "ages up" the character
   (b) As part of this process, a random event will occur (eg, family member dies, your partner wins something, you are pregnant)
   (c) Some events will happen at set times (eg. finishing school)

2. Relationships
   (a) Having a relationship with parents
   (b) Having partners
   (c) Marrying partners
   (d) Divorcing partners
   (e) Being able to pick the characters sexuality
   (f) Having kids with your partner (adopted/biological)

3. Crime
   (a) Ability to commit crime - this will be a button which will randomly generate a crime, then deduct a set amount from the lifescore.

4. Employment
   (a) Get jobs
   (b) Quit jobs
   (c) The longer you stay in a job, the more `jobPoints` a character will get each year.

5. Character Picker
   (a) At the start of each life, the user will be able to pick the character they want to play the game as. This will generate a brief backstory to the character which will crop up throughout the game (deaths of family members, key events in siblings' lives)

6. Education
   (a) There will be a random education element to the game which will allow players to progress through school (which will have randomly generated names). Every time they complete a course or school, they get `educationPoints`.

7. Saving
   (a) The game should be able to write the data out to a file(s) which can be loaded in during runtime to resume the game. These do not need to be readable by the user.
   (b) There should be a way for the user to export the life from the game at the end of the game to a non-editable file format (eg, PDF).

8. Scores
   (a) There should be a number of scores which the actions of the player can affect throughout the game.

1.5.2 Limitations of my solution

Due to the nature of this project being a prototype which would be presented to a funding panel for assessment before further funding, there are a number of limitations of the solution. They are outlined below.

- Some aspects of the game might not be as fully fleshed out as some of the examples of similar games listed above. For example, I don't plan on having such an intricate way of meeting partners as some of the games I played in researching; I will instead have a simple list and the player can indicate which one they wish to date;
- The export life function will just be to view the JSON files which the game will save data to during runtime. This will mean that definitely one element of my success criteria won't be completed;
- There will be a lower number of set life events, for example there will be no reference to driving lessons or other events which happen in life in a similar way;
- The education system will end after 6th form / college;
- The `LifeScore` algorithm won't be complete, however the other scores which will be the direct source of information for the `LifeScore` will be complete;
- The marriage section of the game will not be implemented.

1.6 Hardware Requirements

As the application will be developed using Visual Studio 2019, the minimum software requirements will be the same as Visual Studios requirements:

- 1.8 GHz or faster processor. Quad-core or better recommended;
- 2 GB of RAM; 8 GB of RAM recommended (2.5 GB minimum if running on a virtual machine);
- Hard disk space: Minimum of 800MB up to 210 GB of available space, depending on features installed; typical installations require 20-50 GB of free space;
- Hard disk speed: to improve performance, install Windows and Visual Studio on a solid state drive (SSD);
- Video card that supports a minimum display resolution of 720p (1280 by 720); Visual Studio will work best at a resolution of WXGA (1366 by 768) or higher.

Chapter 2

Design

2.1 User Interface Design

In general, the backgrounds will be a darker colour with a lighter shade used to indicate where the buttons are. Most text will be white, with emphasised text in one of the three accent colours. The title bar and menu bar will be Windows' default colour. All elements of my interface will need to be user friendly, meaning the colours will have to contrast against each other.
Note: all colours have been omitted from diagrams as well as the exact controls used on the forms. Panel controls (which will contain the other controls) are represented by the blue boxes. The final layout of forms can be seen in Appendix D

2.1.1 Main Menu

Figure 2.1: Main menu design

This is the menu which will greet the character when they open the game. See the use case diagram for details of where each button leads.

2.1.2 Main Game Screen

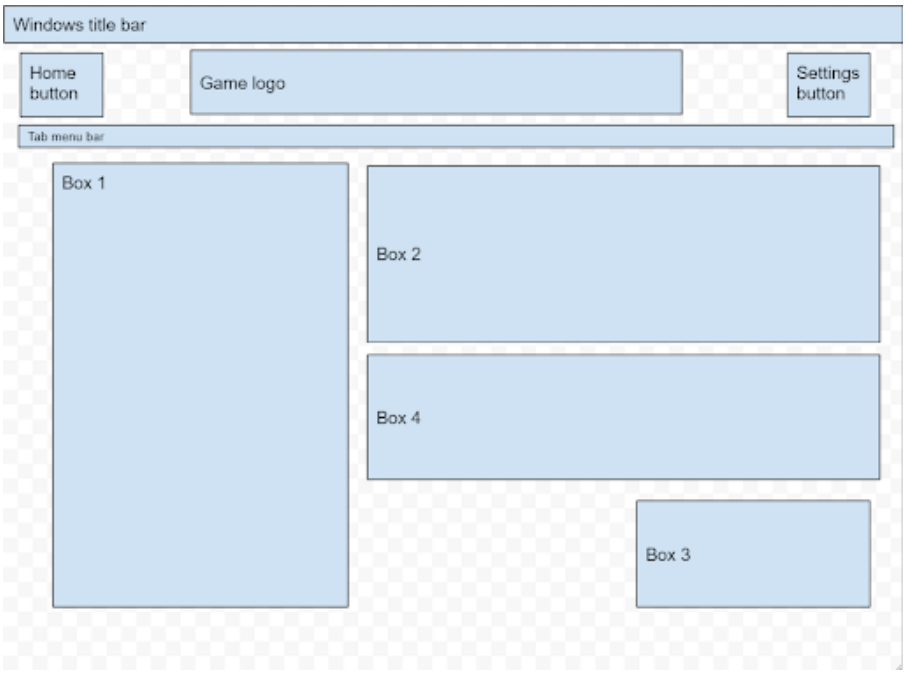


Figure 2.2: Main game screen design

The general layout of all the following will remain the same for ease of use. All in their separate tabs (represented by the "Tab menu bar" underneath the Game logo).
If nothing is going in the box, then it will be left blank.
The home button will also act as a save progress button and prompt the user to say where they want their data saved to.

Home tab

Box 1 - TextBox control containing information about what has just happened in game (generated from the array of event objects).
Box 2 - Information about the character (including their name, age, current education situation).
Box 3 - Age up.
Box 4 -

Education tab

Box 1 - Education events.
Box 2 - Information on current education situation.
Box 3 - Quit school.
Box 4 -

Jobs tab

Box 1 - List of currently available jobs.
Box 2 - Information about the current job situation.
Box 3 - Quit job.
Box 4 - Options regarding new job.

Crime tab

Box 1 - Crime events.
Box 2 -

Box 3 - Commit random crime.
Box 4 -

Customise Character tab

Box 1 - Editable attributes about the character.
Box 2 - End life.
Box 3 - Save button.
Box 4 -

Partners tab

Box 1 - Available partners.
Box 2 - Actions about available partners.
Box 3 -
Box 4 - Information about current partner.

2.1.3 Character Design Form

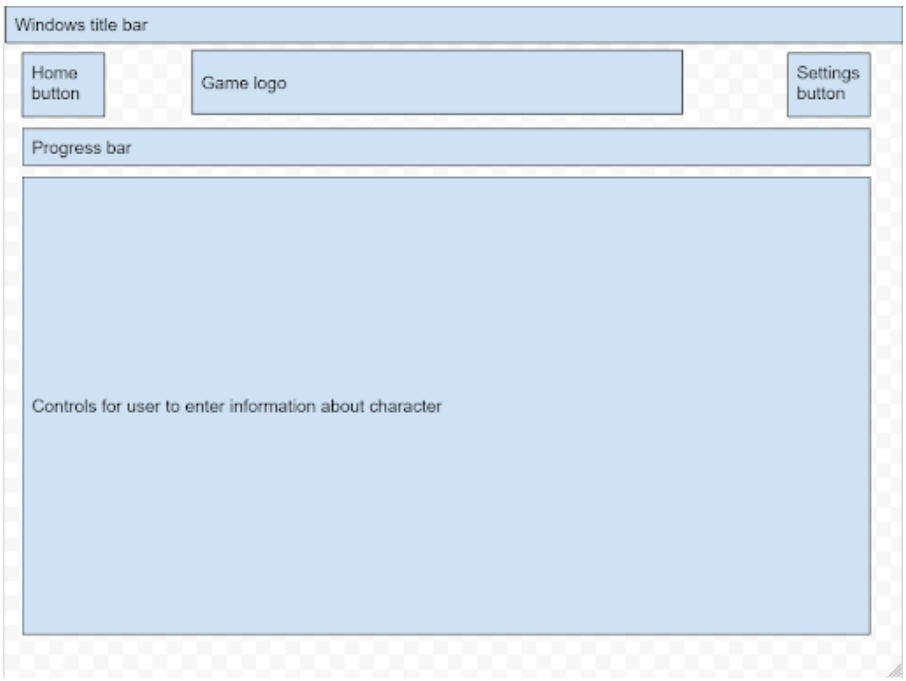


Figure 2.3: Character design form

This will walk you through the steps required to create a new character. The controls on the form will change depending on the current step.

2.1.4 Choice Box

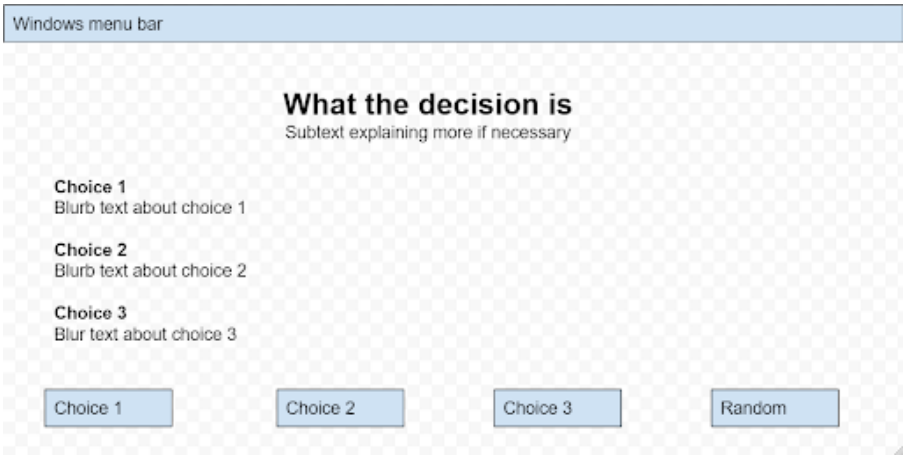


Figure 2.4: Choice box form design

This will be a standard dialogue box (created in a form) which is generated each time there is a 3-way choice for the user to make. The random button will randomly select one of the three choices for the user. It will be the default 'ok' button (it is the button which is pressed when you press the enter key on the keyboard).

2.1.5 Settings

The settings menu will be accessible from within most of the forms of the app. Within it there will be accessibility options.

2.1.6 Icons

Icons will be downloaded from this Flaticon.
Within the website, there is a colour change tool which can be used to change the colour of line icons. I will make use of this feature to make sure all of my icons are visable on the dark background.

2.1.7 Fonts

I will be using Segoe UI in the regular font face, with headings emphasised in bold and a larger size.

2.1.8 Colours

The background of my app will primarily be black, with dark grey buttons.

Where colour will be used	Sample	Hex code	RGB code
App background	Sample text	#1F1F1F	31 31 31
Buttons	Sample text	#666666	102 102 102
Accent 1	Sample text	#64A4D1	100 164 209
Accent 2	Sample text	#A76CB3	167 108 179
Accent 3	Sample text	#F8639A	248 99 154

Table 2.1: Outline of colours used in the app

2.2 In-Game Events

When playing the game, there are a series of events which will happen. The most important one is "ageing" up. This event will progress the game.

2.2.1 Age up event

This event will happen when the player clicks the age up button.
There are lots of random elements to this event which will generate the course of the game for the player.

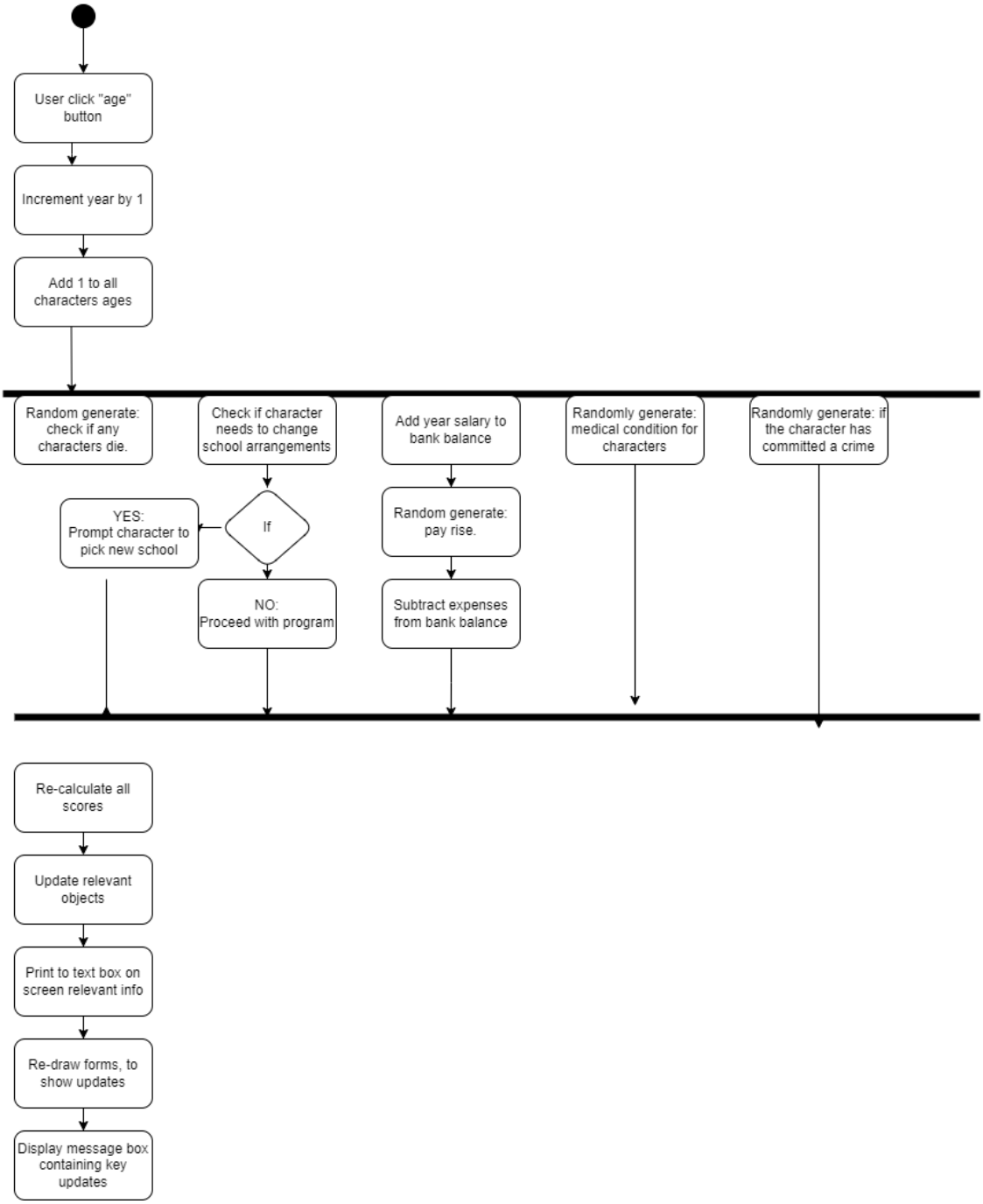


Figure 2.5: Age up event

2.2.2 Pre-Generated Events

Most of the events in the game will be random, meaning each life which is played will have a random and fresh aspect to it. However, there will have to be a set of events which are randomly chosen from, for example - crime (there will be a list of pre-decided crimes that can be randomly picked).

2.3 Usability and Accessibility

The settings menu will contain some accessibility options.
There will be no colour blind friendly options because the main colours of the game will contrast (light text on a darker background). Blues and pinks can sometimes be difficult for red/blue colourblind people to tell apart, as using RGB colours, pinks are achieved by mixing blue and red together. This shouldn't be a problem in my game because even though I am using blues and pinks, they will never be used in critical parts of the game, only to emphasize or accent certain elements.
As this is a windows application, I will make sure to implement the correct tab order to allow people to navigate the entire application using only their keyboard or an on-screen keyboard.

2.4 Class Diagrams

Each attribute will have a get and set method within the class, this is something which C# automatically implements for attributes. All other functions will be elsewhere in the code. Each class will also have a constructor, which will take inputs as parameters and construct the object.

2.4.1 Event

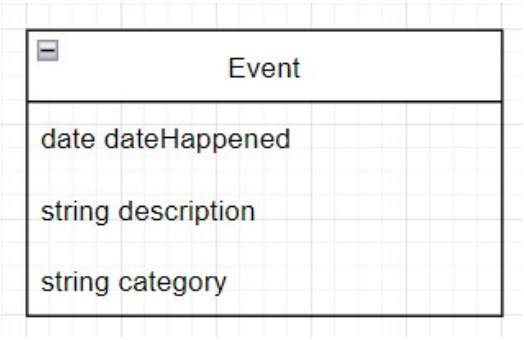


Figure 2.6: Class diagram for Event

The game will be based on events happening, these will be stored in an array of objects in the order in which they occurred. Events will be read-only as there will be no need to modify the event once it has occurred. The category will be used to categorise the events based on what type of events they are. This will allow the user to view only the crime related events for example.

2.4.2 Score

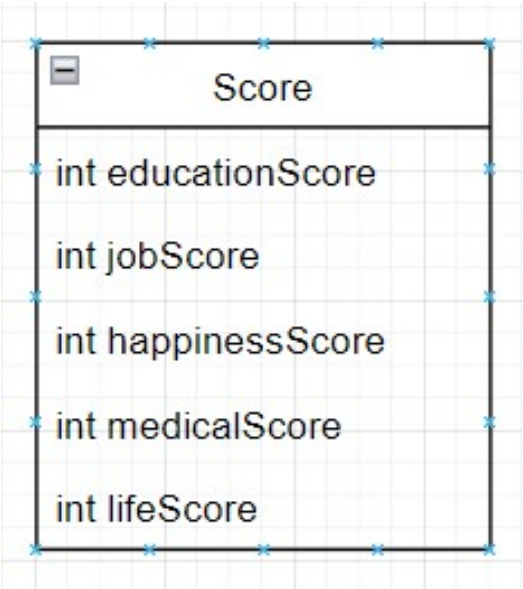


Figure 2.7: Class diagram for Score

This will be used for the main character to keep track of the scores which are key to the game.

2.4.3 Job

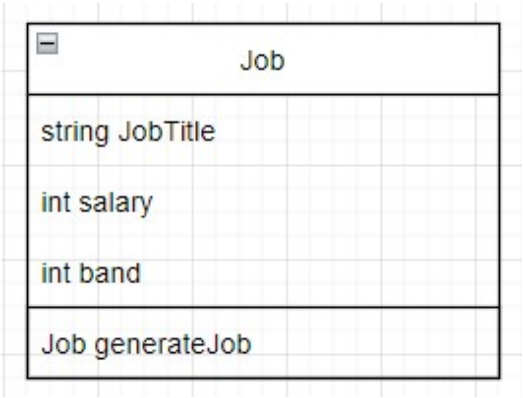
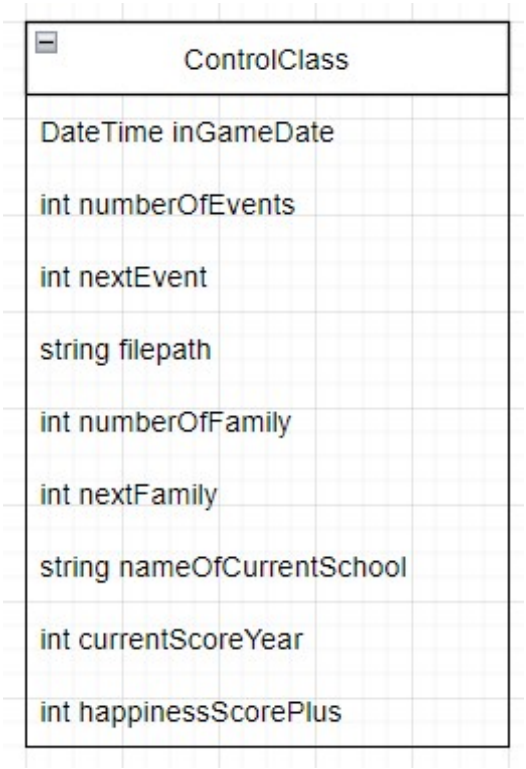


Figure 2.8: Class diagram for Job

There will be multiple uses of this class. One use will be to store the jobs which the game has generated that are available to the main character and the other use will be to store information about the main character's current job.

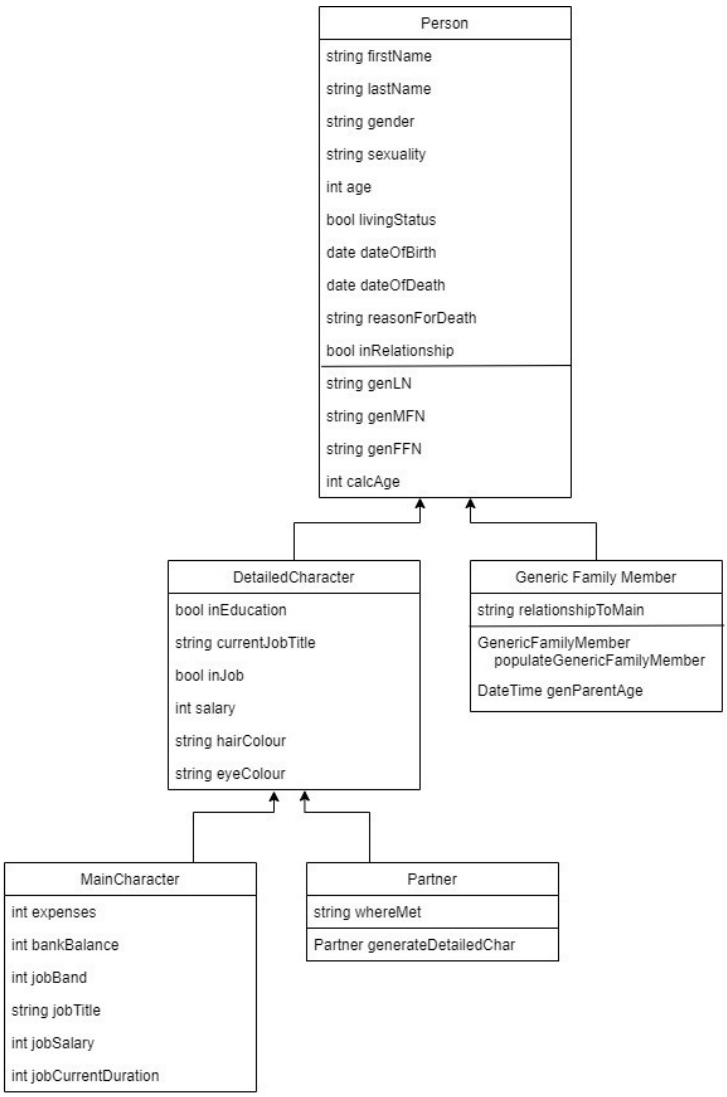
2.4.4 ControlClass



This class will be used to hold information about the current life in the game which means the program can easily manipulate data etc; for example, it stores the supporting information for the main character education element of the game.

Figure 2.9: Class diagram for ControlClass

2.4.5 Person



This is the biggest group of classes, with the superclass person having many subclasses coming off of it. Having so many inherited classes means I can easily control all the aspects of the main character and partners'life.

Figure 2.10: Class diagram for Person

2.5 Use Case Diagrams

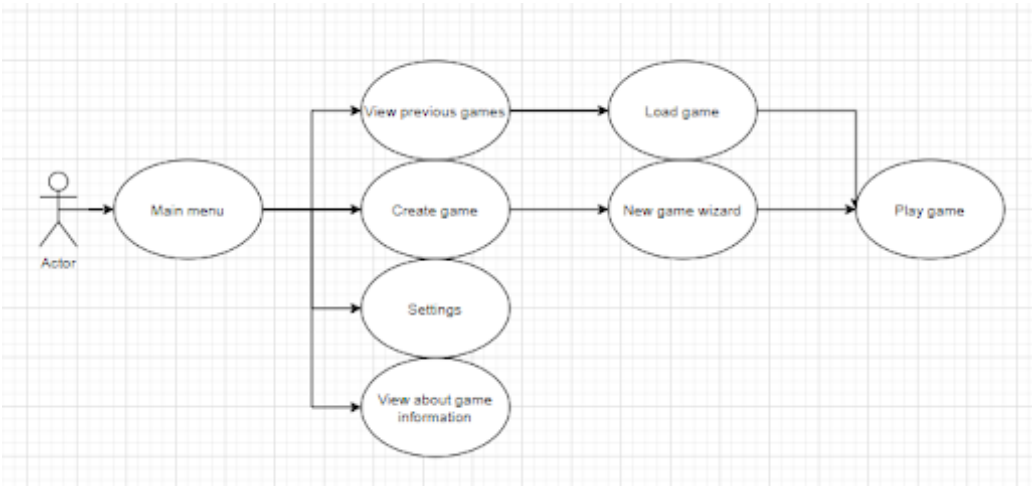


Figure 2.11: Use case diagram for the application

2.6 Algorithms

A lot of my code isn't suitable to be written in pseudocode because it uses lots of Visual Studios tools. For example - the character creations screen will make use of TextBox controls which are accessible by the name assigned; there is no way to write this in pseudocode, and the information from them would be passed into the constructor for the `mainCharacter` class.

2.6.1 Age-Up algorithm

```
on button press age up{
  stopProgress()
  gameDate = gameDate + 1 year
  mainCharacter.setAge(mainCharacter.getAge() + 1)
  ageCharacters()
  random = getRandom(1,10*randomModifier)
  if random == 1 then
    killCharacter(1)
  end if
  returned = checkSchoolChange()
  if returned == true then
    generateAndSelectSchools(mainCharacter.getAge)
  end if
  mainCharacterBank.setBalance(mainCharacterBank.getBalance + mainCharacter.
   ↪ getSalary)
  mainCharacter.setSalary(mainCharacter.getSalary + getRandom(0,1.5)
  mainCharacterBank.setBalance(mainCharacterBank.getBalance - mainCharacter.
   ↪ getExpenses)
  if random == 2 then
    generateMedicalConditionForOtherCharacters()
  end if
  if random == 3 then
    generateMedicalConditionForMainCharacter()
  end if
  if random == 4 then
    generateCrime()
  end if
  if random == 6 then
    generateCrime()
```

```
28      generateMedicalConditionForMainCharacter()
29      killOneCharacter()
30    end if
31    if random == 9 then
32      killMainCharacter()
33    end if
34    updateScores()
35    refillStoryBox()
36    redrawForms()
37    startProgress()
38  }
```

Subroutines called from Age Up algorithm

Age all characters algorithm

`ageCharacters()`

This algorithm will age all the characters apart from the main character by one year. It will also need to make sure that none of the characters are getting too old and if they are change the probability of them dying, to increase the chance of this happening. It will work by using a for loop to loop through all the elements in the characters array, getting their age then setting it to their age plus 1.

Kill one character algorithm

`killCharacter(int numberToKill)`

This will randomly select a character from the array of character objects and kill one of them. If the number passed into it is greater than one then the subroutine will loop around again, killing more than one character. It will generate an event object containing this.

Get random number

`getRandom(int bottom, int top)`

This subroutine will generate a random number between the two numbers specified as parameters.

Check if the main character is due to change schools

`checkSchoolChange()`

This will get the main character's age then check if it is one of the ages where a change of school is needed. If a change of school is needed, then it will return true, if not, it'll return false.

Generate new schools and select which one to go to

`generateAndSelectSchools()`

This will call the `openFile(schools)` function, then loop through a pick 3 of the appropriate aged schools and open the 3-way choice box. It will then take the returned value from the 3-way choice box and update relevant attributes in the main character's object. It will then generate an event object containing information from this.

Generate medical condition

`generateMedicalConditionForOtherCharacters()`

`generateMedicalConditionForMainCharacter()`

These two functions will work in the same way, just for different characters. They both will open the file containing the list of medical conditions, and randomly select one. The main character function will add it to the main character and the other characters one will randomly generate an index in the array of other characters to apply the medical condition to. For both of these, an event object will be generated.

Generate and commit a crime

`generateCrime()`

This will open the file containing a crime and then randomly select one to be committed by the main

character. Crimes will have their name and a severity scale next to them in the file, this means that the program can generate an appropriate punishment for the character. Both of these will be bundled up in an event object.

Killing

`killOneCharacter()`
`killMainCharacter()`
These will work in basically the same way, differing by who they kill. The kill one character will randomly generate a number which is the index in the array in which the person who is going to be killed can be found, then set their status to dead and change the death date to the current in-game date. The kill main character function will change the game state to game over, which will grey out all controls on the form and bring up a game over screen.

Update the scores

`updateScores()`
This will recalculate all the scores in the game, and re-balance the weightings to best reflect what has just happened in the age.

Refill the story box

`refillStoryBox()`
This will loop through the array of event objects and re-fill the TextBox which contains the life events.

Redraw the forms

`redrawForms()`
This will be a standard function used which will re-draw the forms in c# to make sure controls have all the correct data in them.

Stopping Progress

`stopProgress()`
This will stop progress through the game by disabling the buttons which allow progress to happen.

Starting progress

`startProgress()`
This will restart progress through the game by enabling the buttons which allow progress to happen.

2.6.2 Choice Box Algorithm

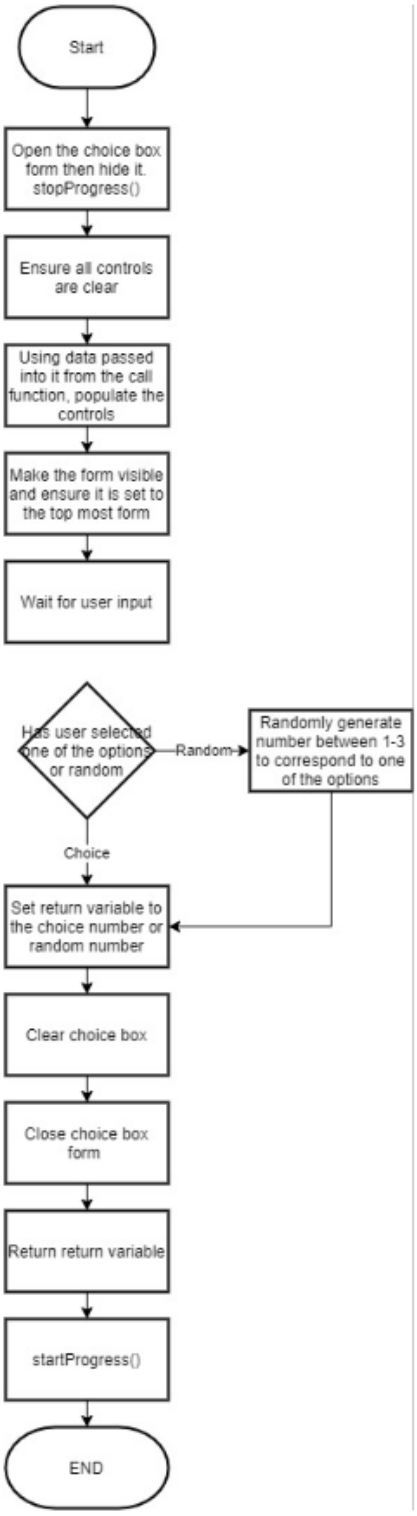


Figure 2.12: Algorithm for the choice box represented as a flowchart

Chapter 3

Testing During Development

During the development of my project, I will need to evaluate each section I program against my success criteria to ensure it works. I will also need to ensure that there are no bugs or errors in the code.

3.1 Types of Testing Used

To test my code as I go along, I will be using the following types of testing;

- White Box testing
- Integration testing

Each type of testing has a different strength. White box testing will allow me to test each programmatic element as I go along as I will know how it should be Integration testing will allow me to test how the different subroutines of the program interface with each other and confirm that data is being passed between them correctly.

3.2 Important elements to test during development

Listed below are the elements of my project which will need to be individually tested before they can be combined into the main game.

- Buttons on the forms
  The buttons on the forms will need to work correctly.
- Generation of classes
  The classes will need to be constructed correctly.
- Files
  The game will need to be able to save data out to files then read it back in.
- Events
  The events in the game will need to be generated and displayed correctly.
- Generation of people
  Both the family members and potential partners need to be generated randomly. These can be tested separately by using message boxes to display information from each person.
- Death
  The death algorithms will need to randomly select a character to die and them kill them.
- Scores
  The scores will need to be calculated and displayed correctly.

- Education
  The education system will need to work as designed my success criteria. This will be programmed in a modular way, so each subroutine can be tested before testing the subsystem as a whole.
- Crimes
  The crimes element will need to work independent of any other system in the game before it is combined.
- Jobs
  This subsystem, can be tested fully on its own before being integrated with the rest of the game.
- Partners
  The partners element to the game can't be tested fully independently because it will make use of some of the same functions which are used in the generate family members systems. This, however, shouldn't be a problem.

3.3 Post Development Testing

After I have completed implementation of the project, I will need to test the complete program to ensure that all the different systems work together correctly. I will be using the test plan detailed in chapter 5.

Chapter 4

Implementation

I will be implementing my project using Microsoft Visual Studio 2019 within a Windows Executable application template using the .NET Core framework version 3.1.
The first form I will design is the main menu as this is what the players will see first when they open the game.

4.1 Designing the Main Menu

I began by setting up the form background colour then added the panel controls then I added the rest of the controls to the screen. I made sure to give all the controls which would have events associated to them in the code a meaningful name, to ensure I could address them without confusion of which control was which. For example, the button which loads a new game is called `btnNewGame` (screenshots of the forms annotated with the control names are available in Appendix D).
I also made sure that the controls on the forms responded dynamically to the size of the window to make sure that as the window was resized, the controls would move to the centre. The final main menu form (design) is shown below:

Figure 4.1: Main menu form design

4.2 Declaring Classes and Passing Objects Between Forms

Now that I have a basic user interface (UI) designed, I am able to begin programming the game which will run behind it. I will start with moving an object between forms. This is fundamental to my

game as I will be constructing an object in one form then moving it to another form where it will be processed. It is important that the object along with all its attributes is moved in its entirety. This means that as the game progresses from the new game screen to the main game play form, the objects already generated are accessible.
To work out how to do this was quite a complicated process as there were lots of different out of date tutorials on the internet, it taught me the importance of testing small elements of the code in a test project before moving them into the main program.

```
frmMainGameScreen fMainGameScreen = new frmMainGameScreen(mainCharacter); //
    ↪ create new frmMainGameScreen with parameter mainCharacter
fMainGameScreen.Show(); //show new main game screen.
```

Listing 4.1: Displaying a new form and passing an object to it

The code above creates the new instance for `frmMainGameScreen` and passes into the constructor the `mainCharacter` object. The `mainCharacter` object is passed from form to form throughout the program.
The code below takes the object (being passed into the new form) as a parameter of the constructor of the form.

```
public frmMainGameScreen(Person mainCharacterTransfer)
{
    InitializeComponent();
    mainCharacter = mainCharacterTransfer; //Set the contents of
    ↪ mainCharacter to mainCharacterTransfer which has come from previous
    ↪ form
}
```

Listing 4.2: New form constructor code containing processing for object being passed in as a parameter

4.3 Creating the Person classes

Using my class diagrams, I then setup the rest of the classes and setup the inheritance for the classes relating to people. For now, I have only inputted the attributes for the classes, methods will be setup later as I need them.

```
 public class GenericFamilyMember : Person
{
    private string relationshipToMain;

    //get and set
    public string RelationshipToMain { get; set; }
}
```

Listing 4.3: GenericFamilyMember class creation

```
public class MainCharacter : DetailedCharacter
{
    //attributes

    private int expenses;
    private int bankBalance;
    private int jobBand;
    private string jobTitle;
    private int jobSalary;
    private int jobCurrentDuration;
    private int salary;

    //get and set
    public int Expenses { get; set; }
    public int BankBalance { get; set; }
```

```
    public int JobBand { get; set; }
    public string JobTitle { get; set; }
    public int JobSalary { get; set; }
    public int JobCurrentDuration { get; set; }
}
```

Listing 4.4: MainCharacter class creation

```
public class Partner : DetailedCharacter
{
    Random rnd = new Random();
    private string whereMet;
    private DateTime whenMet;

    //get and set
    public string WhereMet { get; set; }
    public DateTime WhenMet { get; set; }
}
```

Listing 4.5: Partner class creation

```
public class Person
    {
        //Set all private attributes for the class
        private string firstName;
        private string lastName;
        private string gender;
        private string sexuality;
        private int age;
        private bool livingStatus;
        private DateTime dateOfBirth;
        private DateTime dateOfDeath;
        private string reasonForDeath;
        private bool inRelationship;
        //get and set methods (nb. use uppercase first character for
    ↪ accessable get/set)
        public string FirstName { get; set; }
        public string LastName { get; set; }
        public string Gender { get; set; }
        public string Sexuality { get; set; }
        public int Age { get; set; }
        public bool LivingStatus { get; set; }
        public DateTime DateOfBirth { get; set; }
        public DateTime DateOfDeath { get; set; }
        public string ReasonForDeath { get; set; }
        public bool InRelationship { get; set; }
    }
```

Listing 4.6: Person class creation

4.4 Working More on The Forms

Now I had the basics worked out (how to pass data from one form to another and accessing objects in C#), I designed the new game and show saved games form. Both of these forms are fundamentally the same, with each having text boxes for the attributes of the main character. However, on the show saved games form, the text boxes are read only and they get populated once the user has opened the file.

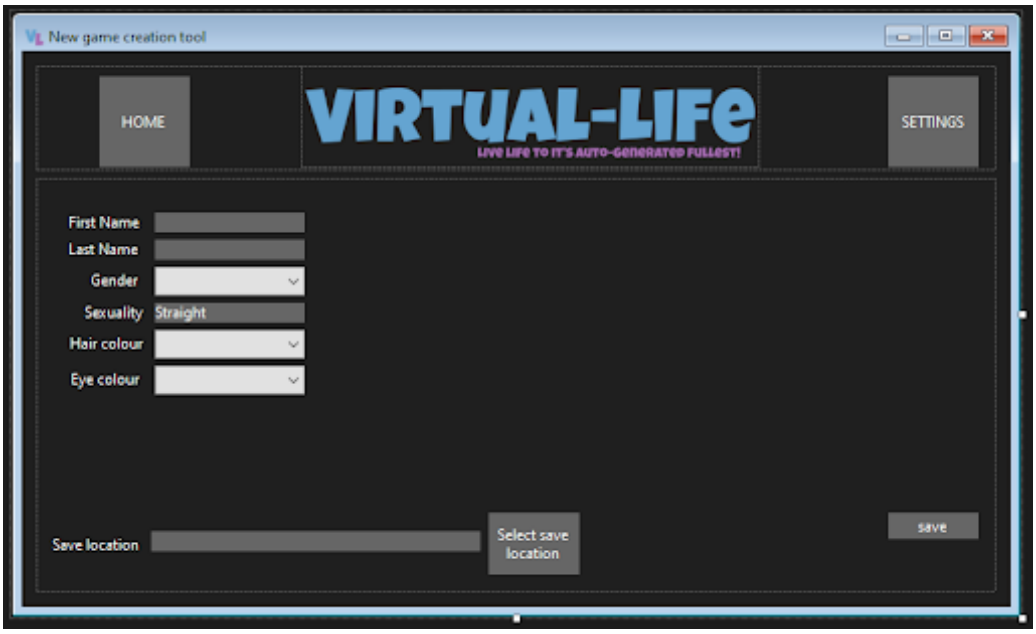


Figure 4.2: Create new game form

The figure below shows the show saved games form. It contains text box controls which will be used to show the main characters information to the player, this will allow them to confirm that they have selected the correct save before the computer loads the data then then the game progresses to the next form.

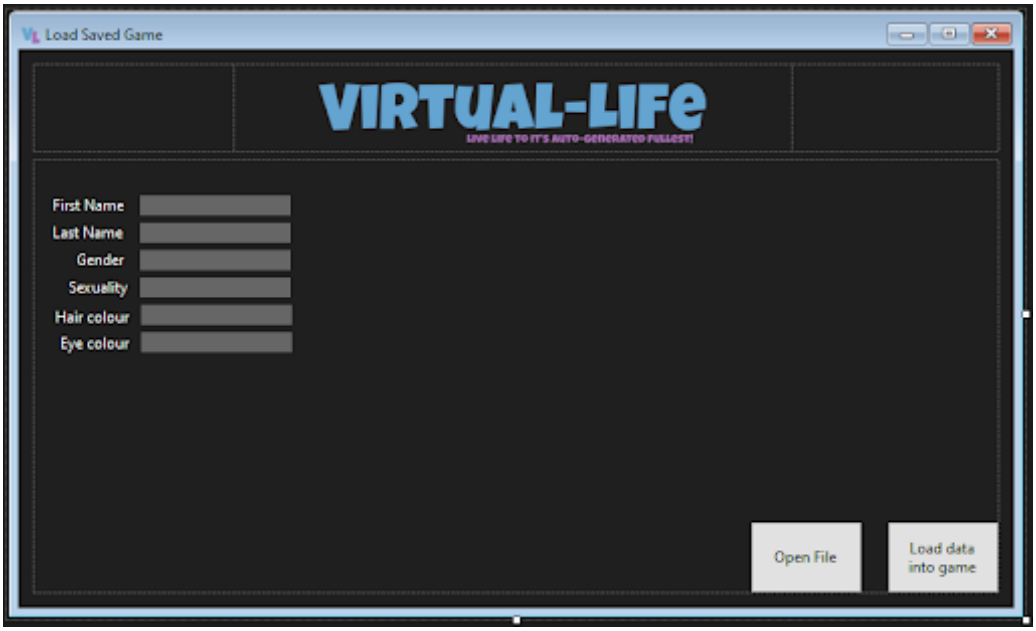


Figure 4.3: Show saved games form

4.5 File Handling

One of the key elements of my success criteria was the ability to save a game to a file and read it back in when the app was opened again.
As I had worked with JSON files in previous projects before, I decided to use them for this project as I understood how to use them effectively. I did some research into saving to a JSON file and found a package called Newtonsoft. This package can serialize and deserialize the JSON file. The documentation for the package is bad at explaining why things work, so I spent quite a lot of time

working out how to manually write the contents of each object out to an individual JSON file before I realised that part of the package might be able to do that for me.
This is on my to-do list as a lower priority task as I feel like I'm focusing on the small details too much and not working on the actual game. I have also got a single object read back in by the program, however this is the only object in the file so I'm not sure if the same code will work when I introduce different objects to the file. At this moment in time, this isn't meeting my success criteria.

4.6 Loading Bar

This wasn't in my design, however I believe it adds some value to the game, especially when handling files as wait times could be increased on slower systems.
It is a small form which appears with a marquee scrolling progress bar which says "Loading" on it, to tell the user that the game is doing something when the program is doing something; for example, when it is saving to a file.
The progress bar begins to move automatically, this is setup in the control properties within Visual Studio.

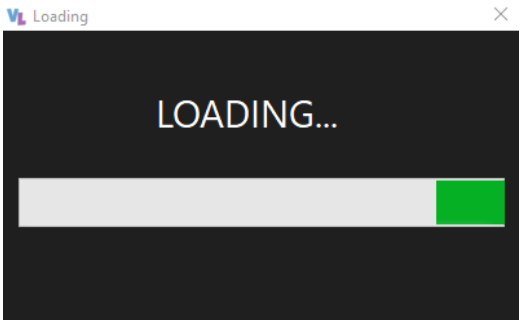


Figure 4.4: Loading bar form

```
public partial class frmLoadingBar : Form
{
    public frmLoadingBar()
    {
        InitializeComponent();
    }
}
```

Listing 4.7: Code for the loadingbBar form

4.7 Building Main Game Screen form

I began to build the form for the main game screen, by adding the TabControl which will allow me to have different tabs of information on one form, reducing the number of different forms which have to be loaded into memory at the same time; this also produces all-on-one-screen design which many of the games I played in research had.
After the TabControl, I added more panel controls to separate the different boxes of the information, to make resizing the form dynamically easier.

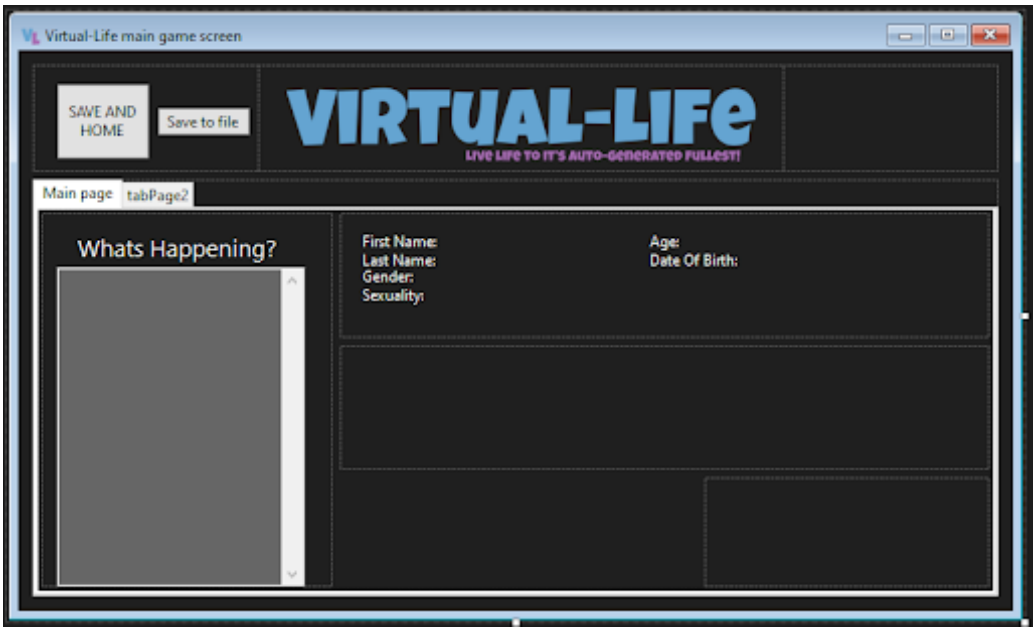


Figure 4.5: Work in progress of main game screen form

4.8 Array of Event Objects

To store the various events I will have throughout the game, I will be creating an object for each one and to make programming and future maintenance easier, I will be storing them in an array.
I first wrote the code which made it work on one form. This worked the first time as I was following a tutorial and looking at errors which Visual Studio was giving me; correcting them as I programmed more. Then I setup the constructor for the next form in the game (main game screen) to receive the array of events object.

```
1 Event[] eventArray = InitializeArray<Event>(numberOfEvents);
```

Listing 4.8: First attempt at declaring an array of objects

This resulted in the following error:
`A field initializer cannot reference the non-static field, method, or property 'frmNewGame.numberOfEvents'`
To fix this, I tried removing the variable numberOfEvents and changing it to a static integer like so:

```
1 Event[] eventArray = InitializeArray<Event>(10);
```

Listing 4.9: Second attempt at declaring an array of objects

This still gave me the following error:
`A field initializer cannot reference the non-static field, method, or property 'frmNewGame.InitializeArray<Event>(int)'`
To try to fix this, I ignored the InitializeArray function and just declared a new array of objects:

```
1 public Event[] eventArray = new Event[10];
```

Listing 4.10: Third attempt at declaring an array of objects

However, this gave me a null error saying the objects aren't defined.
I reverted back to the previous way and did some research - this pointed me at a Microsoft Docs page detailing Compiler Error CS0236.
From previous knowledge, I realised that compiler errors aren't something which are simple to understand, so I tried another option.
This gave me the following error: `System.NullReferenceException:' Object reference not set to an instance of an object.'`

To resolve this error, I did more research and settled on the following solution.
This first block of code comes from when the new game screen form is loaded.

```
//Create array of event object called eventArray
public int numberOfEvents = 1000;
public Event[] eventArray = new Event[1000];
public int nextEvent = 0;
```

Listing 4.11: Final solution for declaring an array of objects part 1

This next block of code is run when the save button is pressed

```
//create event objects needed for game
for (int i = 0; i < controlClass.NumberOfEvents; i++)
{
    eventArray[i] = new Event();
}
```

Listing 4.12: Final solution for declaring an array of objects part 2

This works by setting up the integer variable `numberOfEvents` then the array of type `Event` containing 1000 `Event` objects and then the integer variable `nextEvent` which would be used to find the next available slot in the array meaning I could very quickly input the next item in the array without having to locate it every time I needed to enter something into it. The `numberOfEvents` variable could be set dynamically using an inbuilt C# method for array handling - my intention is to change this over. This change would mean that I would only need to change the length of the array once in the program and the change would be reflected all throughout the code. After I had added the code which is needed to populate the array:

```
eventArray[controlClass.NextEvent].Category = "Life Event";
eventArray[controlClass.NextEvent].DateHappened = DateTime.Today;
eventArray[controlClass.NextEvent].Description = "You were born to parents
    ↪ xxx;
controlClass.NextEvent++;
```

Listing 4.13: Code needed to create the first event in the array

I tested it and I got the outcome which I wanted to see - the text box on the main game screen showing the event.

Figure 4.6: Evidence that the array of Event objects are being constructed correctly

4.9 File Handling - Attempt 2

When I worked on loading the saved files into the game, I realised that the JSON package handler I was using had a serialisation and deserialization function built in which is much more elegant and resource efficient than the method which I was using. I looked into the different methods which came with `Newtonsoft.Json` and found the `JsonConvert.SerializeObject(objectName)` and tested writing out to a file using that. This worked first time with the following code.

```
File.WriteAllText(filepath, JsonConvert.SerializeObject(mainCharacter));
```

Listing 4.14: Code used to write out to JSON files

However, I also had to include `Newtonsoft.Json` separately to `Newtonsoft.Json.Linq` at the top of the project. I then began to look at how to write out the other objects to the file. It was extremely difficult to get the writing out and reading in working. I spent a lot of time following tutorials which ended in an error or not the result I was expecting. I decided that to make this work, I would need to

go back to basics and work out exactly what I needed, not what I wanted. This re-designing allowed me to realise that there was no need for all the objects to be written out to the same file, they could be in fact written out to separate files within one folder. This epiphany resulted in me solving how to get the objects written out to separate files, and read them in again. To write them out, I did it in much the same way as before however, in my `File.WriteAllText` statement, I also had to include the file name. The code below shows how a single object is written out to a JSON file. This is repeated multiple times in the code with the filepath and object name substituted.

```
File.WriteAllText(controlClass.Filepath + "\\mainCharacter.json",
    ↪ JsonConvert.SerializeObject(mainCharacter));
```

Listing 4.15: Code used to write out to JSON files

This was much the same for when a file is being opened.

```
MainCharacter mainCharacter = JsonConvert.DeserializeObject<MainCharacter>(
    ↪ File.ReadAllText(loadFilepath + "\\mainCharacter.json"));
```

Listing 4.16: Code used to open a JSON file and read the contents of it into an object

To select the folder which I was going to be using, I used a FolderBrowserDialog control which comes as standard with the Windows Form Library.

4.10 Control Class

Whilst designing the file management system, I realised that there were lots of data items which would need to be stored alongside the game for the game to work. Up until now, I had been storing them as variables but hadn't thought about how I would need to save them. I decided to save them as an object, this way I can easily save to a file in the same way I am doing for the other objects. This control class also means I am able to pass data between forms much simpler, as I can simply pass the entirety of the `controlClass` to the new form rather than the individual attributes separately.

```
public class ControlClass
{
    //private attributes
    private DateTime inGameDate;
    private int numberOfEvents;
    private int nextEvent;
    private string filepath;
    private int numberOfFamily;
    private int nextFamily;
    private string nameOfCurrentSchool;
    private int currentSchoolYear;
    private int eduScorePlus;
    private int happinessScorePlus;


    //get and sets
    //GENERIC GAME STUFF----
    public DateTime InGameDate { get; set; }
    public string Filepath { get; set; }
    //EVENTS-------
    public int NumberOfEvents { get; set; }
    public int NextEvent { get; set; }
    //FAMILY--------
    public int NumberOfFamily { get; set; }
    public int NextFamily { get; set; }
    //EDUCATION------
    public string NameOfCurrentSchool { get; set; }
    public int CurrentSchoolYear { get; set; }
    public int EduScorePlus { get; set; }
```

```
30
31      //HAPPINESS SCORE-----
32      public int HappinessScorePlus { get; set; }
33
34  }
```

Listing 4.17: Declaration of Control Class

4.11 Random Generation of Family Members

I decided that I would be using the constructor of the `GenericFamilyMember` class to populate the attributes of each object. To differentiate between each family member, I will be passing into the constructor the number in the `familyArray`, allowing me to use a switch statement to differentiate between family members.
To start with, I am using 2 attributes of index 0, which is the father of the main character. This will allow me to test the random generation aspect (in the `FirstName` attribute) and the fixed value aspect (in the `RelationshipToMain` attribute).
I wrote the following code.

```
1   public GenericFamilyMember(int charNumber)
2   {
3       //construct here
4       //go through a switch statement to set the character based on what their
        ↪  relationship to the main is.
5       switch(charNumber)
6       {
7           case 0:
8               //Dad to main character
9               FirstName = genMFN();
10              RelationshipToMain = "Father";
11              break;
12          default:
13              FirstName = null;
14              break;
15      }
16  }
17
18  //Generate a Male first name
19  public static string genMFN()
20  {
21      //generate male first name
22      string[] name = { "Liam", "Noah", "Oliver", "Elijah", "William", "James"
        ↪ , "Benjamin", "Ben", "Lucas", "Henry", "Alex", "Ethan", "Daniel", "
        ↪ Sebstian", "Jack", "Matt", "John", "Joe", "David", "Josh", "Julien", "
        ↪ Leo", "Isaac", "Thomas", "Max", "Andy", "Phill", "Harvey", "Ryan" };
23      string returnName = name[rnd.Next(1, 29)];
24      return returnName;
25  }
```

Listing 4.18: Initial test of generating a Random Character

This produced the following error: `CS0120 An object reference is required for the non-static field, method or property 'GenericFamilymember.rnd'`
To solve this, I moved the constructor for the `Random` class to the function `genMFN`. This worked, and allowed me to generate a random number which I then used to select one of the names in the array name.

```
1   Random rnd = new Random();
```

Listing 4.19: Creating a Random object of class Random for later use

The data now appeared as I was expecting it to do in the `familyArray` file.

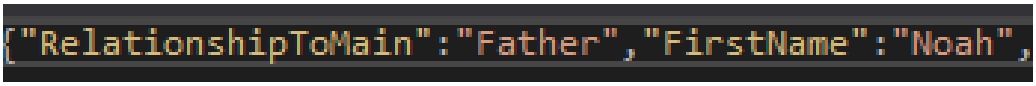


Figure 4.7: Screenshot of familyArray file containing correct data

As I was creating more of the constructor methods, I realised that it would make more sense to move the random generation functions into the `Person` class so that all of the subclasses can use them for their own character generation. The code now looks like this

```
 //functions used in constructors
public static string genLN()
{
    //Generate female first name
    Random rnd = new Random();
    string[] name = { "Smith", "Brown", "Wilson", "Thomson", "Robertson", "
    ↪ Campbell", "Stewart", "Macdonald", "Murray", "Reid", "Taylor", "Clark"
    ↪ , "Mitchell", "Ross", "Watson", "Miller", "Gray", "Simpson", "Duncan",
    ↪  "Bell", "Grant", "Mackenzie", "Allan", "Wood", "Muir", "Watt", "King"
    ↪ , "Bruce", "Boyle", "Douglas" };
    string returnName = name[rnd.Next(1, 29)];
    return returnName;
}
public static string genMFN()
{
    //generate male first name
    Random rnd = new Random();
    string[] name = { "Liam", "Noah", "Oliver", "Elijah", "William", "James"
    ↪ , "Benjamin", "Ben", "Lucas", "Henry", "Alex", "Ethan", "Daniel", "
    ↪ Sebstian", "Jack", "Matt", "John", "Joe", "David", "Josh", "Julien", "
    ↪ Leo", "Isaac", "Thomas", "Max", "Andy", "Phill", "Harvey", "Ryan" };
    string returnName = name[rnd.Next(1, 29)];
    return returnName;
}

public static string genFFN()
{
    //Generate female first name
    Random rnd = new Random();
    string[] name = { "Olivia", "Sophia", "Maria", "Mia", "Evelyn", "Jess",
    ↪ "Ella", "Zoe", "Jemma", "Gemma", "Emily", "Nuala", "Maggie", "Ciara",
    ↪ "Scarlett", "Layla", "Chloe", "Ellie", "Hazel", "Lucy", "Niamh", "Kat"
    ↪ , "Victoria", "Lily", "Hannah", "Chloe", "Lara", "Bella", "Ruby" };
    string returnName = name[rnd.Next(1, 29)];
    return returnName;
}
```

Listing 4.20: Random name generation functions

I then tested opening a saved game and got an error as the JSON deserialize function creates new instances of the `GenericFamilyMember` to populate as it reads in the file. To fix this, I moved the constructor into a separate method within the `GenericFamilyMember` class and called it like this (shown from the new game form)

```
for (int i = 0; i < controlClass.NumberOfFamily; i++)
{
    GenericFamilyMember tempFamily = new GenericFamilyMember(); //make new
    ↪ instance of the object
    tempFamily.populateGenericFamilyMember(i, mainCharacter); //populate
    ↪ that object using the 'constructor' in the GenericFamilyMemberclass
```

```
    familyArray[i] = tempFamily; //transfer the temp object into the array.
}
```

Listing 4.21: Code used to generate generic family members

I then moved onto the generation of the date of birth. To do this, I wrote the following function, which produces a date between 16 and 30 years ago.

```
static DateTime genParentAge(DateTime mcDOB)
{
    //need to generate a date between 16 and 30 years ago from main
    ↪ character birthday.
    DateTime dateOfBirth;
    Random rand = new Random();
    int randomNumber = rand.Next(5845, 10957);
    dateOfBirth = mcDOB.AddDays(-randomNumber);


    return dateOfBirth;
}
```

Listing 4.22: Function which generates DoB of parent

When I tested it, I got the following result; proving it works.

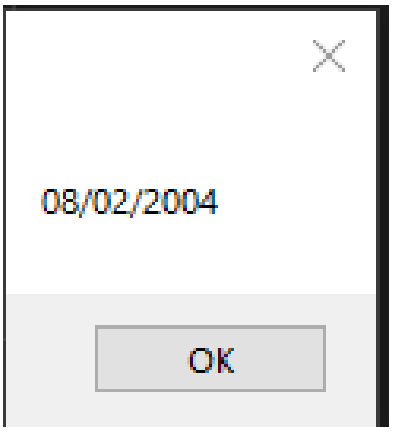


Figure 4.8: Proof that the parent DoB generation algorithm works

As this is working for the father, I implemented it for the mother as well.
I then implemented the age calculation algorithm, I thought about a number of different ways I could do this and settled on passing in the character's birthday and returning an integer containing their age.

```
public int calcAge(DateTime input)
{
    int difference = (DateTime.Today.Date - input.Date).Days;
    float toround = (difference / 365);
    int age = ((int)toround);

    return age;
}
```

Listing 4.23: Age generation algorithm

This worked first time so I implemented it for the mother as well.
The age was the last element I had to programmatically generate for the main character's parents. The finished constructor for the mum and dad can be seen below:

```
public GenericFamilyMember populateGenericFamilyMember(int charNumber,
    ↪ MainCharacter mainCharacter)
```

```
{
    //construct here
    //go through a swtich statement to set the character based on what their relationship to the main is.
    //setup random number generation
    Random rnd = new Random();
    GenericFamilyMember familyMember = new GenericFamilyMember();
    switch (charNumber)
    {
        case 0:
            //Dad to main character
            RelationshipToMain = "Father";
            FirstName = genMFN();
            LastName = mainCharacter.LastName;
            Gender = "Male";
            Sexuality = "Straight";
            DateOfBirth = genParentAge(mainCharacter.DateOfBirth);
            LivingStatus = true;
            InRelationship = true;
            Age = calcAge(DateOfBirth);
            break;
        case 1:
            //mum to main character
            RelationshipToMain = "Mother";
            FirstName = genFFN();
            switch (rnd.Next(1, 3))
            {
                case 1:
                    //same last name as father and main character
                    LastName = mainCharacter.LastName;
                    break;
                case 2:
                default:
                    //different last name to father and main character - need to randomGen this
                    LastName = genLN();
                    break;
            }
            Gender = "Female";
            Sexuality = "Straight";
            DateOfBirth = genParentAge(mainCharacter.DateOfBirth);
            LivingStatus = true;
            InRelationship = true;
            Age = calcAge(DateOfBirth);
            break;
        default:
            FirstName = null;
            break;
    }
    return familyMember;
}
```

Listing 4.24: Final code for generation of mother and father to main character

While generating a new family member, I received an index out of bounds error for the random name generation. To fix this, I change the top random number to 30 as I had forgotten that arrays are zero-based therefore they don't actually have an item 30 therefore the random generation top bound would need to be reduced by 1. I also moved the random generation lower bound down to 0, to accommodate for the zero-based nature of the array.

4.12 Family Tab

Now I have developed the generation of the various characters, I need a way to display them. I am using the `DataGridView` control which presents the data in table. I did some research and found the following line of code which works and displays the following.

```
dgvFamily.DataSource = familyArray;
```

Listing 4.25: First attempt at using a DataGridView control to show the family tab.

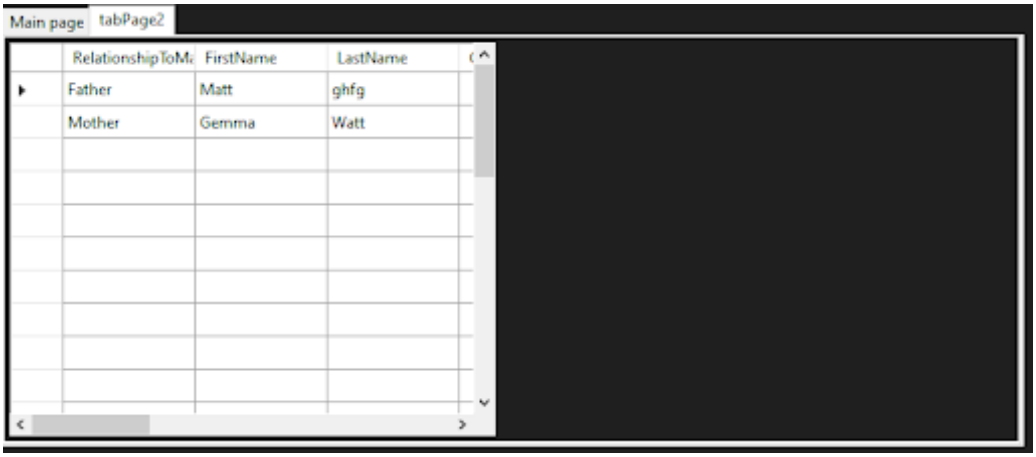


Figure 4.9: Result of initial test of using the DataGridView control for the family tab.

This displays the whole object at once - meaning the user has to scroll horizontally to see all the data. This is not what I wanted so I went back through my old projects and found the code that did what I wanted.

```
 //first, set var for which column contains which data
int relationshipCol = 0;
int firstNameCol = 1;
int lastNameCol = 2;

//now loop through familyArray and populate dgvFamily
// can use a for loop as number of family members are in the controlClass

for(int x=0; x<controlClass.NumberOfFamily; x++)
{
    int i = dgvFamily.Rows.Add();
    dgvFamily.Rows[i].Cells[relationshipCol].Value = familyArray[x].
    ↪ RelationshipToMain;
    dgvFamily.Rows[i].Cells[firstNameCol].Value = familyArray[x].FirstName;
    dgvFamily.Rows[i].Cells[lastNameCol].Value = familyArray[x].LastName;
}
```

Listing 4.26: Improved method of populating dgvFamily

This gave me an error: `System.InvalidOperationException: 'Rows cannot be programmatically added to the DataGridView's rows collection when the control is data-bound.'`
Which I rectified by removing the previous line of code used by commenting it out. This worked and gave me the result I expected. (At this point, I had also added some colors to the control using Visual Studio's property panel).

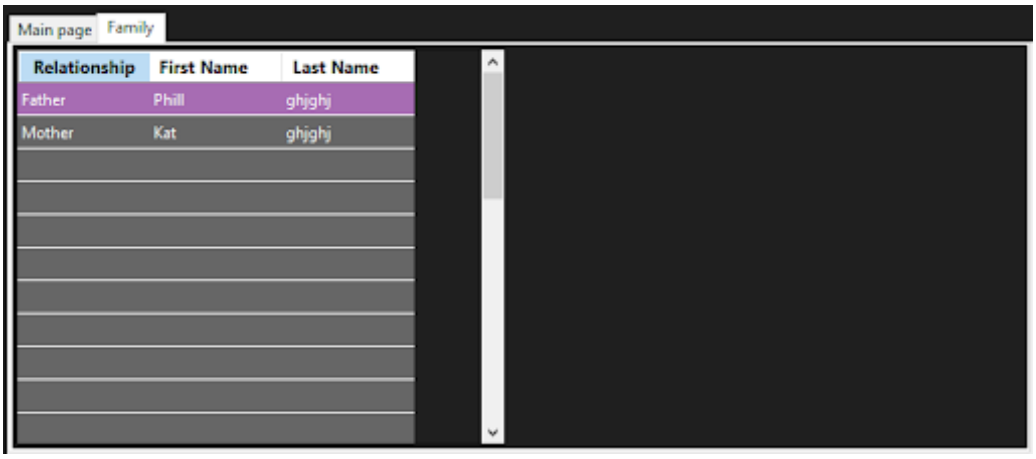


Figure 4.10: Second test of displaying information on the DataGridView control

Now I have the basic code working, I want to change it so it only shows the entries in the array which have data in them. I will do this by including an IF statement inside the FOR loop, like so:

```
for(int x=0; x<controlClass.NumberOfFamily; x++)
{
    if (familyArray[x].FirstName != null)
    {
        int i = dgvFamily.Rows.Add();
        dgvFamily.Rows[i].Cells[relationshipCol].Value = familyArray[x].
    ↪ RelationshipToMain;
        dgvFamily.Rows[i].Cells[firstNameCol].Value = familyArray[x].
    ↪ FirstName;
        dgvFamily.Rows[i].Cells[lastNameCol].Value = familyArray[x].LastName
    ↪ ;
    }
}
```

Listing 4.27: Improved code for displaying data in the DataGridView

This works and the DataGridView now looks like this:

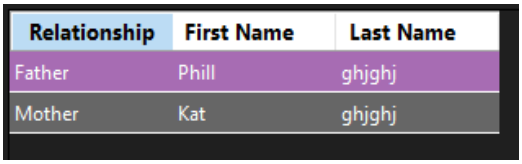


Figure 4.11: The DataGridView showing populated rows only

I then added the code which populates the label controls on the form - this allows the person's information to be displayed to the player.

```
private void dgvFamily_CellContentClick(object sender,
    ↪ DataGridViewCellEventArgs e)
{
   // MessageBox.Show(dgvFamily.CurrentCell.RowIndex.ToString());
    int currentIndex = dgvFamily.CurrentCell.RowIndex;
    //MessageBox.Show(familyArray[currentIndex].Age.ToString());

    //populate info on form
    lblFamRealtionshipToMain.Text = mainCharacter.FirstName + "'s " +
    ↪ familyArray[currentIndex].RelationshipToMain;
    lblFamFirstName.Text = "First Name: " + familyArray[currentIndex].
    ↪ FirstName;
```

```
10      lblFamLastName.Text = "Last Name: " + familyArray[currentIndex].LastName
    ↪ ;
11      lblFamGender.Text = "Gender: " + familyArray[currentIndex].Gender;
12      lblFamSexuality.Text = "Sexuality: " + familyArray[currentIndex].
    ↪ Sexuality;
13      lblFamAge.Text = "Age: " + familyArray[currentIndex].Age.ToString();
14      lblFamLivingStatus.Text = "Living Status: " + familyArray[currentIndex].
    ↪ LivingStatus.ToString();
15      lblFamDateOfBirth.Text = "Date Of Birth: " + familyArray[currentIndex].
    ↪ DateOfBirth.ToShortDateString();
16      lblFamInRelationship.Text = "In Relationship: " + familyArray[
    ↪ currentIndex].InRelationship.ToString();
17      if(familyArray[currentIndex].LivingStatus == false)
18      {
19          //person is dead so can show death date and reason for death
20          lblFamDateOfDeath.Text = "Date Of Death: " + familyArray[
    ↪ currentIndex].DateOfDeath.ToShortDateString();
21          lblFamReasonForDeath.Text = "Reason For Death: " + familyArray[
    ↪ currentIndex].ReasonForDeath;
22      }
23      else
24      {
25          //person is not dead - don't show death date and reason for death
26          lblFamDateOfDeath.Text = "Date Of Death: N/A (haven't died yet!)";
27          lblFamReasonForDeath.Text = "Reason For Death: N/A (haven't died yet
    ↪ !)";
28      }
29  }
```

Listing 4.28: Code to populate labels on Family Tab containing information about the selected person

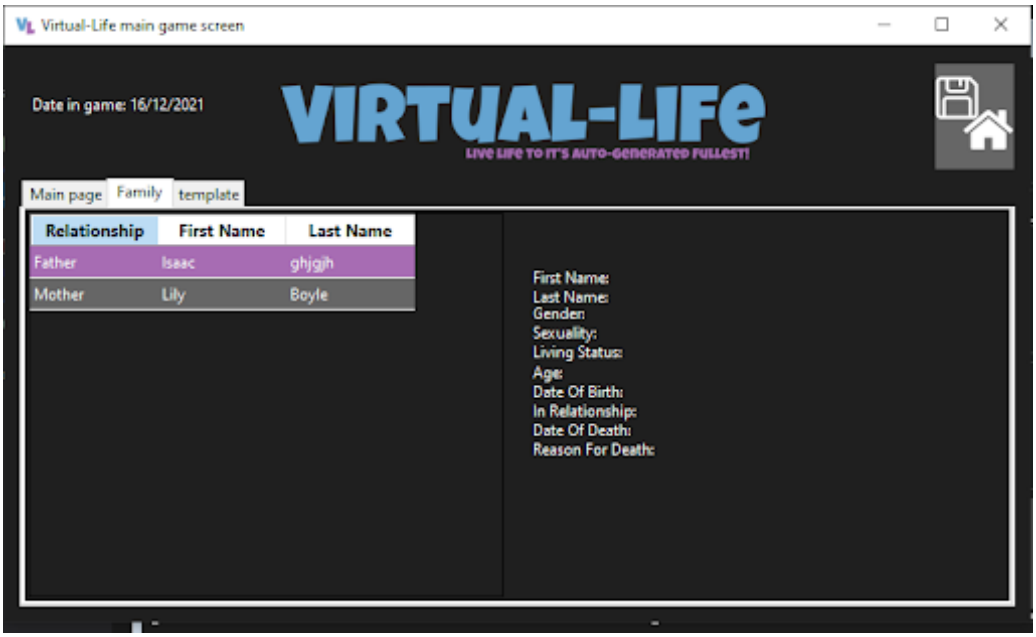


Figure 4.12: Empty labels, before any information is put into them

Figure 4.13: Filled out labels, after the father has been selected

In the figures above, it looks like the father has been selected before it actually has. This is a feature of the `DataGridView`, in that there has to be a row 'selected'at all times.

4.13 Loading the Files in

During developmental testing, I realised that I had quite a serious bug. From moving from system to system, the filepath of the game wasn't dynamically adapting to the system, meaning I couldn't use a save I made on the college computer at home as the file path in the control class was wrong. To resolve this issue, I added this line of code to the opening files form:

```
1 controlClass.Filepath = filepath;
```

Listing 4.29: Line of code used to fix the filepath issues

This overrides whatever had been read in from the control class file with the location specified by the user to find the saved data from the game.

4.14 Death

Now, I need to start processing the events of the age up algorithm, the first thing I am going to program is the death checker. This algorithm will check based on the character's age the probability of them dying and return a boolean for if they shall live or not, if they don't live, then their data will be sent to another subroutine which will kill' them and update the relevant variables to reflect this. I started by testing the ageing up of family members and realised that I hadn't yet programmed this. This means I have to do this first.
I added a procedure in which loops through all the elements of the array `familyArray` and if the living status is true, then it adds 1 to the age of the character. This will allow me to not age up characters once they have died.

```
1 private void ageUpFamilyArray()
2 {
3     //loop through all the members of the family array and add 1 to their
    ↪ age if livingstatus is true
4     for(int x=0; x<controlClass.NumberOfFamily; x++)
5     {
6         if(familyArray[x].LivingStatus == true)
```

```
        {
            //can age them up
            familyArray[x].Age = familyArray[x].Age + 1;
        }
    }
}
```

Listing 4.30: Method which increases the age of the family array

To confirm that this was working as intended, I ran the code and confirmed the father of the main characters age before and after I pressed the age up button. It behaved as expected; therefore I was satisfied to move onto the next aspect of the death checker.
I added the following code which changes the chance of the main character dying based on their age.

```
public bool deathCheckerMainCharacter(int age)
{
    bool deathBool = false;

    int age = mainCharacter.Age;
    int deathChance = 1;
    //for as main char gets older, their death chance increases
    /* 00-10 1/30
     * 11-16 1/25
     * 17-30 1/15
     * 31-50 1/25
     * 51-67 1/20
     * 68-80 1/15
     * 81-100 1/8
     * >100 1/3
     */
    if (age <= 10)
    {
        deathChance = 30;
    }
    else if (age >= 11 && age <= 16)
    {
        deathChance = 25;
    }
    else if (age >= 17 && age <= 30)
    {
        deathChance = 15;
    }
    else if (age >= 31 && age <= 50)
    {
        deathChance = 25;
    }
    else if (age >= 51 && age <= 67)
    {
        deathChance = 20;
    }
    else if (age >= 68 && age <= 80)
    {
        deathChance = 15;
    }
    else if (age >= 81 && age <= 99)
    {
        deathChance = 8;
    }
    else if (age >= 100)
    {
        deathChance = 3;
    }
```

```
49      //now generate random number based off of 1 and deathChance
50      Random rnd = new Random();
51      int randomReturn = rnd.Next(1, deathChance);
52      if (randomReturn == 2)
53      {
54          //DEATH
55          deathBool = true;
56      }
57      else
58      {
59          deathBool = false;
60      }
61
62      //return deathBool to main program where it will be processed.
63      return deathBool;
64  }
```

Listing 4.31: Main character death checker algorithm

I then added the code which will programmatically kill the main character:

```
1   public void killMainCharacter()
2   {
3       //very rough for now, will need to polish at some point
4
5       btnAgeUp.Enabled = false;
6       mainCharacter.DateOfDeath = controlClass.InGameDate;
7       mainCharacter.LivingStatus = false;
8       mainCharacter.ReasonForDeath = generateReasonForDeath();
9       //generate event showing this
10      eventArray[controlClass.NextEvent].Category = "Death";
11      eventArray[controlClass.NextEvent].DateHappened = controlClass.
    ↪ InGameDate;
12      eventArray[controlClass.NextEvent].Description = "You died due to " +
    ↪ mainCharacter.ReasonForDeath;
13      controlClass.NextEvent++;
14      MessageBox.Show(mainCharacter.FirstName + " has died a tragic death due
    ↪ to " + mainCharacter.ReasonForDeath, mainCharacter.FirstName + " has
    ↪ died", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
15      saveToFile();
16  }
```

Listing 4.32: Kill main character method

This produces the outcome of:

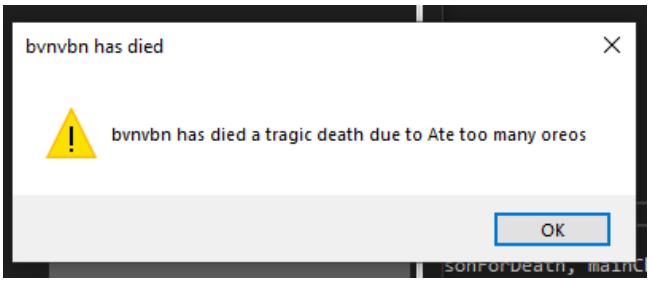


Figure 4.14: Alerting message box displayed when Main Character Dies

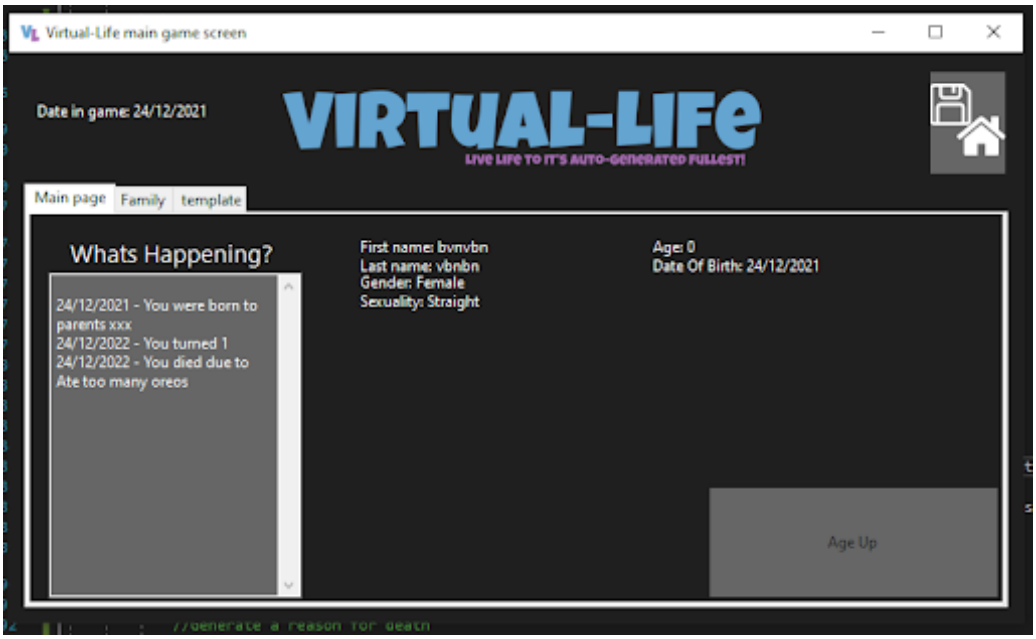


Figure 4.15: Main game screen showing death event and age up button disabled

I then moved onto the killing algorithm of the other characters, this will be done in a similar way to the increasing age of them - looping through the array and if they aren't already dead then passing them through the death checker then if they are to die, passing them through a kill procedure which will make use of the same `generateReasonForDeath` function. This was very easy to program as I had already worked out the fundamentals of this in the main character killing functions, so I was able to copy, paste and adapt the code to suit the `genericFamilyMember` objects.

```
public void checkKillGenericFamilyMember()
{
    //loop through all members of familyArray, passing their age into deathCheck. if return=true then kill, else move onto next.

    for(int x=0; x<controlClass.NextFamily; x++)
    {
        if(familyArray[x].LivingStatus == true)
        {
            bool deathCheckReturn = deathCheck(familyArray[x].Age);
            if(deathCheckReturn == true)
            {
                //RIP - put death code here
                //MessageBox.Show("DEATH" + x.ToString());
                familyArray[x].DateOfDeath = controlClass.InGameDate.AddDays(randomNumber(1, 300));
                familyArray[x].LivingStatus = false;
                familyArray[x].ReasonForDeath = generateReasonForDeath();
                //Now gen event
                eventArray[controlClass.NextEvent].Category = "Death";
                eventArray[controlClass.NextEvent].DateHappened = controlClass.InGameDate;
                eventArray[controlClass.NextEvent].Description = familyArray[x].FirstName + " (your " + familyArray[x].RelationshipToMain.ToLower() + ") died due to " + familyArray[x].ReasonForDeath;
                controlClass.NextEvent++;
                //now reduce mc happiness score
                mainCharacterScores.HappinessScore = mainCharacterScores.HappinessScore - randomNumber(1, 100);
                saveToFile();
            }
```

```
        }
    }
}
```

Listing 4.33: Death algorithm for a non-main character

For the purposes of testing, I added in a line of code which will force the character to die. This is omitted from the code above.

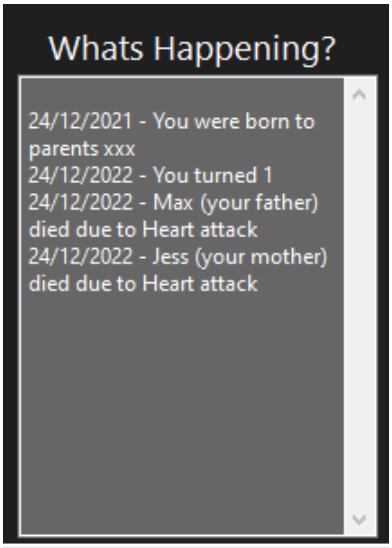


Figure 4.16: Event box on main game screen showing someone has died

I then included more reasons for death. My reasons for death array now contains the following:

```
string[] reasonsForDeath = { "a heart attack", "eating too many oreos", "
    ↪ being clawed at by cat" , "slipping on a banana peel", "falling in the
    ↪  otter enclosure at a zoo and being licked to death", "dropping a
    ↪ toaster into the bath", "being crushed by a falling grand piano", "
    ↪ being strangled by a fez tassel", "walking into a cactus", "getting
    ↪ hit by a train", "being clubbed by old lady in the street with her
    ↪ walker", "being crushed by a stampede of hungry campers"};
```

Listing 4.34: Reasons for death array

4.15 Random Generation Of a Main Character

At this point in developmental testing, I was creating a new game relatively frequently, to allow me to test different aspects. It takes about a minute to setup the main character in the new game form, even with just entering gobbledygook. To improve on this time, I created a random generation button with the following code behind it.

```
private void btnRandom_Click(object sender, EventArgs e)
{
    Random rnd = new Random();
    int gender = rnd.Next(1, 3);
    switch (gender)
    {
        case 1:
            //male
            txtFirstName.Text = Person.genMFN();
            cboGender.Text = "Male";
            break;
        case 2:
```

```
            //female
            txtFirstName.Text = Person.genFFN();
            cboGender.Text = "Female";
            break;
        default:
            txtFirstName.Text = "Sam";
            break;
    }
    txtLastName.Text = Person.genLN();
    int hair = rnd.Next(1,5);
    int eye = rnd.Next(1, 6);

    switch (hair)
    {
        case 1:
            cboHairColour.Text = "Blonde";
            break;
        case 2:
            cboHairColour.Text = "Brown";
            break;
        case 3:
            cboHairColour.Text = "Red";
            break;
        case 4:
            cboHairColour.Text = "Black";
            break;
        default:
            cboHairColour.Text = "Blonde";
            break;
    }

    switch (eye)
    {
        case 1:
            cboEyeColour.Text = "Brown";
            break;
        case 2:
            cboEyeColour.Text = "Blue";
            break;
        case 3:
            cboEyeColour.Text = "Green";
            break;
        case 4:
            cboEyeColour.Text = "Gray";
            break;
        case 5:
            cboEyeColour.Text = "Amber";
            break;
        default:
            cboEyeColour.Text = "Brown";
            break;
    }

}
```

Listing 4.35: Random generation of main character algorithm

This works as expected and produces a random character.

4.16 Scores

Before I programmed any more of the game, I decided that I would need to add in the code for my scores. I first added the `Score` class.

```
public class Score
{
    private int educationScore;
    private int jobScore;
    private int happinessScore;
    private int medicalScore;
    private int lifeScore;

    public int EducationScore { get; set; }
    public int JobScore { get; set; }
    public int HappinessScore { get; set; }
    public int MedicalScore { get; set; }
    public int LifeScore { get; set; }
}
```

Listing 4.36: Score class

I then added the code to the constructor of the create new game form which will create a new object called `mainCharacterScores`.

```
public Score mainCharacterScores = new Score();
```

Listing 4.37: Declaration of the mainCharacterScores object

I then added the the initialisation code with placeholder values.

```
//setup the scores object for the main (use random generation to calculate
    ↪ values)
mainCharacterScores.EducationScore = 0;
mainCharacterScores.JobScore = 0;
mainCharacterScores.HappinessScore = 0;
mainCharacterScores.MedicalScore = 0;
mainCharacterScores.LifeScore = 0;
```

Listing 4.38: Setting the score values to placeholder values

Now, I need to add the code to write this new object out to a file (on the `MainGameScreen` form)

```
File.WriteAllText(controlClass.Filepath + "\\mainCharacterScores.json",
    ↪ JsonConvert.SerializeObject(mainCharacterScores));
```

Listing 4.39: Code to write the Scores object out to a file

To test this first, I created a new game then saved it and viewed the `mainCharacterScores.Json` file in Notepad++. I saw what I was expecting

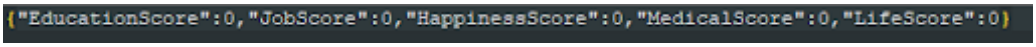


Figure 4.17: Screenshot of Notepad++ showing scores object

Now I can write the data out to a file, I need a way to read it back into the game. I added the code needed in the show saved games form for this.

```
Score mainCharacterScores = JsonConvert.DeserializeObject<Score>(File.
    ↪ ReadAllText(filepath + "\\mainCharacterScores.json"));
```

Listing 4.40: Code to read mainCharacterScores back in

I then added the code in to show the life score in a message box, so I could test opening the same saved game files.

```
MessageBox.Show(mainCharacterScores.LifeScore.ToString());
```

Listing 4.41: Test code to make sure code above is working

This worked as expected and produced the following result

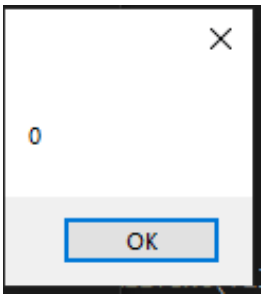


Figure 4.18: Proof the scores are being read back into the program correctly

I then removed the code for the message box because now that I know that the file management is working, I can now add in the algorithms for all these values and the displays in the game window for them. To start, I will add in the code to the create new game form to randomly generate the initial score values. I will be using an instance of the `Random` class called `rnd` to generate the scores. Each score will have a maximum value of 100.

```
//setup the scores object for the main (use random generation to calculate
    ↪ values)
mainCharacterScores.EducationScore = rnd.Next(0, 101);
mainCharacterScores.JobScore = rnd.Next(0, 101);
mainCharacterScores.HappinessScore = rnd.Next(0, 101);
mainCharacterScores.MedicalScore = rnd.Next(0, 101);
mainCharacterScores.LifeScore = rnd.Next(0, 101);
```

Listing 4.42: Setting the score values random values

To display this information, I will take inspiration from one of the games which I was playing during research - using a bar to represent the information. I will use the control ProgressBar as they are easy to set the value of and a nice visual representation of the value.

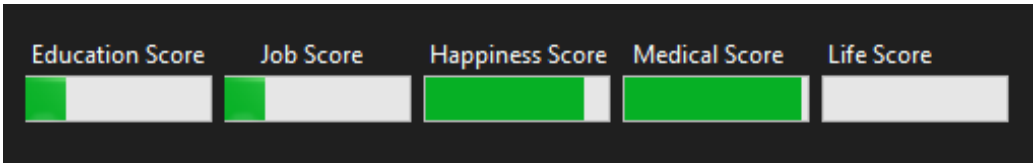


Figure 4.19: Progress bar controls containing the score values

(Colours and placement are subject to redesign)
I then added the code to make each of the progress bars fill depending on the value of the score.

```
public void updateScoresPB()
{
    prbEducationScore.Value = mainCharacterScores.EducationScore;
    prbJobScore.Value = mainCharacterScores.JobScore;
    prbHappinessScore.Value = mainCharacterScores.HappinessScore;
    prbMedicalScore.Value = mainCharacterScores.MedicalScore;
    prbLifeScore.Value = mainCharacterScores.LifeScore;
}
```

Listing 4.43: Code to display the score in the progress bar

As with most of my program, I am breaking it down into smaller functions or procedures as I will need to use the code in many different places.

4.17 Education

Now I have my scores and displaying of said scores working, I can begin writing the code which will alter the scores. The first score I will program is the education score. To start off, I setup the code which will calculate the next 1st September based off of the character's date of birth then work out what year they should be moving into in school.

```
public void checkEducationStatus()
{
    int age = mainCharacter.Age;
    DateTime mcBirthday = mainCharacter.DateOfBirth;
    DateTime dateOfEvent;
    if (mcBirthday.DayOfYear < 245)
    {
        //mc is born before 2nd sept therefor events need to be for this
    ↪ september
        int yearOfEvent = controlClass.InGameDate.Year;
        int monthOfEvent = 9;
        int dayOfEvent = 1;
        dateOfEvent = new DateTime(yearOfEvent, monthOfEvent, dayOfEvent);
    }
    else
    {
        //mc is born after 2nd sept therfore events need to be for the
    ↪ following september
        int yearOfEvent = (controlClass.InGameDate.Year) + 1;
        int monthOfEvent = 9;
        int dayOfEvent = 1;
        dateOfEvent = new DateTime(yearOfEvent, monthOfEvent, dayOfEvent);
    }
    //need to get date of 1st september after mcs birthday
    switch (age)
    {
        case 4:
            //move into reception
            eventArray[controlClass.NextEvent].Category = "Education";
            eventArray[controlClass.NextEvent].DateHappened = dateOfEvent;
            eventArray[controlClass.NextEvent].Description = "Move into year
    ↪  4";
            controlClass.NextEvent++;
            break;
```

Listing 4.44: Procedure to check the education status of the main character

For testing, I have just used the first year - age 4 where the character will be entering reception. The test code I wrote works and displays the following. Where it says 'move into year4', this should read 'move into reception'. This is not an program error, this is a typing error on my part.

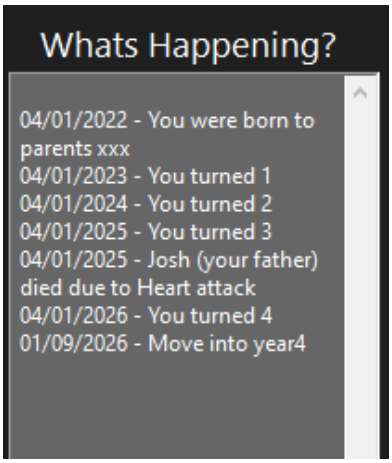


Figure 4.20: Event box showing first school event

I then setup the page of the TabControl which will display the education information. It is basic for the moment, this may be improved at a later point.

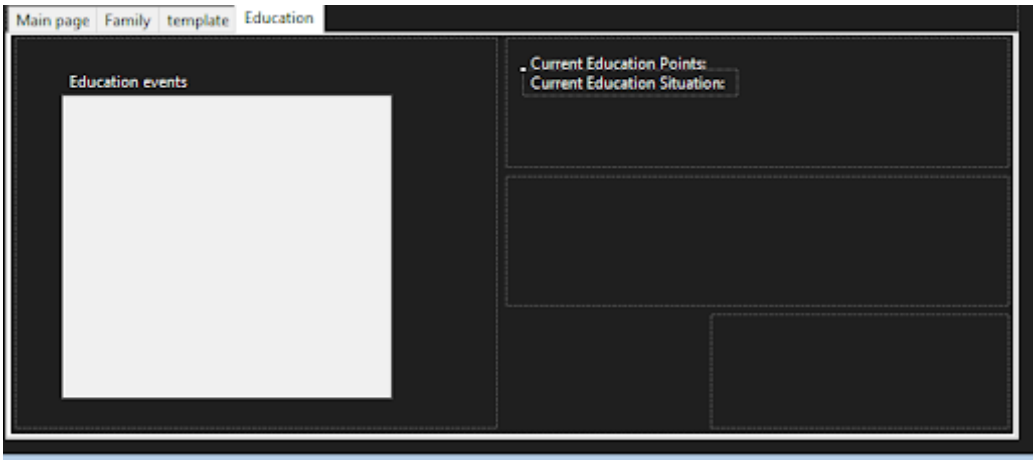


Figure 4.21: TabControl showing education page

Now, I will add the code to populate the form.

```
public void updateEducationPage()
{
    //first update the events box
    txtEduEducationEvents.ResetText();
    for (int i = 0; i < controlClass.NextEvent; i++)
    {
        if(eventArray[i].Category == "Education")
        {
            txtEduEducationEvents.Text = txtEduEducationEvents.Text +
    ↪ Environment.NewLine + eventArray[i].DateHappened.ToShortDateString() +
    ↪  " - " + eventArray[i].Description;
        }

    }

    //now update the lables
    lblEduCurrentEducationPoints.Text = "Current Education Points: " +
    ↪ mainCharacterScores.EducationScore.ToString();
    lblEduEducationSituation.Text = "Current Education Situation: At school
    ↪ - " + controlClass.NameOfCurrentSchool;
```

```
17      lblEduCurrentYear.Text = "Current Year: " + controlClass.
    ↪ CurrentSchoolYear.ToString();
18  }
```

Listing 4.45: Code to populate the education page on the main game form

Which will produce the following result, again please ignore the typo in the 'move into year4' line.

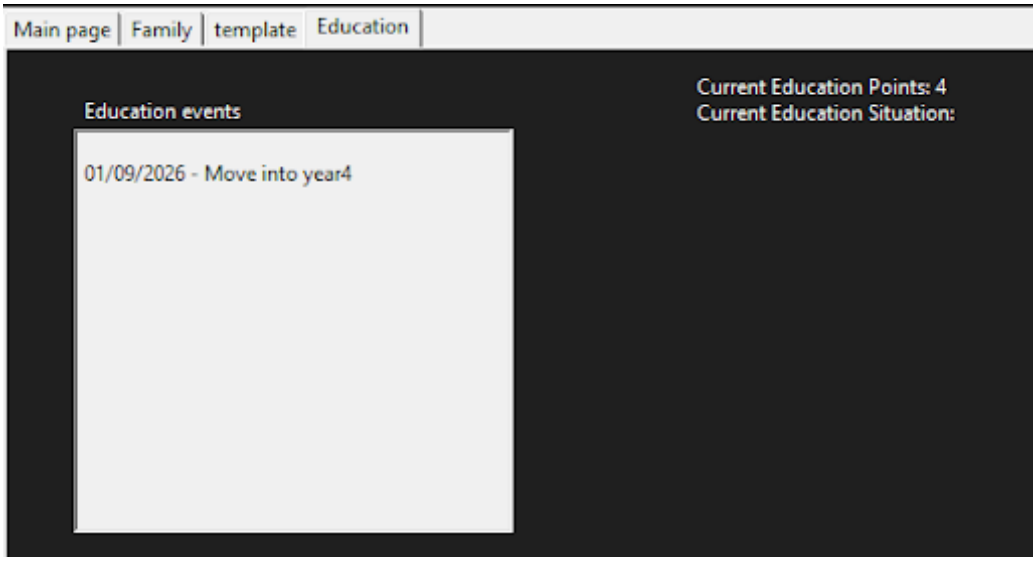


Figure 4.22: Populated education page on main form

Now, I have the basics working, I need to go through each of the years and add in the events. These will be hard coded for now to save time. If I get more time, I will add in some customisation for this from the players perspective.
Shown below is the code for each age of the character. Only shown below is for when the character is 4.

```
1   case 4:
2       //move into reception
3       eventArray[controlClass.NextEvent].Category = "Education";
4       eventArray[controlClass.NextEvent].DateHappened = dateOfEvent;
5       eventArray[controlClass.NextEvent].Description = "Start infant school in
    ↪  Reception.";
6       controlClass.NextEvent++;
7       mainCharacterScores.EducationScore = mainCharactedScores.EducationScore
    ↪ + randomNumber (1,4);
8       break;
```

Listing 4.46: First improvement to the education generation procedure

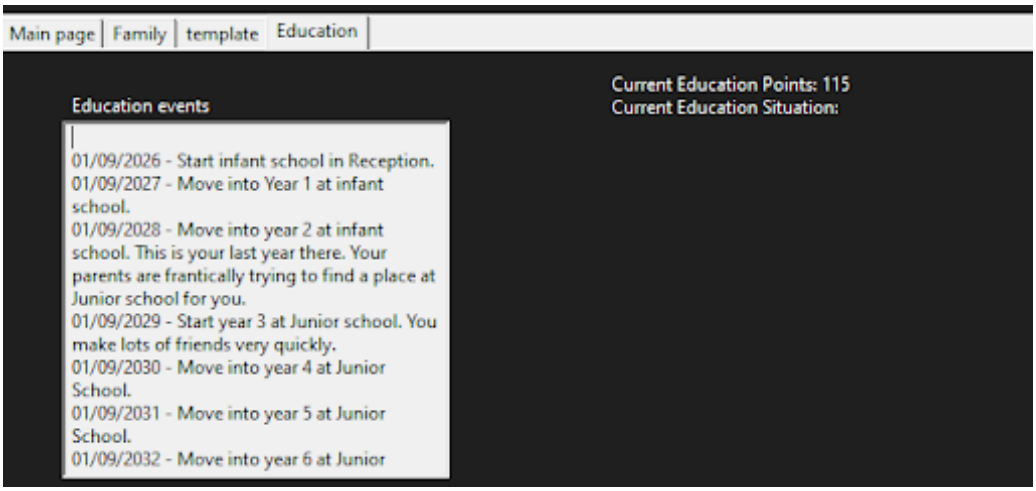


Figure 4.23: Output of the first improvement of the education system

I got an error during testing where the score value was greater than the max value of the progress bar. This is caused by the progress bar control which displays the education score having a maximum value of 100, and the education score value being set to 101 in the code.
To resolve this, I added in an if statement to the update progress bars method.

```
public void updateScoresPB()
{
    if (mainCharacterScores.EducationScore < prbEducationScore.Maximum &&
    ↪ mainCharacterScores.EducationScore > prbEducationScore.Minimum)
    {
        prbEducationScore.Value = mainCharacterScores.EducationScore;
    }
    if(mainCharacterScores.HappinessScore < prbHappinessScore.Maximum &&
    ↪ mainCharacterScores.HappinessScore > prbHappinessScore.Minimum)
    {
        prbHappinessScore.Value = mainCharacterScores.HappinessScore;
    }
    if (mainCharacterScores.MedicalScore < prbMedicalScore.Maximum &&
    ↪ mainCharacterScores.MedicalScore > prbMedicalScore.Minimum)
    {
        prbMedicalScore.Value = mainCharacterScores.MedicalScore;
    }
    if (mainCharacterScores.LifeScore < prbLifeScore.Maximum &&
    ↪ mainCharacterScores.LifeScore > prbLifeScore.Minimum)
    {
        prbLifeScore.Value = mainCharacterScores.LifeScore;
    }
}
```

Listing 4.47: Improved updateScoresPB procedure

This only updates the progress bar if the education score is less than the maximum value and greater than the minimum value of the progress bar.
To progress further with the education generation, I need to now program the choice box form. This will allow the user to select one of three pre-determined options on a new form.

4.18 Choice Box Form

I began by designing the form. All the label controls will be populated programmatically as part of the construction of the form when it is needed; this is why they have been left as the default placeholder text.

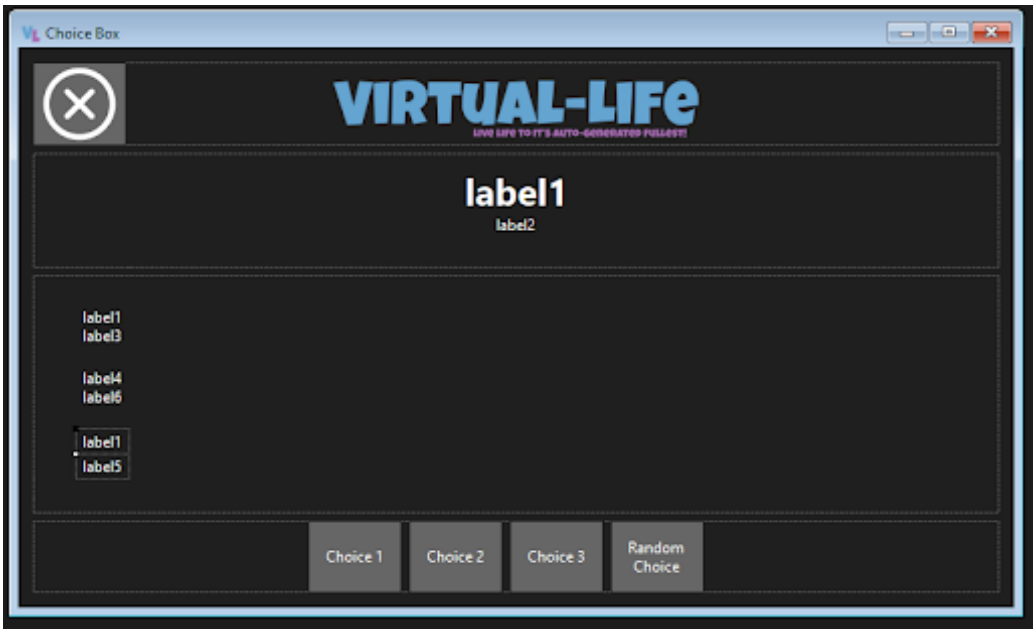


Figure 4.24: Choice Box form design

Now, I will add the code to pass some test data into the form from my test button on the main game form. Below is the code on `frmMainGameScreen` which calls the choice box.

```
using (var form = new frmChoiceBox("text here"))
{
    var result = form.ShowDialog();
    if (result == DialogResult.OK)
    {
        string val = form.ReturnValue1; //values preserved after close
        //Do something here with these values
        MessageBox.show(val);
    }
}
```

Listing 4.48: Code which generates the choice box form and recieves its response

Below is the code on the choice box form which responds to the user pressing `btnChoice1`.

```
public partial class frmChoiceBox : Form
{
    public string ReturnValue1{get; set; }
    string headerHere;
    public fromChoiceBox(string header)
    {
        InitializeComponent();
        headerHere = header;
    }

    private void frmChoiceBox_Load(object sender, EventArgs e)
    {
        MessageBox.show(headerHere);
    }

    private void btnChoice1_Click(object sender, EventArgs e)
    {
        this.ReturnValue1 = "Something";
        this.DialogResult = DialogResult.OK;
        this.Close();
    }
```

```
22 }
```

Listing 4.49: Choice box form code to send the result to where it was called from

When running this, I will expect to see a message box containing "Text here", then the form will appear. When I click the `btnChoice1`, I expect the form to close then a message box to appear containing the word "something".
I will now run the program and check what happens. First, this message box appears

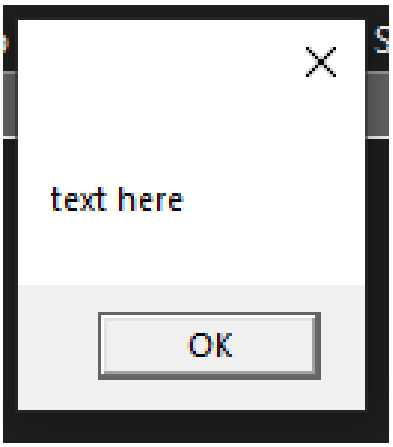


Figure 4.25: Testing choice box 1a

Then the form appears

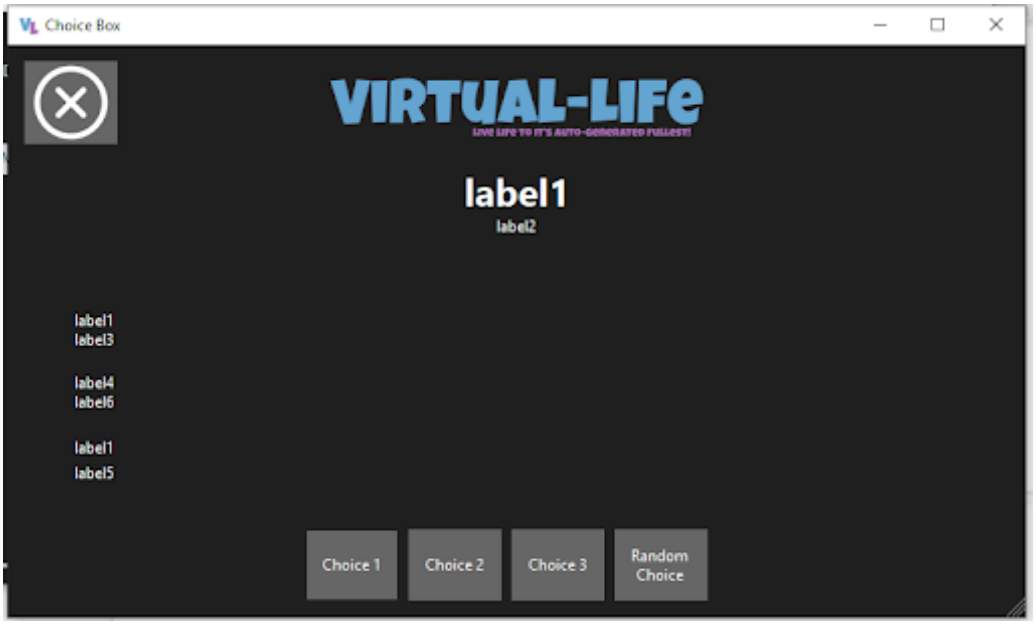


Figure 4.26: Testing choice box 1b

Then when I click choice 1, the form closes then this message box appears.

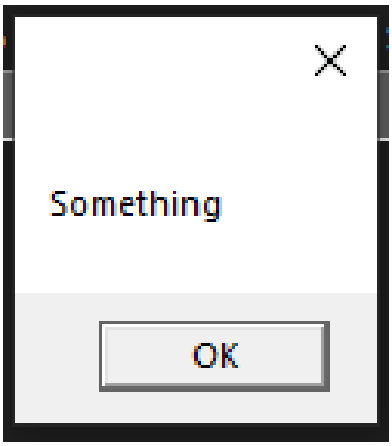


Figure 4.27: Testing choice box 1c

This means that this code is working and I can progress onto adding more data passed into the form's constructor.
First though, I need to stop the user from selecting anything on the `mainGameScreen` form while the choice box is active. I could either disable all the controls on it or I could just hide the form then re-show it when it's needed. I'm going to use the latter. To test this, I added in the code to hide the main game form once it had loaded the choice form, then when a response is received from the choice form, re-show the main form.

```
using (var form = new frmChoiceBox("text here"))
{
    this.Hide();
    var result = form.ShowDialog();
    this.Show();
    if (result == DialogResult.OK)
    {
        string val = form.ReturnValue1; //values preserved after close
        //Do something here with these values
        MessageBox.show(val);
    }
}
```

Listing 4.50: Code which generates the choice box form and receives its response now with show and hide functions

Now, I need to write the constructor function which will generate the box and process the box. I am writing it this way as I will potentially need to use this form multiple times in one game, meaning I will need to have a reusable piece of code which can be used to generate it. I added the code needed to send text into and return a value from the choice box. I am expecting a message box to show containing 556 when I close out of the choice box.

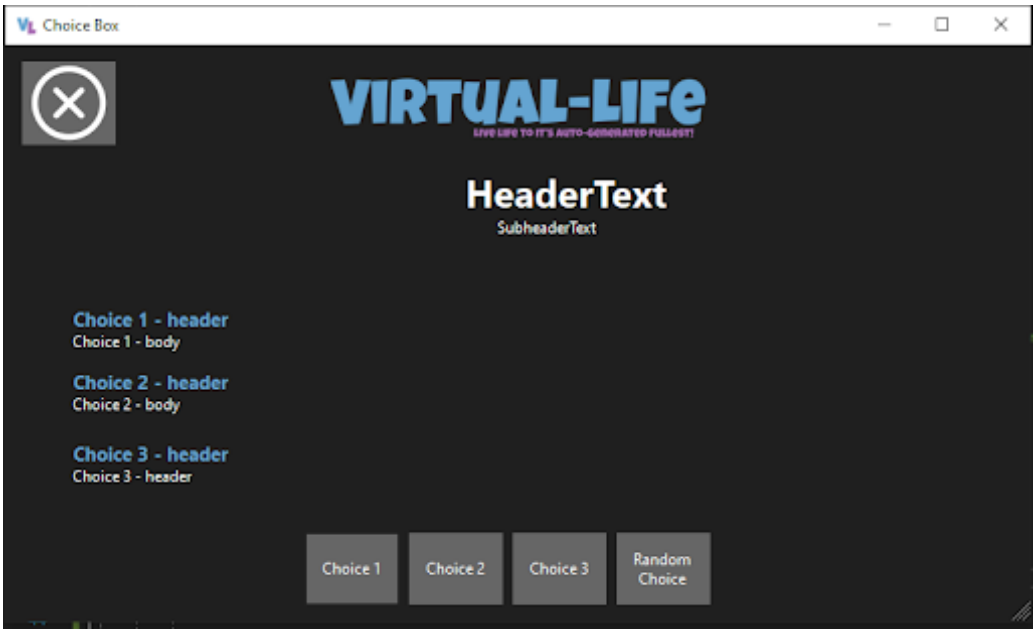


Figure 4.28: Form with data passed into it programmatically

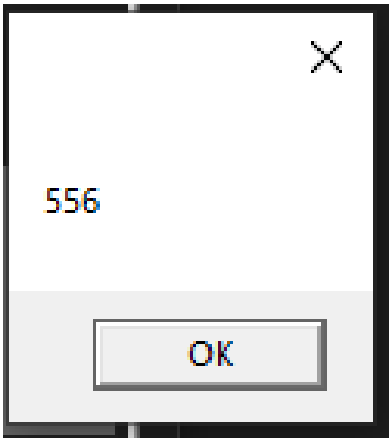


Figure 4.29: Result of pressing button "Choice 1"

This means I have a working choice box, for choice 1 at least. Now I will program the other 4 buttons and test them. I have written the following code to return the value of the button that the user presses.

```
private void btnChoice1_Click(object sender, EventArgs e)
{
    this.ReturnValue1 = 1;
    this.DialogResult = DialogResult.OK;
    this.Close();

}

private void btnChoice2_Click(object sender, EventArgs e)
{
    this.ReturnValue1 = 2;
    this.DialogResult = DialogResult.OK;
    this.Close();
}

private void btnChoice3_Click(object sender, EventArgs e)
{
```

```
18      this.ReturnValue1 = 3;
19      this.DialogResult = DialogResult.OK;
20      this.Close();
21  }
22
23  private void btnRandomChoice_Click(object sender, EventArgs e)
24  {
25      Random rnd = new Random();
26      int random = rnd.Next(1, 4);
27      this.ReturnValue1 = random;
28      this.DialogResult = DialogResult.OK;
29      this.Close();
30  }
31
32  private void button1_Click(object sender, EventArgs e)
33  {
34      this.DialogResult = DialogResult.Cancel;
35      this.Close();
36  }
```

Listing 4.51: Code which returns the outcome of the choice box

This is working as expected.
Now I have a working choice box which I can feed data into, I need to use it to make some decisions in the game. I am first going to use it when the character turns 4 and has to decide what school to go to.

4.19 Generating names error

While generating a new game for testing, I came across a bug which is where I have set the upper bound of a random generation too high for the array it is in. This has been a recurring issues with each of the three name generation functions. For example, I have given 31 as the upper bound for an array which contains exactly 30 items, when you take into account the zero based nature of arrays in C#, the error is generated.
To fix this error, I have reduced the upper bound to 29. This is working.

4.20 Random Generation of Names of Schools

Now I have the choice box and education system vaguely worked out. I can begin to randomly generate names of schools. I added the following code to my program

```
1  public int choiceBox(int typeOfChoice, string opt1, string opt2, string opt3
   ↪ )
2  {
3      /*
4       * Will need to pass the following into this function and into the
   ↪ choice box form.
5       * -type of choice (this will determine the header and subheader text)
6       * --> using this rather than passing all information in to simplifiy
   ↪ calls to this function.
7       * -opt1
8       * -opt2
9       * -opt3
10      */
11
12      //declare vars used for passing data to choice box
13      string headerText = "HeaderText";
14      string subheaderText = "SubheaderText";
15      string opt1header = "Choice 1";
```

```
    string opt1body = opt1;
    string opt2header = "Choice 2";
    string opt2body = opt2;
    string opt3header = "Choice 3";
    string opt3body = opt3;


    //first setup the typeOfChoice
    switch (typeOfChoice)
    {
        case 1:
            //choose infant school
            headerText = "Infant school";
            subheaderText = "Select the infant school you want to spend the next 3 years at";
            break;
        case 2:
            //choose junior school
            headerText = "Junior school";
            subheaderText = "Select the junior school you want to spend the next 4 years at";
            break;
        case 3:
            //choose secondary school
            headerText = "Secondary school";
            subheaderText = "Select the secondary school you want to spend the next 5 years at";
            break;
        case 4:
            //choose college
            headerText = "College";
            subheaderText = "Select the college you want to spend the next 2 years at";
            break;
        default:
            //no option found
            break;
    }

    using (var form = new frmChoiceBox(headerText, subheaderText, opt1header, opt1body, opt2header, opt2body, opt3header, opt3body))
    {

        this.Hide();
        var result = form.ShowDialog();

        this.Show();
        if (result == DialogResult.OK)
        {
            int val = form.ReturnValue1;            //values preserved after close
            //Do something here with these values

            //for example

            return (val);
        }

    }
    return 0;
```

```
70 }
```

Listing 4.52: Random generation of schools code

When I ran it for the first time, I got the following output. This is erroneous as there is no option for choice 3.

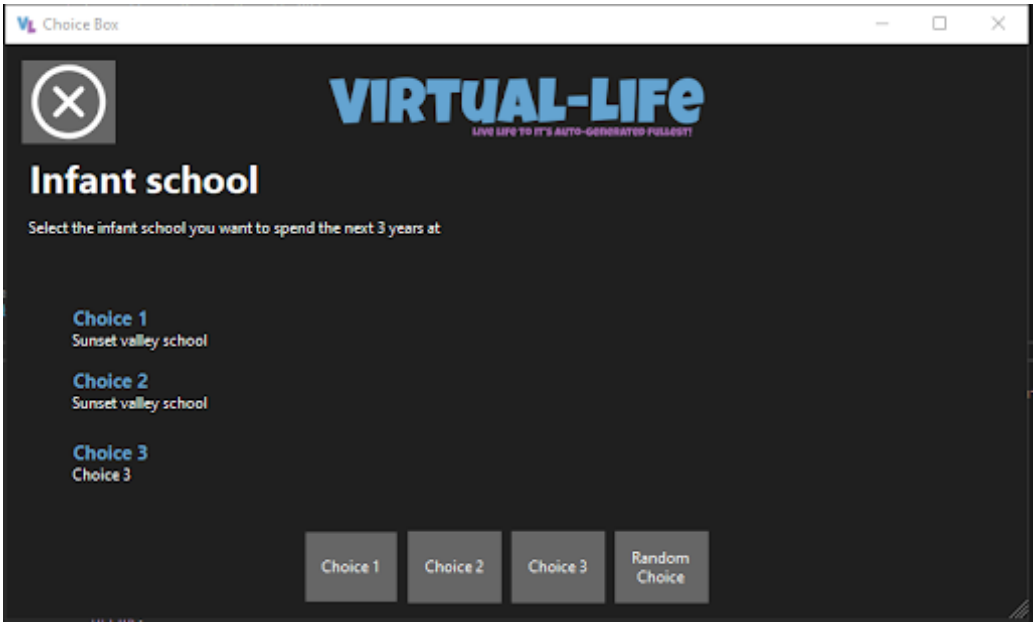


Figure 4.30: Output from first run of Random School Generation Code

I don't mind that choices 1 and 2 are the same as this could add some comedic value to the game. On debugging, I was able to find a typo in the constructor for the choice box form.

```
1 opt3body = opt3h;
```

Listing 4.53: Erronious line

```
1 opt3body = opt3b;
```

Listing 4.54: Corrected line

This correction worked, and I got the result which I was expecting.

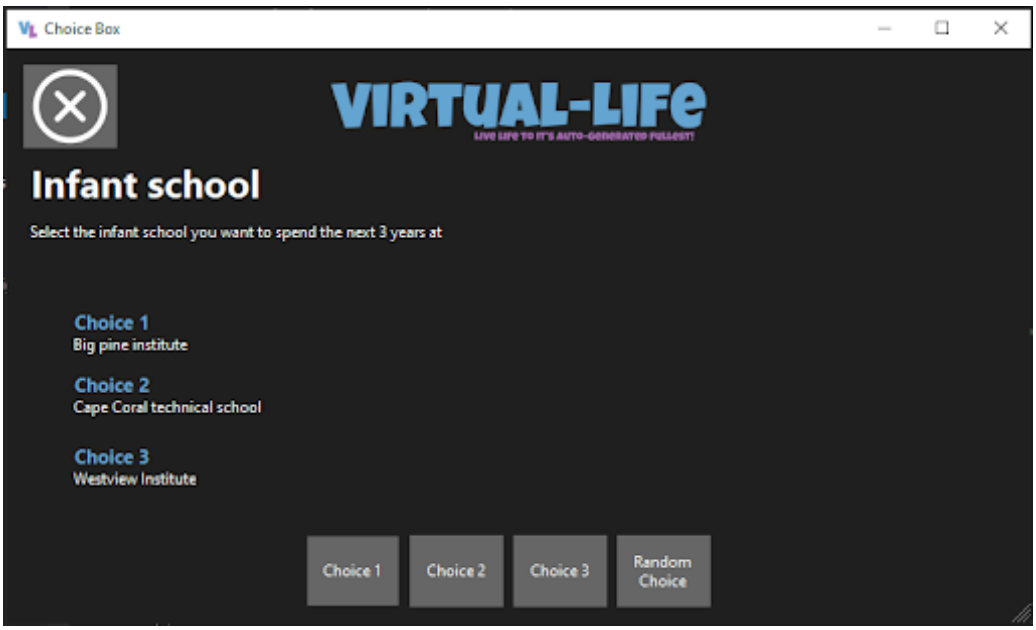


Figure 4.31: Correct output from random school name generation code

Now I have the school name generation worked out for the infant school, I can re-use the functions to generate names of schools for junior and secondary.
After adding the code to make this work. I tested this and it works as expected.

4.21 Textboxes

During testing, I was finding it annoying having to scroll down through the events text box and education text box to find the latest event. To solve this, I've changed the code which adds the latest event to the text box from

```
txtEventBox.Text = txtEventBox.Text + Environment.NewLine + eventArray[i].
    ↪ DateHappened.ToShortDateString() + " - " +  eventArray[i].Description;
```

Listing 4.55: Old method of adding text to textbox

to

```
txtEventBox.AppendText(Environment.NewLine + eventArray[i].DateHappened.
    ↪ ToShortDateString() + " - " + eventArray[i].Description);
```

Listing 4.56: New method of adding text to a textbox

This has solved my problem and now the textbox scrolls down as the new content is added to it.

4.22 Withdraw from Education

Part of my design is having a way for the character to withdraw from Education.
First, I added the button onto the form which will allow this to happen.

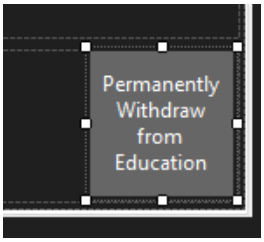


Figure 4.32: Permanently withdraw from education button

I then added the code to support the button

```
private void btnEduWithdraw_Click(object sender, EventArgs e)
{
    mainCharacter.InEducation = false;
    eventArray[controlClass.NextEvent].Category = "Education";
    eventArray[controlClass.NextEvent].DateHappened = controlClass.
    ↪ InGameDate;
    eventArray[controlClass.NextEvent].Description = "You permanently
    ↪ withdrew from education. You said it was too boring.";
    controlClass.NextEvent++;
    mainCharacterScores.EducationScore = 0;
}
```

Listing 4.57: Withdraw from Education button code

I then added an if statement to the checkEducationStatus procedure to check if the character was currently in education or not.
This feature works. On the click of age up directly following pressing the withdraw button it gives you the following event:

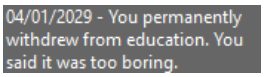


Figure 4.33: Event generated after pressing Withdraw button

4.23 Being Born Event

Currently, when a new game is started, the first event generated states You were born to parents xxx. Now I have my parent generation code written, I can add in the parent's names. To do this, I will need to move the code which generates the event from the middle of the new life algorithm to the end, this will make sure that the parents have been generated before their information is inputted to the event array.
I also need to change the code which generates the event so it has the names in it.

```
eventArray[controlClass.NextEvent].Category = "Life Event";
eventArray[controlClass.NextEvent].DateHappened = DateTime.Today;
eventArray[controlClass.NextEvent].Description = "You were born to parents "
    ↪  + familyArray[0].FirstName + " and " + familyArray[1].FirstName;
controlClass.NextEvent++;
```

Listing 4.58: Improved code for the being born event

This worked the first time I ran the code and gave me the result I was expecting.

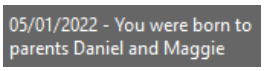


Figure 4.34: Event generated after being born

4.24 Generate Event Procedure

It is quite laborious having to write 4 lines of code to generate a single event. As I am using this more and more, I am going to write a procedure which will do it for me.
I used my standard 4 lines of code and copied them into a procedure then wrote the test code to test it with.

```
public void generateEvent(string category, string content, DateTime date)
{
    eventArray[controlClass.NextEvent].Category = category;
    eventArray[controlClass.NextEvent].DateHappened = date;
    eventArray[controlClass.NextEvent].Description = content;
    controlClass.NextEvent++;
}
```

Listing 4.59: Code for generateEvent procedure

This worked the first time I ran it and produced the result I was expecting.

4.25 Scores

Now I have some of the bigger elements of the game programmed, I can start thinking more about the scores. To do this, I need to generate modifiers; this is where the `ControlClass` will be used to store these modifiers. Then I will be able to write my `lifeScore` algorithm.
First, I am going to change how the scores are initially assigned to the character. The character's job score and education score will start off at zero, as in real life this is how it works. For now, I am also setting the medical score to 0 as I still need to write the algorithm which will calculate this and generate conditions for the characters. Happiness score will still be randomly generated as this is relatively random in life. After making these changes, this is what the declaration of the main character scores looks like:

```
mainCharacterScores.EducationScore = 0;
mainCharacterScores.JobScore = 0;
mainCharacterScores.HappinessScore = rnd.Next(0, 101);
mainCharacterScores.MedicalScore = 0;
mainCharacterScores.LifeScore = 0;
```

Listing 4.60: Initial score assignment to character

Now I am going to change the maximum value of each of the progress bar controls using the control property panel in Visual Studio to 1000, this will give me more room to manipulate the value.

```
public void calculateEducationScore()
{
    //Calculate the education score based off of the current score and the modifier
    //1 in 8 chance of the score being decreased rather than being increased. When decreased, needs to go down a fair chunk.

    int randomReturn = randomNumber(1, 8);
    if (randomReturn == 6)
    {
        //Decrease
        mainCharacterScores.EducationScore = mainCharacterScores.EducationScore - controlClass.EduScorePlus;
    }
    else
    {
        //increase
        //1 in 2 chance of modifier being upped by 0.3 of the current modifier.
        int randomReturn1 = randomNumber(1, 3);
        if(randomReturn1 == 1)
        {
            //normal modifier
            mainCharacterScores.EducationScore = mainCharacterScores.EducationScore + controlClass.EduScorePlus;
        }
        else if (randomReturn1 == 2)
        {
            //increased modifier
            int plus = (int)Math.Round(controlClass.EduScorePlus * 0.3);
            mainCharacterScores.EducationScore = mainCharacterScores.EducationScore + plus;
        }
    }

}
```

Listing 4.61: Original education score calculator

The first time I ran this code, the education score stayed extremely low the entire time. To combat this, I am going to change the decrease check to a chance of 1 in 30 and increase the original eduscoreplus to 10. This has improved the situation somewhat. I got a score of 92 at the point at which the character left school. I am still not happy though, as I was hoping it would be possible to max out the progress bar and keep scoring more. I am going to increase the beginning eduscoreplus value again, this time to 50. Again, this didn't produce the answer I wanted. So I doubled that value, giving me an eduscoreplus value of 100. This worked much better and produced me a score of 840 when the character turned 18. I am satisfied with this algorithm and can now begin to look at the happiness score algorithm.
The happiness score is going to be a mixture of various factors which will be combined together to create one score. Initially, the factors I am going to start with is an age based calculation, with the

thinking that when a person is younger they will be happier (however, there will be a chance that this is a low score)
I will first write the function to update the score.

```
public void updateHappinessScore(int externalModifier)
{
    //add to the current happiness score the generic value to increse by
    ↪ plus a modifier
    mainCharacterScores.HappinessScore = mainCharacterScores.HappinessScore
    ↪ + controlClass.HappinessScorePlus + externalModifier;

}
```

Listing 4.62: Happiness score calculation algorithm

On running this code, it isn't increasing the value. On investigation, I have found that I didn't change a line of code in the generate initial value section.

```
public int generateHappinessScorePlusVal()
{
    //generate the happinessScorePlus value.
    //random chance that this is a low plus score initially,
    Random rnd = new Random();
    int randomChance = rnd.Next(1, 30);
    int val = 10;
    if (randomChance == 4)
    {
        //goes up slowly
        val = rnd.Next(1, 10);
    }
    else
    {
        //goes up quickly
        val = rnd.Next(11, 30);
    }
    return 0;
}
```

Listing 4.63: Generation of happiness score modifier algorithm

The return value shouldn't be set to 0, it should be set to the integer val.
This is now working and on running the code, I realised that the value is increasing far too quickly. To solve this, I changed the goes up quickly value to 11-30. This is working much better, with a considerably slower increase allowing me to change factors more throughout the rest of the implementation stage.
Now, I need to add the code which will decrease the happiness score a random amount when a family member dies.

```
//now reduce mc happiness score
mainCharacterScores.HappinessScore = mainCharacterScores.HappinessScore -
    ↪ randomNumber(1, 100);
```

Listing 4.64: Code to reduce the main character happiness score once a family member has died

For now, this is all the implementation of scores that I can do. To be able to implement the medical and job scores, I will need to to write the medical condition algorithms and job algorithms.

4.26 More displaying of scores

The progress bars which I am using to display the scores are good at a visual representation of the max and min values and where the player is; but they don't give accurate representations. To solve

this, I am going to use a tooltip which will display the score.
This is the result of the code shown below.

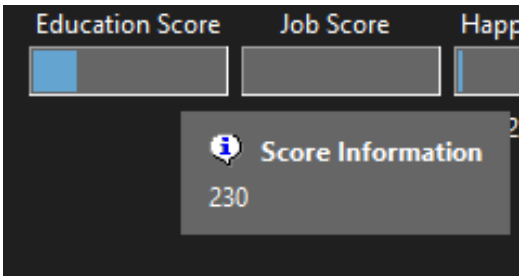


Figure 4.35: Scores progress bar with ToolTip visible

This is the code which is run when the progress bars are updated.

```
ttMainToolTip.SetToolTip(prbEducationScore, prbEducationScore.Value.ToString
    ↪ ());
ttMainToolTip.SetToolTip(prbHappinessScore, prbHappinessScore.Value.ToString
    ↪ ());
ttMainToolTip.SetToolTip(prbJobScore, prbJobScore.Value.ToString());
ttMainToolTip.SetToolTip(prbLifeScore, prbLifeScore.Value.ToString());
ttMainToolTip.SetToolTip(prbMedicalScore, prbMedicalScore.Value.ToString());
```

Listing 4.65: Code to populate ToolTips showing score information

4.27 Disable clickables

During testing, I realised that I have some clickable items (buttons etc) which are still enabled when the main character dies. To solve this, I am going to write a procedure which will disable all the clickable controls when it is run.
To do this, I will need a new procedure, shown below.

```
public void disableClickables()
{
    //procedure to disable all clickable elements on this form, for use when
    ↪  the main char dies.
    //There is no way to reverse this.

    btnAgeUp.Enabled = false;
    btnEduWithdraw.Enabled = false;
}
```

Listing 4.66: Disable clickables subroutine

As the comments say, there is no way to overturn this procedure as this is only to be used at the very end of the game when the character is dead.
This worked the first time I tried this, shown below

Figure 4.36: Buttons disabled after the procedure above has been run

I am satisfied with this procedure as it works as expected.

4.28 About Form

This form is used to credit the assets I have downloaded from the internet. There are no programmatically elements to it apart from opening it. All design will be done in Visual Studio's designer. Below is the form.

Figure 4.37: About form design

4.29 Crimes

Now I need to add the code in which will let the player generate a random crime for their player to commit. It will be a very basic system, which gives the player no choice on what crime they commit and all the sentences will be between 1 and 12 years community service. For added humour, I won't be disabling this button until the character reaches a certain age as I feel it would be more amusing to have a 2 year old stealing a car.
See below for the code which runs the page on the tab control.

```
private void btnCriCommitCrime_Click(object sender, EventArgs e)
{
    //first generate new crime then time of community service for it.

    string[] crimes = { "Burgle a shop", "Steal a car", "Pirate a DVD" };
    int crimesLength = crimes.Length;
    int rand = randomNumber(0, crimesLength);
    int years = randomNumber(1, 12);
    string crimeText = "You committed a crime " + crimes[rand] + " and you
    ↪ recieved " + years.ToString() + " years community service.";
    generateEvent("Crime", crimes[rand], controlClass.InGameDate);
    updateCrimes();
}

public void updateCrimes()
{
    txtCriCrimeEvents.ResetText();
    for (int i = 0; i < controlClass.NextEvent; i++)
    {
        if (eventArray[i].Category == "Crimes")
        {
```

```
21              txtCriCrimeEvents.Text = txtCriCrimeEvents.Text + Environment.
    ↪ NewLine + eventArray[i].DateHappened.ToShortDateString() + " - " +
    ↪ eventArray[i].Description;
22          }
23
24      }
25  }
```

Listing 4.67: Procuedres which are run when the "Comit Random Crime" button is pressed

This initially didn't work as expected as the crime only textbox doesn't show crimes however the main event box does. To rectify the issue with the crime only box, I corrected the line of code in `updateCrimes()` in the if statement from Crimes to Crime. This works and produces the results I want.

Figure 4.38: Crime event box

4.30 Customising Character

To start working on this section, I need to design the form.

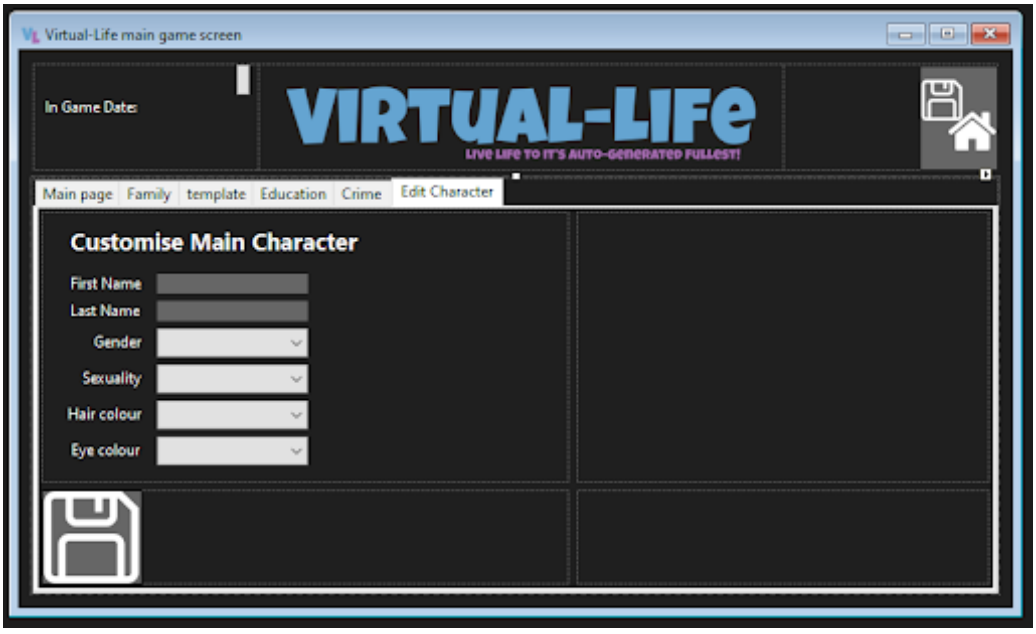


Figure 4.39: Customise character options

This just has the customisation options at the moment and the save button. I still need to add the end life section to the right hand side of the screen. Now I need to add the code which will unlock these options once the main character has turned 13.

```
public void yearlyCheck()
{
    //this is a place which will run every time the mc ages up. If they are older than the specified age then will run code in relevant section.

    if(mainCharacter.Age >= 13)
    {
        //'unlock' character customisation screen.
        txtEditFirstName.Enabled = true;
        txtEditLastName.Enabled = true;
        cboEditGender.Enabled = true;
        cboEditSexuality.Enabled = true;
        cboEditEyeColour.Enabled = true;
        cboEditHairColour.Enabled = true;
    }
}
```

Listing 4.68: Yearly Check subroutine

I have done this by writing a new procedure yearlyCheck() which gets run every time the player clicks age up. It will force the character customisation options to be enabled once the character turns 13. Next, I need to write the code which will populate these controls with the current information which the main character has.

```
public void updateCharCustomisationOptions()
{
    //fill all the Character Customisation option controls with current information from main char object
    txtEditFirstName.Text = mainCharacter.FirstName;
    txtEditLastName.Text = mainCharacter.LastName;
    cboEditGender.Text = mainCharacter.Gender;
    cboEditSexuality.Text = mainCharacter.Sexuality;
    cboEditHairColour.Text = mainCharacter.HairColour;
    cboEditEyeColour.Text = mainCharacter.EyeColour;
}
```

Listing 4.69: Subroutine to update character customisaiton options

This procedure is called as part of the `updateForms` procedure so it will be called every time age up is clicked.
Now I need to write the save button algorithm.

```
public void saveCharCustomisationOptions()
{
    //if the contents of the text boxes etc is NOT null and is different to the current contents of the main character object, then it needs to be written into the object

    //start with the first name
    if(txtEditFirstName.Text != null && txtEditFirstName.Text != mainCharacter.FirstName)
    {
        mainCharacter.FirstName = txtEditFirstName.Text;
    }
    //next - do the last name
    if(txtEditLastName.Text != null && txtEditLastName.Text != mainCharacter.LastName)
    {
        mainCharacter.LastName = txtEditLastName.Text;
    }
    if(cboEditGender.Text != null && cboEditGender.Text != mainCharacter.Gender)
    {
```

```
17          mainCharacter.Gender = cboEditGender.Text;
18      }
19      if(cboEditSexuality.Text != null && cboEditSexuality.Text !=
    ↪ mainCharacter.Sexuality)
20      {
21          mainCharacter.Sexuality = cboEditSexuality.Text;
22      }
23      if(cboEditHairColour.Text != null && cboEditHairColour.Text !=
    ↪ mainCharacter.HairColour)
24      {
25          mainCharacter.HairColour = cboEditHairColour.Text;
26      }
27      if(cboEditEyeColour.Text != null && cboEditEyeColour.Text !=
    ↪ mainCharacter.EyeColour)
28      {
29          mainCharacter.EyeColour = cboEditEyeColour.Text;
30      }
31  }
```

Listing 4.70: Save character customisation procedure

This half worked the first time I tried it. The edit aspect worked and the main display tab updated as I expected it to, however when I tried a null entry, the program allowed this to happen.

Figure 4.40: Textbox on Customisation form

Figure 4.41: Main info point on main game screen

To resolve this, I changed `null` to `""`. This worked and I now have a working character customisation page.
Next, I need to add the end life button. This will still generate a random reason for death. First, I added the controls to the form then I added the code which will power it.

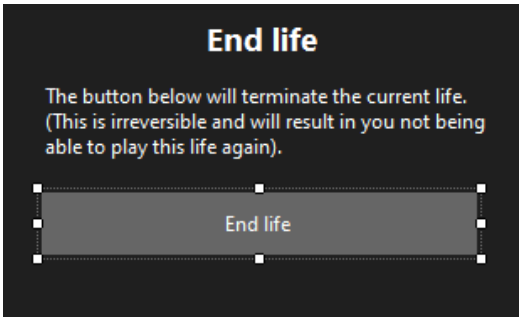


Figure 4.42: Button which ends the main characters life

```
1  private void btnEndLife_Click(object sender, EventArgs e)
2  {
3      DialogResult dialogResult = MessageBox.Show("Are you sure you want to
    ↪ end this life?", "End life", MessageBoxButtons.YesNo);
4      if (dialogResult == DialogResult.Yes)
```

```
5       {
6           killMainCharacter();
7       }
8       else if (dialogResult == DialogResult.No)
9       {
10          //do something else
11      }
12  }
```

Listing 4.71: End life algorithm

As you can see from the code, it calls the same procedure that would be called if the main character was to die randomly.

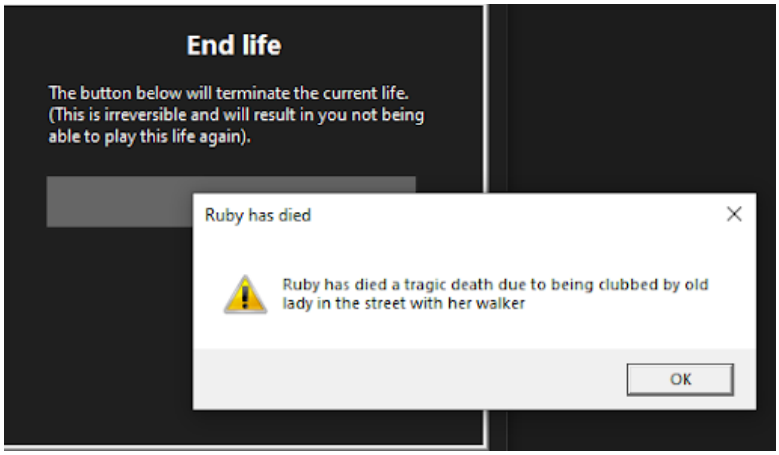


Figure 4.43: Output from pressing the end life button

This worked as expected.

4.31 Jobs

The next part of the game which I am going to program is the jobs and employment feature. To start off, I am going to follow my design and create a new tab page on the main tab control with the controls outlined in my design.

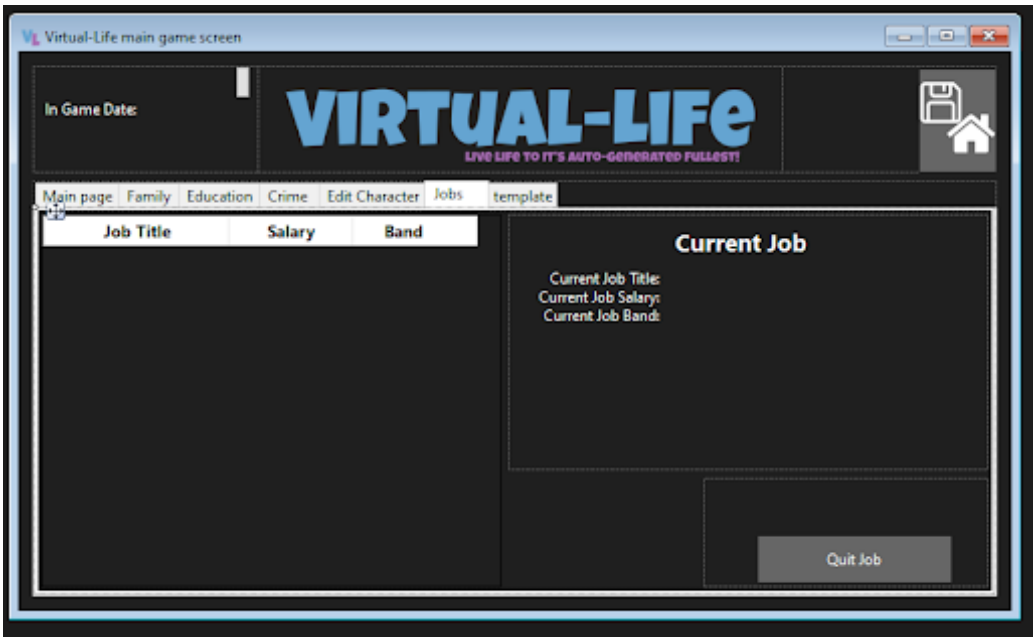


Figure 4.44: Design of the jobs form

I then began to write the algorithm to generate jobs.

```
public Job generateJob (int band)
{
    string[] band1Job = { "Cleaner", "Wearhouse opperative", "Grocery picker ", "Sahara delivery driver", "Burger flipper", "Hospital porter", " Community care worker", "Care assistant", "Handyperson", "Nursery assistant" };
    string[] band2Job = { "College security guard", "Pet groomer", "Engineer ", "Secretary", "Line cook", "Plumber", "Cleaning company owner", "HR advisor", "Vehicle Technician", "Tanker driver" };
    string[] band3Jobs = { "Supervisor", "Senior Vehicle Technician", "Head of Resources", "Product manager", "Web developer", "Comms manager", " Software engineer", "Project manager", "Site manager", "Sales Director " };
    string[] band4Jobs = { "Store manager", "Finance Manager", "Site supervisor", "Nursing home manager", "IT Solutions Architect", "Senior  API Developer", "Electrical Installation Lecturer", "Matron", " Defects manager", "Family Lawyer" };
    string[] band5Jobs = { "CEO", "Retail park manager", "Hotel manager", " Specialty doctor", "User interface designer", "SQL Developer", "DevOps  engineer", "Senior Software consultant", "Executive head teacher", " Hospital director" };
    Job newJob = new Job();
    Random rnd = new Random();
    switch (band)
    {
        case 1:
            Salary = 17000;
            Band = 1;
            JobTitle = band1Job[rnd.Next(1, 9)];
            break;
        case 2:
            Salary = 34000;
            Band = 2;
            JobTitle = band2Job[rnd.Next(1, 9)];
            break;
        case 3:
```

```
            Salary = 51000;
            Band = 3;
            jobTitle = band3Jobs[rnd.Next(1, 9)];
            break;

        case 4:
            Salary = 68000;
            Band = 4;
            jobTitle = band4Jobs[rnd.Next(1, 9)];
            break;
        case 5:
            Salary = 150000;
            Band = 5;
            jobTitle = band5Jobs[rnd.Next(1, 9)];
            break;
    }
    return newJob;
}
```

Listing 4.72: Algorithm to generate jobs

I wrote some test code which would use a test button to generate a job and output the job title in a message box.

```
private void btnTestButton_Click(object sender, EventArgs e)
{
    Job newJob = new Job();
    newJob.generateJob(1);
    MessageBox.show(newJob.JobTitle);
}
```

Listing 4.73: Procedure to call the generateJob function

When I ran the code, it gave me the following result; this means that my job generation code is working.

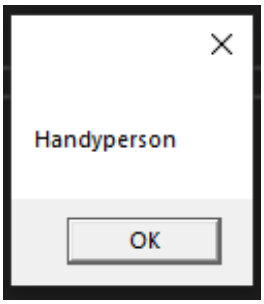


Figure 4.45: Result from pressing test button

Next, I need to write the algorithm which once the main character turns 18 generates band 1 jobs and populates the data grid view which they will be stored in. This is going to be done using a procedure - reshuffle Jobs. This is going to also be able to be called from the button which will reshuffle jobs. This button will initially be disabled, however once the main character turns 18, it will be enabled (in the same way as the edit character controls). To start writing `reshuffleJobs`, I will first generate the array of Job objects each containing different band jobs.It will then output these jobs to a `DataGridView` control.

```
public void reshuffleJobs(int band)
{
    //first generate the jobs
    int maxNumb = 10;
    for (int i = 0; i < maxNumb; i++)
    {
```

```
        Job tempJob = new Job(); //make new instance of the object
        tempJob.generateJob(band);
        availableJobs[i] = tempJob; //transfer the temp object into the
    ↪ array.
    }

    //now populate data grid view
    //first have to clear it
    dgvEmpAvailableJobs.Rows.Clear();
    dgvEmpAvailableJobs.Refresh();

    //use code from openldbws app to populate dgvEmpAvailableJobs

    //first, set var for which column contains which data
    int jtCol = 0;
    int salCol = 1;
    int bandCol = 2;

    //now loop through familyArray and populate dgvEmpAvailableJobs
    //as jobs are only ever generated and populated here - can use max
    ↪ number from job generation loop for loop here

    for (int x = 0; x < maxNumb; x++)
    {
        if (availableJobs[x].JobTitle != null)
        {
            int i = dgvEmpAvailableJobs.Rows.Add();
            dgvEmpAvailableJobs.Rows[i].Cells[jtCol].Value = availableJobs[x
    ↪ ].JobTitle;
            dgvEmpAvailableJobs.Rows[i].Cells[salCol].Value = availableJobs[
    ↪ x].Salary;
            dgvEmpAvailableJobs.Rows[i].Cells[bandCol].Value = availableJobs
    ↪ [x].Band;
        }
    }
}
```

Listing 4.74: Reshuffle available jobs algorithm

Which look like this

Job Title	Salary	Band
Community care worker	17000	1
Hospital porter	17000	1
Community care worker	17000	1
Grocery picker	17000	1
Grocery picker	17000	1
Care assistant	17000	1
Community care worker	17000	1
Hospital porter	17000	1
Hospital porter	17000	1
Wearhouse opperative	17000	1

Figure 4.46: Output of DataGridView control

This worked first time as I copied and altered the code from the generation and population of the

`genericFamilyMember` object array and display for that.
Now I need to be able to apply for the job and display information about the job. First I added the apply button to the main form, then I wrote the code which will populate the label controls (displaying the current job information) and save the job which the main character has selected to the main character object.

```
public void updateCurrentJobInformation()
{
    lblJobCurrentJobBand.Text = "Current Job Band: " + mainCharacter.JobBand
    ↪ .ToString();
    lblJobCurrentJobTitle.Text = "Current Job Title: " + mainCharacter.
    ↪ JobTitle.ToString();
    lblJobCurrentJobSalary.Text = "Current Job Salary: " + mainCharacter.
    ↪ JobSalary.ToString();

}

private void btnJobApplyForJob_Click(object sender, EventArgs e)
{
    mainCharacter.JobTitle = dgvEmpAvailableJobs.SelectedCells[0].Value.
    ↪ ToString();
    mainCharacter.JobSalary = Int32.Parse(dgvEmpAvailableJobs.SelectedCells
    ↪ [1].Value.ToString());
    mainCharacter.JobBand = Int32.Parse(dgvEmpAvailableJobs.SelectedCells
    ↪ [2].Value.ToString());
    updateCurrentJobInformation();
}
```

Listing 4.75: Applying for job and updating job information procedures

When I ran this for the first time, I got the following error: Object reference not set to an instance of an object with the displaying `JobTitle` line highlighted.
To resolve this, I removed the `.ToString()` method from the attributes which were already strings. This worked, however I hadn't added the attribute titles to the code, which means it gets removed when the button is pressed. This gives me the result of

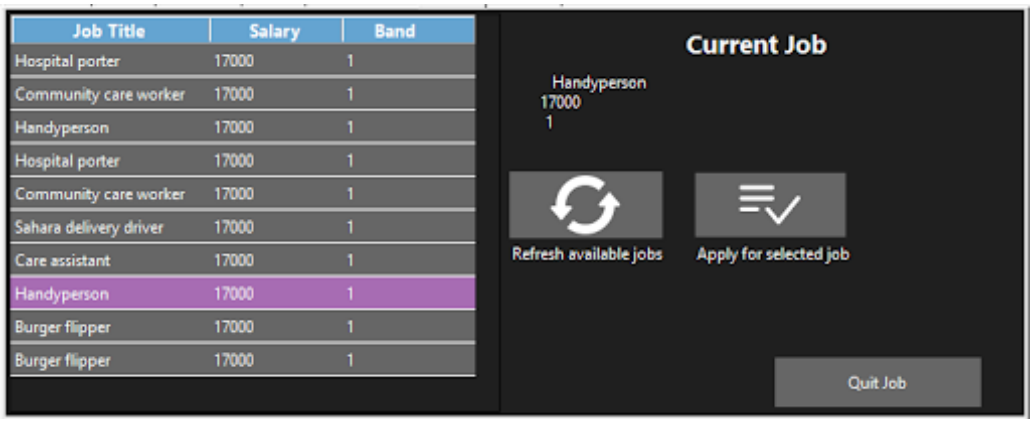


Figure 4.47: Job page of main form (without label titles)

Now I've added in the label titles, I get

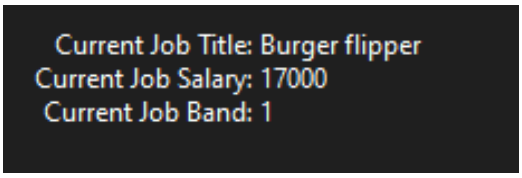


Figure 4.48: Correct display of job information

I then tried to click on the apply for select job before I had loaded the available jobs, this resulted in the following error.

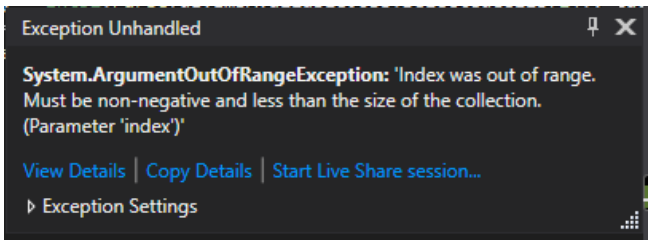


Figure 4.49: Error from not selecting a job before applying for it

To solve this, I added a `Try-Catch` block with the exception handled by displaying a message to the user.

```
 private void btnJobApplyForJob_Click(object sender, EventArgs e)
{
    try
    {
        mainCharacter.JobTitle = dgvEmpAvailableJobs.SelectedCells[0].Value.
    ↪ ToString();
        mainCharacter.JobSalary = Int32.Parse(dgvEmpAvailableJobs.
    ↪ SelectedCells[1].Value.ToString());
        mainCharacter.JobBand = Int32.Parse(dgvEmpAvailableJobs.
    ↪ SelectedCells[2].Value.ToString());
        mainCharacter.InJob = true;
        generateEvent("Job", "You applied for and got the job: " +
    ↪ mainCharacter.JobTitle, controlClass.InGameDate);
        updateCurrentJobInformation();
    } catch(Exception)
    {
        MessageBox.Show("Please select a job before applying for it.");
    }
}
```

Listing 4.76: Apply for job procedure now with try-catch block

This results in

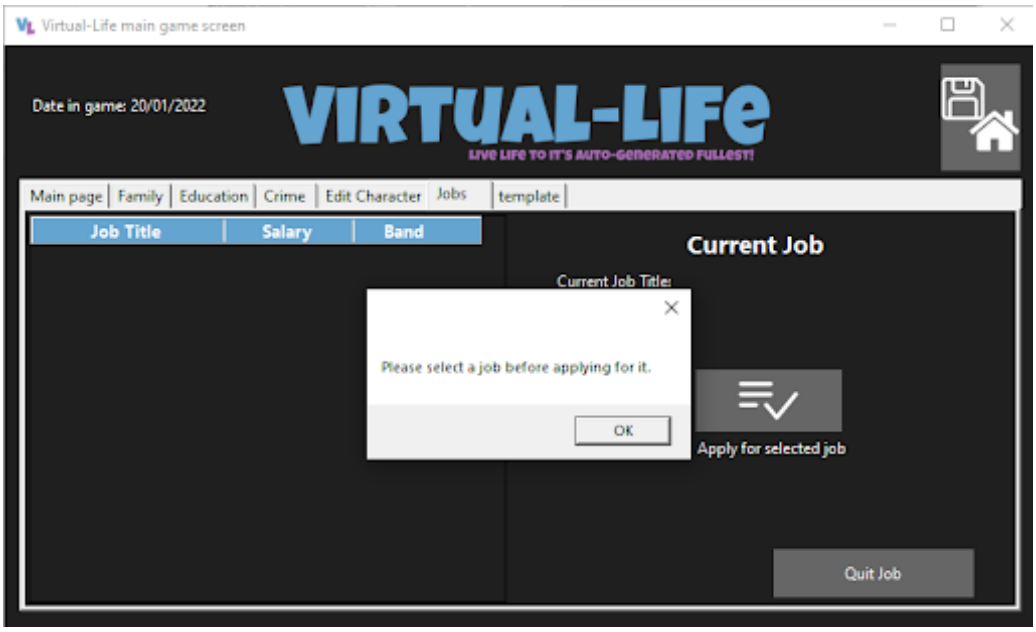


Figure 4.50: Result of applying for job without selecting a job after adding try-catch block

Now I need to add the quit job code. As this button will be enabled even if the main character isn't currently in a job, I will use a try catch block again; to catch any exceptions.

```
private void button1_Click(object sender, EventArgs e) //quit job
{
    try
    {
        mainCharacter.JobTitle = "";
        mainCharacter.JobSalary = 0;
        mainCharacter.JobBand = 0;
        mainCharacter.InJob = false;
        mainCharacter.JobCurrentDuration = 0;
        generateEvent("Job", "You quit your job", controlClass.InGameDate);
    }
    catch (Exception)
    {
        MessageBox.Show("Please make sure you have a job before you try to
    ↪ quit it.");
    }
    updateCurrentJobInformation();
}
```

Listing 4.77: Quit job algorithm

This worked as expected and gave me the results I wanted.

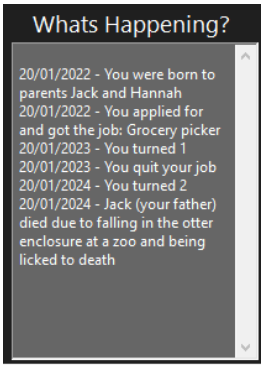


Figure 4.51: Result of quitting a job in the events box

Now that I have obtain and quit job code working, I need to write the job progression algorithms. This algorithm is going to work by looking at the `JobCurrentTime` attribute of the main character and if the character has been in their current job for over a certain time then they will be offered the next band of job up. To start this algorithm, I will change the generate job procedure so if the main character has been in a job for less than 5 years the game will generate band 1 jobs. This is the event handler code for when the button is pressed:

```
private void btnJobReshuffleJobBoard_Click(object sender, EventArgs e)
{
    if(mainCharacter.JobCurrentDuration < 5)
    {
        //can only get band 1 job
        reshuffleJobs(1);
    }
}
```

Listing 4.78: Start of the Reshuffle Job Board algorithm

This worked so I can now add the second band code

```
private void btnJobReshuffleJobBoard_Click(object sender, EventArgs e)
{
    if(mainCharacter.JobCurrentDuration <= 5)
    {
        //can only get band 1 job
        reshuffleJobs(1);
    }
    else if (mainCharacter.JobCurrentDuration >=6 && mainCharacter.
    ↪ JobCurrentDuration <= 14)
    {
        //get band 2 job
        reshuffleJobs(2);
    }
}
```

Listing 4.79: Second iteration of the reshuffle job board algorithm

This is also working and generating band 1 jobs while the main character has had a job for under 6 years then generating a band 2 job after that. As this is working, I can write the code to generate the rest of the bands of jobs.
This is working and producing the results I expect it to.
The next part of the jobs algorithm I need to program is the salary calculation This algorithm will be run every time the main character ages up, so the first thing which I need to do is check if the main character currently has a job. I am then able to write the code which will take the current salary, band and duration of which the main character has been in a job.
Below is the algorithm which will calculate the salary

```
public void calculateSalary()
{
    Random rnd = new Random();
    //algorithm to generate the salary of the main character
    //needs to take into consideration the current band of job, base band of
    ↪  that salary and how long the mc has been in that job
    int salary = mainCharacter.JobSalary;
    int band = mainCharacter.JobBand;
    int duration = mainCharacter.JobCurrentDuration;

    if(mainCharacter.InJob == true)
    {
        //mc is in job so can now calc salary.
        int increaseAmount = rnd.Next(0, (band*duration));
        mainCharacter.JobSalary = salary + increaseAmount;
```

```
    }
}
```

Listing 4.80: Calculate job salary algorithm

To test this, I ran the program and found that the increases are too marginal for the game to be fun. I will need to increase the max increase amount by a factor of 10.

```
int increaseAmount = rnd.Next(0, (band*duration)*10);
```

Listing 4.81: Salary increase amount calculation

This still isn't increasing the salary as much as I would like it to, so I am going to change the increase factor to 50. This is working much better and I am now happy with the salary calculation and salary increase algorithms. Now I have built the jobs segment of the game, I need to remove the test settings, so players can only get jobs when they turn 18. I am going to do this in the same way that I did for the character customisation options, making the controls on the form default to be disabled, then every time age up is clicked they are enabled.

```
if (mainCharacter.Age >= 18)
{
    //'unlock' character job options
    btnJobApplyForJob.Enabled = true;
    btnJobQuitJob.Enabled = true;
    btnJobReshuffleJobBoard.Enabled = true;
}
```

Listing 4.82: Age restriction for the jobs algorithms

This is working as I want it to. Finally, the job score will simply be the duration which you have had the job for. I have put the code for this in the `updateCurrentJobInformation` procedure as this is both run every time a change is made to the employment situation and whenever the main character is aged up.

```
mainCharacterScores.JobScore = mainCharacter.JobCurrentDuration*4;
prbJobScore.Value = mainCharacterScores.JobScore;
```

Listing 4.83: JobScore algorithm

4.32 Medical

This is going to be a very much behind the scenes algorithm which is going to randomly assign the medical score as well as enter some messages into the events box denoting what has happened medically. There will be no player interaction with this algorithm. It will first get the main character's current medical score, depending on what this is, the algorithm will generate a medical condition which will have a 5% chance of affecting the main character. All the medical conditions generated will either be life-long or will never need to be referenced again, therefore I can save some processing by not needing to keep detailed information about each medical condition, meaning the game will be more lightweight to run. Before I can write the algorithm, I'm going to have to change how the medical score is assigned when a new game is generated, it will have to be set to 100 not 0.

```
public void generateMedicalScore()
{
    Random rnd = new Random();
    int currentMedScore = mainCharacterScores.MedicalScore;
    if(currentMedScore == 100)
    {
        //has perfect health, 1/10 chance of it being lowered so can then be
    ↪  diseased.
        int chance = rnd.Next(0, 11);
        if(chance == 5)
        {
```

```
            currentMedScore = 99;
        }

    }
    if(currentMedScore < 100)
    {
        //can be diseased. 1 in 20 chance of getting disease
        int chance1 = rnd.Next(0, 21);
        if(chance1 == 12)
        {
            //is gonna be diseased
            string condition = getMedicalCondition();

            generateEvent("Medical","You " + condition, controlClass.
    ↪ InGameDate);

            mainCharacterScores.MedicalScore = mainCharacterScores.
    ↪ MedicalScore - rnd.Next(0, 50);

        }
    }
}

public string getMedicalCondition()
{
    string[] medicalConditions = {"broke your arm", "broke your leg", "
    ↪ stubbed your toe" };
    Random rnd = new Random();

    string returnString = medicalConditions[rnd.Next(0, medicalConditions.
    ↪ Length)];
    return returnString;
}
```

Listing 4.84: Medical score algorithms

I tested this algorithm and it worked.

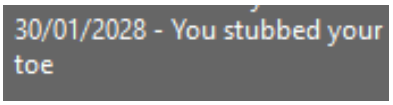


Figure 4.52: Result of medical condition generation

Now I just need to change the medical score value to 1000 and increase the decrease med score value.

4.33 Partners

The final main element of the game I need to program is the partner and courtship element. The courtship element will be an extremely simplified version of real life as every couple has a different path. To start off with, I need to program the generation of a potential partner. There will be no lower age limit on this and the partner will have to be within 2 years of the main character. This will be generated in much the same way as the generic family members when a new game is created. This is the code I have written in the main game screen form which is run whenever the button to generate potential partners is pressed.

```
private void btnParGenMore_Click(object sender, EventArgs e)
{
    generateAndPopulateMorePartners();
}
```

```
5
6  public void generateAndPopulateMorePartners()
7  {
8      //Now create family members
9      for (int i = 0; i < 10; i++)
10     {
11         DetailedCharacter tempFamily = new DetailedCharacter();
12         tempFamily.generateDetailedChar();
13         MessageBox.Show(tempFamily.FirstName);
14     }
```

Listing 4.85: Code to generate more potential partners

This calls the function in the detailed person class which looks like this.

```
1  public DetailedCharacter generateDetailedChar()
2  {
3      DetailedCharacter potential = new DetailedCharacter();
4      FirstName = "Dave";
5      return potential;
6  }
```

Listing 4.86: Detailed char class code to generate a partner

For testing, I have just programmed the first name to be Dave. This will allow me to make sure that everything is being called properly before I move onto the next part.
To test this so far, I can click the generate button and run the code.

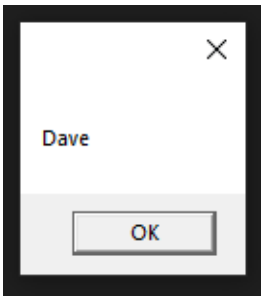


Figure 4.53: Outcome from pressing generate button

This gave me the outcome which I expected. I will now move onto the display feature of the potential partners. To do this, I will need to write the code which will add the generated temporary object to the array and write the code which will output the information to the user. The later I already have as I have been using `DataGridViews` lots throughout the project, and the former I have in a different state which I can use here.
After running the code and once again confirming that the correct number of `Daves` appeared, the `DataGridView` was populated as follows, confirming that the code is working this far.

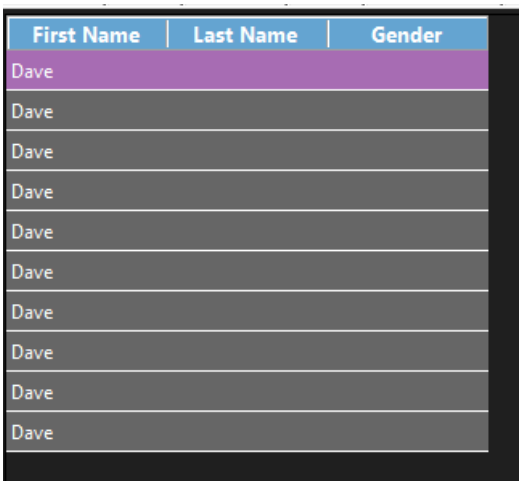


Figure 4.54: Initial test of the data grid view used to show potential partners

Now I have the display working, I have a way to test the development of the different potential partners. The different combinations of sexuality and gender will be generated within the detailed character class.
Initially, I will be using a straight woman to start development.

```
public DetailedCharacter generateDetailedChar(string gender, string
    ↪ sexuality, DateTime today)
{
    DetailedCharacter potential = new DetailedCharacter();

    if (gender == "Female")
    {
        if (sexuality == "Straight")
        {
            //generate a man
            FirstName = genMFN();
            LastName = genLN();
            Sexuality = "Straight";
            Gender = "Male";
            LivingStatus = true;
            DateOfBirth = today.AddDays(-rnd.Next(5840, 14600));
            Age = calcAge(DateOfBirth);
        }
    }
    FirstName = "Dave";
    return potential;
```

Listing 4.87: Code to generate potential partners for a straight woman

This worked as expected except for the fact I had left a line of debug code in from last time, hence all first names are Dave

First Name	Last Name	Gender
Dave	Grant	Male
Dave	Ross	Male
Dave	Robertson	Male
Dave	Brown	Male
Dave	Allan	Male
Dave	Watt	Male
Dave	Wood	Male
Dave	Simpson	Male
Dave	Bell	Male
Dave	Brown	Male

Figure 4.55: Second test of displaying potential partners in the DataGridView

First Name	Last Name	Gender	Date of Birth
Josh	Mitchell	Male	25/07/1983
Jack	Mitchell	Male	08/06/2001
Ethan	Murray	Male	26/08/1996
Oliver	Watson	Male	06/04/1985
Sebstian	King	Male	29/05/1993
Elijah	Wilson	Male	04/08/2005
Harvey	Miller	Male	17/11/1985

Figure 4.56: Final display of information about potneial partners in the DataGridView

This gives me the output which I am happy with.
The next thing I need to do is add the button which will initiate the relationship. To do this, I will need to add another column to the datagridview containing the index which the person is in the array, this means it will be really easy for me to access this and pull their information out of the `DataGridView`.
Below is the code to ask the person on a date

```
private void btnAskOnDate_Click(object sender, EventArgs e)
{
    //basically the same as applying for a job
    try
    {
        int index = Int32.Parse(dgvPartPotential.SelectedCells[4].Value.
    ↪ ToString());
        partner = potentialPartners[index];
        MessageBox.Show(partner.FirstName);
    }
    catch(Exception)
    {
        MessageBox.Show("Please select a partner before trying to go on a
    ↪ date with them.");
    }
    updatePartnerInformation();
```

```
}
```

Listing 4.88: Code to ask someone on a date

This didn't work at first, as I was trying to cast one object to another without the proper conversion process, to get around this, I have had to change the class type from `DetailedCharacter` to `Partner` in all the code in this section so far.
This works and produces the result I was expecting, the first name of the person in a message box.

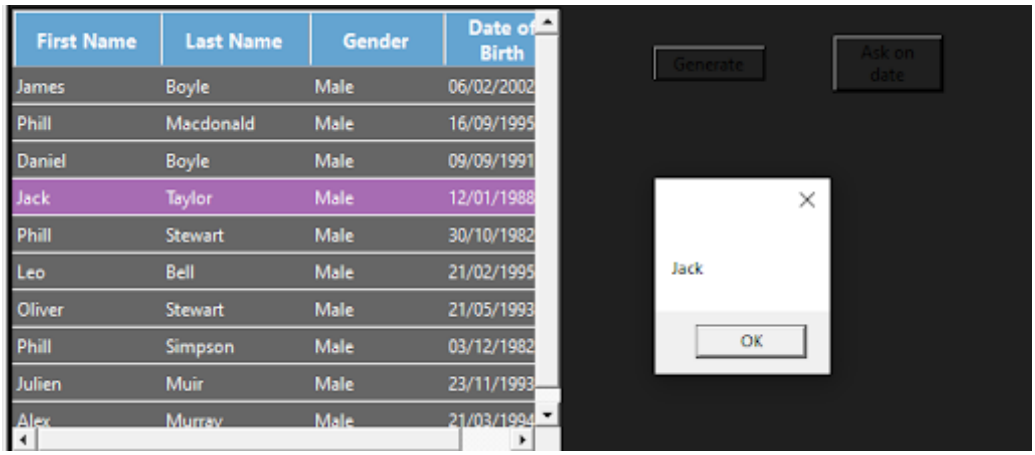


Figure 4.57: Output when clicking on ask partner on a date

Now I need to write the code which will generate the event for this

```
generateEvent("Relationships", "You asked " + partner.FirstName + " on a
    ↪ date and they said yes!", controlClass.InGameDate);
```

Listing 4.89: Code to create event saying that you have a new partner

Now I need to write the code to save the partner's information to a file. This worked first time and produced the result I was expecting in the JSON file.
Now I will write the code for all the different combinations of gender and sexuality to produce different partners.

```
public Partner generateDetailedChar(string gender, string sexuality,
    ↪ DateTime today)
{
    Partner potential = new Partner();

    if (gender == "Female")
    {
        if (sexuality == "Straight")
        {
            //generate a man
            FirstName = genMFN();
            LastName = genLN();
            Sexuality = "Straight";
            Gender = "Male";
            LivingStatus = true;
            DateOfBirth = today.AddDays(-rnd.Next(5840, 14600));
            Age = calcAge(DateOfBirth);
        }else if (sexuality == "Homosexual")
        {
            //generate woman
            FirstName = genFFN();
            LastName = genLN();
            Sexuality = "Homosexual";
            Gender = "Female";
```

```
            LivingStatus = true;
            DateOfBirth = today.AddDays(-rnd.Next(5840, 14600));
            Age = calcAge(DateOfBirth);
        }
    }
    if (gender == "Male")
    {
        if (sexuality == "Homosexual")
        {
            //generate a man
            FirstName = genMFN();
            LastName = genLN();
            Sexuality = "Homosexual";
            Gender = "Male";
            LivingStatus = true;
            DateOfBirth = today.AddDays(-rnd.Next(5840, 14600));
            Age = calcAge(DateOfBirth);
        }
        else if (sexuality == "Straight")
        {
            //generate woman
            FirstName = genFFN();
            LastName = genLN();
            Sexuality = "Straight";
            Gender = "Female";
            LivingStatus = true;
            DateOfBirth = today.AddDays(-rnd.Next(5840, 14600));
            Age = calcAge(DateOfBirth);
        }
    }


    return potential;
}
```

Listing 4.90: All the different combinations for generating different partners

This works as expected and generates me the combinations I was expecting it to. Now I can write the code which will display the partners information to the player.

```
public void updatePartnerInformation()
{
    if (mainCharacter.InRelationship == true)
    {
        lblPartFirstName.Text = "First Name: " + partner.FirstName;
        lblPartLastName.Text = "Last Name: " + partner.LastName;
        lblPartDateOfBirth.Text = "Date of Birth: " + partner.DateOfBirth.
    ↪ ToShortDateString();
        lblPartAge.Text = "Age: " + partner.Age.ToString();
    }
    else
    {
        lblPartFirstName.Text = "First Name: ";
        lblPartLastName.Text = "Last Name: ";
        lblPartDateOfBirth.Text = "Date of Birth: ";
        lblPartAge.Text = "Age: ";
    }
}
```

Listing 4.91: Code to update the current partners information to the player

This worked the first time I tried it as I have written very similar code many times before. Next, I will write the code which will allow the user to end the relationship with the partner.

```
private void btnPartDump_Click(object sender, EventArgs e)
{
    mainCharacter.InRelationship = false;
    partner.FirstName = null;
    partner.LastName = null;
    partner.Sexuality = null;
    partner.Gender = null;
    partner.LivingStatus = false;
    partner.Age = 0;
    updatePartnerInformation();
}
```

Listing 4.92: Dump partner algorithm

This worked as I was expecting it to. The confirmation that the partner had been dumped is that the partner information area is cleared.
Finally, for the partner section, I will write the code which will increase the partners age by 1 year.

```
if(mainCharacter.InRelationship == true)
{
    partner.Age = partner.Age + 1;
}
```

Listing 4.93: Increase age of partner

This also worked as I was expecting it to.

Chapter 5

Testing Post Development

Once I have finished programming my game, I will need to test all the elements of the app to ensure that I have (a) correctly programmed it, (b) there are no bugs, (c) the game meets my success criteria.

5.1 Final Test Plan

Test *Nº*	**Test Title**	**Details**	**Expected Outcome**	**Final Outcome**	**Vid** *Nº*
1a	Loading the main menu	Start the application and wait for the first form to load.	The main menu is the first form to load.	The main menu is the first form to load. Pass	1a
1b	Main menu buttons test 1	From the main menu, press the 'New Game' button.	The main menu form closes and the new game creation tool form opens.	The main menu form closes and the new game creation tool form opens. Pass	1b
1c	Main menu buttons test 2	From the main menu, press the 'Show saved games' button.	The main menu form closes and the show saved games form opens.	The main menu form closes and the show saved games form opens. Pass	1c
1d	Main menu buttons test 3	From the main menu, press the 'Settings' button.	The main menu form closes and the settings form opens.	The button clicks but nothing happens as there is no settings form. Fail	1d
1e	Main menu buttons test 4	From the main menu, press the 'About' button.	The main menu form closes and the about form opens.	The main menu form closes and the about form opens. Pass	1e
1f	Main menu accessibility test pt.1	From the main menu form, maximise the window using the Windows Menu Bar.	All controls on the form will increase in size and increase the spacing between them.	All controls on the form will increase in size and increase the spacing between them. Pass	1f
1g	Main menu accessibility test pt.2	From the main menu form, restore down the window using the Windows Menu bar.	The controls on the form should respond by shrinking and keeping their spacing according to the new size of the window	The controls on the form should respond by shrinking and keeping their spacing according to the new size of the window Pass	1g

Test *Nº*	**Test Title**	**Details**	**Expected Outcome**	**Final Outcome**	**Vid** *Nº*
2a	Generating new game with correct data	Create a new game, entering valid data and setting the save location. Then click the save button.	The app displays the main game screen form with the correct main character information displayed.	The app progresses to the main game screen. Pass	2a
2b	Generating new game with some incorrect data	Create a new game, entering the wrong data types into the text boxes and setting the save location. For example, entering 123 into the first name field.	The app progresses as normal; numbers for names are valid entry.	The app progresses to the main game screen. Pass	2b
2c	Generating new game leaving text boxes blank	Create a new game, without entering and information into any of the text boxes except save location.	The app will reject the input and instruct the player to enter the information or press the random button.	The app progresses to the main game screen. Fail	2c
2d	Random generation	Create a new game and use the random button and setting the save location.	The app populates all the fields on the form with random data and then progresses onto the main game screen form.	The app populates all the fields on the form with random data and then progresses onto the main game screen form. Pass	2d
2e	Re-roll random generation	Create a new game and use the random generation button, then press it again and setting the save location.	The app will generate two different sets of random information and display them sequentially. They should be different. The app will then progress onto the main game screen.	The app will generate two different sets of random information and display them sequentially. They should be different. The app then progresses onto the main game screen. Pass	2e
2f	New game form accessibility	Move between the controls without using mouse.	Click in the first textbox then navigate through the form by using keyboard only.	Had to click in first textbox to enter the tab chain. Was able to move between character information successfully, however unable to move into save location or select the save button so had to use mouse. Fail	2f

Test *Nº*	**Test Title**	**Details**	**Expected Outcome**	**Final Outcome**	**Vid** *Nº*
2g	New game form accessibility pt.2	From the main menu form, maximise the window using the Windows Menu Bar.	All the controls on the form should respond by growing and keeping their spacing according to the size of the window.	Some of the controls resized, however the majority didn't. Fail	2g
2h	New game form accessibility pt.3	From the main menu form, restore down the window using the Windows Menu bar.	The controls on the form should respond by shrinking and keeping their spacing according to the new size of the window	All controls responded by shrinking and sorting their spacing according to the new size of the window. Pass	2h
2i	Not specifying a save location	Generate a random game and click save without specifying a save location	The game rejects the input and displays a message box instructing users to select a save location.	The game rejects the input and displays a message box instructing users to select a save location. Pass	2i
3a	Loading from file with correct file placement	Load the game from the a folder with all the files intact and correctly placed.	The app will load the data then display the main game screen.	The app behaves as expected and opens the correct data. Pass	3a
3b	Loading from file with missing files	Load the game from a folder with missing files.	The app will say that files are missing and not load the game. It will clear the form ready for the next attempt.	The app crashes and displays a runtime error. Fail	3b
3c	Loading from file with a completely wrong folder	Load the game from a random folder that doesnt contain any of the correct files.	The app will say that files are missing and not load the game. It will clear the form ready for the next attempt.	The app crashes and displays a runtime error. Fail	3c
3d	Show saved games form accessibility test pt.1	From the main menu form, maximise the window using the Windows Menu Bar.	All the controls on the form should respond by growing and keeping their spacing according to the size of the window.	Some of the controls resized, however the majority didn't. Fail	3d
3e	Show saved games form accessibility test pt.2	From the main menu form, restore down the window using the Windows Menu bar.	The controls on the form should respond by shrinking and keeping their spacing according to the new size of the window.	All controls responded by shrinking and sorting their spacing according to the new size of the window. Pass	3e

Test *Nº*	**Test Title**	**Details**	**Expected Outcome**	**Final Outcome**	**Vid** *Nº*
4a	Pressing the save and home button	From the main game screen, press the save and home button.	The app will save the data to the files then close the main game screen form and open the main menu form.	The app saved the data to the files then close the main game screen form and open the main menu form. Pass	4a
4b	Pressing the 'age up' button	Click the age up button.	The age up algorithm is followed correctly; with no unnecessary alerts to the player. The scores should also be re-calculated and their displays updated.	The age up algorithm is followed correctly. No unnecessary alerts are displayed to the user. The score displays change. Pass	4b
4c	Changing tabs	Using the tab menu bar control, click from tab to tab.	When clicking from tab-to-tab, the correct controls only should be displayed.	When clicking from tab-to-tab, the correct controls only should be displayed. Pass	4c
4d	Main game screen accessibility test pt.1	From the main game screen form, maximise the window using the Windows Menu Bar.	All the controls on the form should respond by growing and keeping their spacing according to the size of the window	The controls on the form do nothing. Fail	4d
4e	Main game screen accessibility test pt.2	From the main game screen form, restore down the window using the Windows Menu bar.	The controls on the form should respond by shrinking and keeping their spacing according to the new size of the window	All controls responded by shrinking and sorting their spacing according to the new size of the window. Pass	4e
5a	Family tab - alive select	From the family tab, select an alive family member.	The family member information populates accordingly with 'N/A' for reason and date of death.	The family member information populates accordingly with 'N/A' for reason and date of death. Pass	5a
5b	Family tab - select dead	From the family tab, select a dead family member.	The family member information populates accordingly with correct reason and date of death	The family member information populates accordingly with correct reason and date of death. Pass	5b

Test Nº	**Test Title**	**Details**	**Expected Outcome**	**Final Outcome**	**Vid Nº**
6a	Choice box	Play though the game which will trigger the choice box to appear in time with education events.	The choice box appears at the correct times throughout the game with the correct information on it.	The choice box appears at the correct times throughout the game with the correct information on it. Pass	6a
6b	Education events are visible in the correct places	After entering the age bracket where education events are generated, the education events should appear both in the main event box and in the education specific box	The education events will appear in the main event box and in the education specific event box	The education events appear in the main event box and in the education specific event box. Pass	6b
6c	Education information is correct on the form	Before pressing age up, observe the education information and observe again after pressing age up	The education information (points, current situation) are correct on the form	The education information (points, current situation) are correct on the form. Pass	6c
6d	Permanently withdraw from education button prevents any more education events from being generated	Press the 'Permanently Withdraw From Education' button	No more education events will generate	No more education related events get generated. Pass	6d
7a	Commit random crime button	Press the 'Commit random crime' button	A random crime will be generated and displayed	A random crime is generated and displayed. Pass	7a
7b	Crime event box showing the correct data only	After pressing commit random crime button, observe the crime event box	Only crime events are displayed in the crime event box.	Only crime events are displayed in the crime event box. Pass	7b
8a	Edit character unlocks correctly	Before the main character is 13, the edit character options should be 'locked' and after they turn 13, they should be 'unlocked'	After main character turns 13, the edit character options unlock	After main character turns 13, the edit character options unlock. Pass	8a
8b	Edit character saves data when correct data is entered	Enter correct data into the edit character options, then press save.	The main game tab should update accordingly	The main game tab updates accordingly. Pass	8b

Test *Nº*	**Test Title**	**Details**	**Expected Outcome**	**Final Outcome**	**Vid** *Nº*
8c	Edit character rejects erroneous data	Empty one of the textbox controls on the edit character options, then press save	The game rejects the input.	The game rejects the input. Pass	8c
8d	end life works as expected press yes	Press the end life button then press 'yes' on the dialogue box	The game should trigger the end life algorithm and 'kill' the main character	The game triggers the end life algorithms and 'kills' the main character. Pass	8d
8e	end life works as expected. press no	Press the end life button then press 'no' on the dialogue box	The game should return to the edit character screen.	The game returns to the edit character screen. Pass	8e
9a	Refresh available jobs once	Press the 'Refresh available jobs' button	The game should generate a new set of jobs.	The game generates a new set of jobs. Pass	9a
9b	Refresh available jobs twice	Press the 'Refresh available jobs' button twice	The game should generate two different sets of jobs.	The game generates two different sets of jobs. Pass	9b
9c	Select an available job and apply	Highlight an available job and press the 'apply' button	The game should set the current job information to the job which has just been applied for and should generate a suitable event.	The game sets the current job information to the job which has just been applied for and generates a suitable event. Pass	9c
9d	Quit job	Press the quit job button	The game should clear the current job information.	The game clears the current job information. Pass	9d
9e	Apply for a new job while main character currently has one	Select and apply for a job while the main character currently has one	The game should reject this and display a message box to the player saying they need to quit their current job first.	The game allows the main character to apply for a job without quitting their current one first. Fail	9e
10a	Generate one set of potential partners	From the partners screen, press 'Generate more partners' button	The game should generate a new set of potential partners	The game generates a new set of potential partners. Pass	10a
10b	Generate two sets of potential partners	From the partners screen, press 'Generate more partners' button twice	The game should generate two different sets of potential partners.	The game generates two different sets of potential partners. Pass	10b
10c	Select a potential partner and take on date	From the partners screen, select a potential partner and take press take on date	The game should populate the current partner information and generate an event.	The game populates the current partner information and generates an event. Pass	10c

Test Nº	Test Title	Details	Expected Outcome	Final Outcome	Vid Nº
10d	Dump the partner	When you have a partner, press the 'dump' button	The game should clear the current partner information.	The game clears the current partner information. Pass	10d
11a	Generating partners after changing main characters gender and/or sexuality	After changing the main characters gender and/or sexuality, press the generate more partners button	The game should generate the correct potential partners for the main characters new gender and/or sexuality	The game generates the correct potential partners for the main characters new gender and/or sexuality. Pass	11a
12a	Death	Play through the game and wait for death to occur.	The game should randomly kill all people in the game. Family members should have an event generated and the main character should have a message box displayed to the player	The game randomly kills people in the game and displays it appropriately. Pass	FL1; FL2

NB: The saved game data from video FL2 is available in the Appendix B

5.2 User Acceptance Testing

For the final testing for my project, I will ask some of my friends & classmates who fit into the stakeholders category to test my project and give me feedback. Their findings are listed below and will be discussed in the Evaluation chapter.

5.2.1 Findings from User Acceptance Testing

Liked:

- reasons for death
- the changes in lengths of life
- the random generation of names
- the random generation of crimes
- the random generation of jobs
- the silly death reasons
- `TAB` key working to move between controls on character creation form

Didn't like:

- that the same death could occur twice in the same game
- the death frequency
- the age gap possibilities between the partner and main character
- when main character died at age 1
- even if both the parents have died, the school events reference them

Suggested Improvements:

- buttons need labels as well as icons
- make the label controls on the choice box clickable too
- neaten the 'Save & Home' icon so they don't overlap
- the main character should automatically get a job when the turn 18

5.2.2 Observations from User Acceptance Testing

These are observations I made while watching other people play my game:

- Quit band 1 jobs quite frequently
- Didn't attempt to edit the character
- Didn't say but inferred liked progress bars for score display
- Flicked between tabs lots
- Users struggled with using the app without any guidance or input from me
- Users stayed on the main page mostly, not interacting with other elements of the game

5.2.3 Evidence of my testers testing

Shown below are screenshots of direct messages with my testers, evidencing that they were involved in the user acceptance testing.

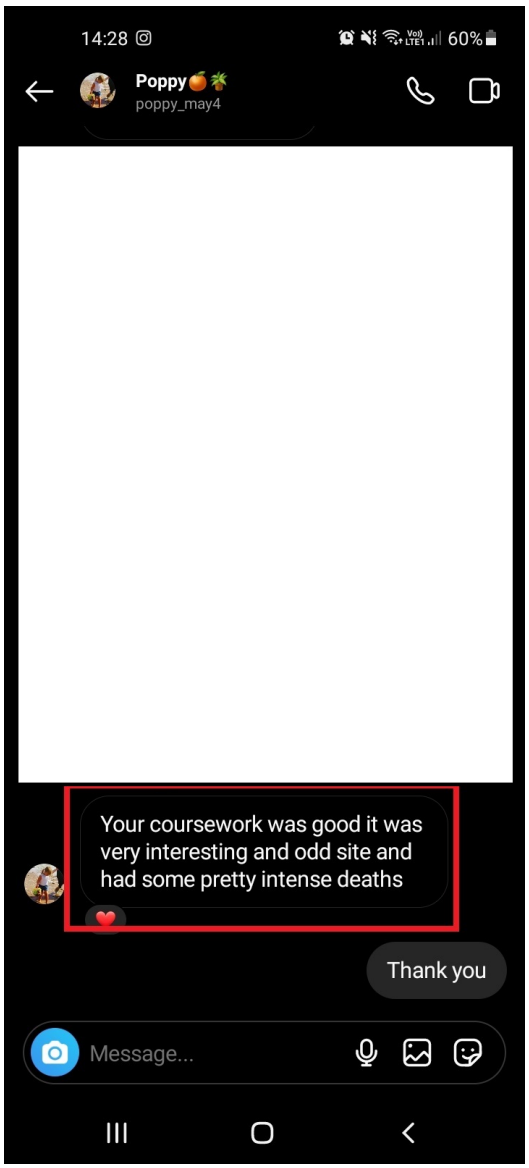


Figure 5.1: Evidence of tester 1 - Poppy

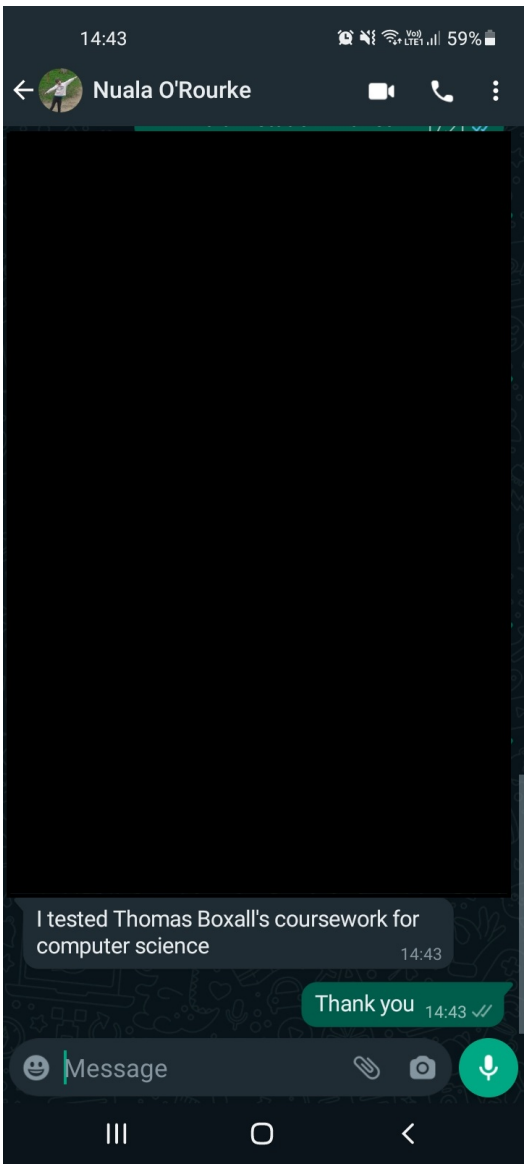


Figure 5.2: Evidence of tester 1 - Nuala

Chapter 6

Evaluation

After completing implementation, I am now able to evaluate my project using the success criteria and testing. Due to the nature of this project being a prototype, it will naturally have a smaller, more condensed scope than a fully developed project.

6.1 Reviewing the Success Criteria

6.1.1 Success Criteria

Listed below is a repeat of the success criteria from the Analysis chapter. I will review each point below it.

1. Ageing Up
    (a) The game has a button which ages up the character
    (b) As part of this process, a random event will occur (eg, family member dies, your partner wins something, you are pregnant)
    (c) Some events will happen at set times (eg. finishing school)
2. Relationships
    (a) Having a relationship with parents
    (b) Having partners
    (c) Marrying partners
    (d) Divorcing partners
    (e) Being able to pick the characters sexuality
    (f) Having kids with your partner (adopted/biological)
3. Crime
    (a) Ability to commit crime - this will be a button which will randomly generate a crime, then deduct a set amount from the lifescore.
4. Employment
    (a) Get jobs
    (b) Quit jobs
    (c) The longer you stay in a job, the more `jobPoints` a character will get each year.
5. Character Picker
    (a) At the start of each life, the user will be able to pick the character they want to play the game as. This will generate a brief backstory to the character which will crop up throughout the game (deaths of family members, key events in siblings' lives)

6. Education
   (a) There will be a random education element to the game which will allow players to progress through school (which will have randomly generated names). Every time they complete a course or school, they get `educationPoints`.
7. Saving
   (a) The game should be able to write the data out to a file(s) which can be loaded in during runtime to resume the game. These do not need to be readable by the user.
   (b) There should be a way for the user to export the life from the game at the end of the game to a non-editable file format (eg, PDF).
8. Scores
   (a) There should be a number of scores which the actions of the player can affect throughout the game.

6.1.2 Comments about the Success Criteria

1. (a) This point has been fully completed. There is a button on the main game screen which ages up the main character and progresses the game forward. This button is also the button which triggers the majority of the game events to occur.
   (b) This point has been mostly completed. So far, I have implemented the framework for the random elements, however I haven't included as many random events as I would like. Further developments to complete this point as I have the framework completed.
   (c) This point is complete. I have completed this using two functions, one which generates one off events (eg, finishing school) and one which changes game rules. This is used for things like unlocking the jobs and character customisation elements.
2. (a) This point has been completed fully. There are two additional people generated when a new life is generated, one is designated the mother and one is designated the father. Throughout the game there is a random chance of these people dying each year and a random chance of them contracting a medical condition too.
   (b) This point has been mostly completed. There is a partners area to the game where the user can select a person which they want to begin a relationship with. This person is then assigned partner status where the main character can view their information. To complete this point fully, I will need to develop the partner more, including more information about and concerning them
   (c) This is not complete due to time constraints.
   (d) As point 2c is incomplete, this is also incomplete as they are intertwined.
   (e) This point is fully complete. There is a tab on the main game screen where, once the character has turned a certain age, you can set their name, gender and sexuality.
   (f) This is not complete due to time constraints; however the code for adopted children would've been extremely similar to that of the partner generation and the code for a newborn born baby would be extremely similar to that of creating the main character for the first time.
3. (a) This is complete. There is a button which will randomly generate a crime from the list of crimes I have programmed. There is no forced nature to committing crimes which adds a less violent approach to the game. Once you have committed a crime, an event will be generated where the user can see the crime and punishment also generated.
4. (a) This point is complete. On the jobs tab on the main game screen, there is a button which will generate a jobs board where the user can select the job which they want the character to take. The jobs board generation algorithm is set so it takes into account the current job

score, duration and band. This is an advancement on the basic system I had in mind when designing this success criteria however I feel it adds quite a lot to the game.

(b) This point is complete. On the jobs tab on the main game screen, there is a button which allows you to quit the job you currently have. Doing this results in a decreased jobscore.

(c) This point is complete. There is an algorithm which is run every time the main character ages up which takes into account how long the character has been in a job and increases proportional to the time you have been in the job for.

5. (a) I have fully completed this element and am extremely satisfied with how the implementation has gone. During implementation I found myself creating lots of lives manually which took quite a lot of time. As a result, I implemented an additional random generation of character button. This has proved extremely useful and is a valuable addition to the game.

6. (a) This point is fully complete. The game forces you through the basic education system, containing infant, junior and secondary school then two years of education in a 6th form college.

7. (a) This is fully complete. The program will write data out to a number of JSON files which it can also read in. During designing, I was thinking I would have to write the code to do this myself, however I found a package which has the code pre-written.

(b) This is not complete due to time constraints.

8. (a) This is mostly complete. I have been able to successfully program all of the subscores, but not the main score. This is due to me leaving the main score to be programmed at the end of development and running out of time.

6.1.3 Further analysis of the success criteria

Reviewing the success criteria, I have completed a satisfactory number of the points. The points I haven't completed weren't completed as I thought it was best to focus on the breadth of the app rather than completing each element to a fully implemented standard. For example, I thought it was better to move on from generating family members once I had the mother and father generating, as the other members of the family would be done in a very similar way; this is very much similar to the kids generation which hasn't been implemented. It would be easy to go back in and add this code due to the modular approach I have taken to development.
The points which I have completed have generally been completed to an acceptable level of satisfaction. Each of the points which I have marked as fully complete are fully functioning elements of the game, however there are still bugs within them. These bugs are primarily from me using poor programming techniques from the start. I should've been using Try Catch blocks from the very beginning, logging errors to an error file where I could view and analyse them after each run. This would prevent a lot of the runtime errors which I am encountering.

6.2 Evaluating User Acceptance Testing results

There were a number of things which the testers liked. This was a smaller list than I was expecting there to be. I was hoping the random generation element to the game would be enjoyed and it was - with most users liking the random character generation button as well as the random names of characters throughout the game.

The testers thought that the user interface has been partially well designed, with a consistent theme across all pages and a nice use of colours. A drawback to the user interface is lack of clear text indicating what buttons do. I made the choice to use icons to represent the actions of a button rather than including a label on the button. This resulted in some confusion for the users and me having to explain how the software works to them. They also liked the fact that the `TAB` key on the keyboard would

allow them to move between the controls on the character designer screen. Accessibility wasn't a part of my success criteria originally, however it should have been as usability is an extremely important to modern application development. This is a really good example of where I have used keyboard shortcuts to enable users to use the keyboard to navigate the app by keyboard alone. The testers also liked the variety and use of controls on the application, especially the progress bar controls which were used to represent the scores, especially when paired with the ToolTips which gave an accurate read out of the scores. There were some other issues with their User Interface and User Experience, most of which are surrounding the buttons. Many users thought that the button controls should also have a label stating what their purpose is as the icons are nice but sometimes unclear. I completely agree with this comment and throughout development of the application I have noted that the icons are unclear, especially when the icons used are 'less common' than some. For example, the save icon is now a standard icon across digital applications whereas the reshuffle jobs board icon isn't. The buttons where I have merged two icons together need neatening, another user commented. The final User Interface improvement which was suggested to me, is to make the labels clickable on the choice box. This is something which I could easily implement and would be released in the next update of the game.

It was interesting watching the users use the app without any prompting from me on what to do. Initially, they mainly stayed on the main game page; this could be because the tab control menu bar is quite small. An improvement I will look into making is to increase the size of this menu bar. After I said to the testers that there were different tabs which contained different features, they explored and worked out how to use the different elements of the game. From playing with the different elements, I could see that they were watching the scores quite a lot, remarking to the other people in the room when a score changed. This pleased me as it was something which I had hoped would have the same reaction. None of the testers attempted to edit the character, however this is quite a minor feature. A few commented on the main characters sexuality being assigned to 'straight' at birth and not being able to change it. I think this does need changing to be more inclusive. I would change it so the sexuality at birth is 'undefined' then that option can be changed all the time throughout the game; I would also change the ages on the name changing to accurately reflect the real world, as you can't change your name until you are 16.

The job aspect to the game confused a number of my testers, this is probably because I designed and implemented an intricate job system with different bands and different salaries. They did enjoy watching the salary increase throughout the years of holding a band one job. After some instruction from myself, they were able to understand how the system worked and obtain jobs. As with the other scores, they enjoyed watching it increase after holding down a job. An improvement to the job system which a few testers independently commented is: make the main character automatically get a band one job when they turn 18. This is something which I had considered doing; however, at the time, I decided against it to add more value and activity to the game. A happy medium between me wanting to maintain the player lead strategy to the game and wanting to listen to feedback from my users, could be to add an alert to the player in the form of a message box each year (or just for significant years) which lets them know that they've aged up, any life events which have happened in that year and any decisions/new elements of the game which are now available to them.

An area of the game which many of my testers enjoyed was the death algorithms. All my testers enjoyed the different reasons for deaths which flashed up when a parent died, laughing at some of them. Some testers didn't like the fact that the same death reason could be used multiple times in the same run of the game. They also found that the death frequency was too high; there should be a timeout on deaths. This is something which I had noticed during development as well. The majority of the parents deaths were in the earlier stages of the main characters life. This will be less of a problem if I design and implement a greater back story to the main character. This upgrade is high on my list, especially as an area which the testers didn't like was the fact that even if both the parents have died, the word 'parents' is used in the education events.

None of the testers explored the partners aspect of the game in great depths. However, the one that did found something which will need re-balancing during the next update. There is quite a big age gap possible between the main character and the partner. The tester that found this character was 4 at the time and their boyfriend was 33. This is obviously wrong, not only with the age gap but also being able to be in a relationship at age 4. In the real world, this would be considered child abuse so will need re-designing completely before the next update. Relationships, sexuality and gender is a really difficult area to develop apps in, especially considering there isn't a one-size-fits-all approach which can be taken to it as everyone is different. This is something which I would like to put more time into designing and implementing as I don't feel like I have done it justice.

6.3 Evaluating the Final Testing

From the final testing, I can see that generally the app is bug free, however there are a few areas which are still buggy.

The main menu tests (test 1a through 1g inclusive) are mostly graded as pass. This was expected as it was the first form which I designed therefore I had the most time to put into it. The only test which was graded as a fail was test 1d. This was expected as I hadn't got around to designing the settings form yet. This form would have included some accessibility options and would have been accessible from all forms in the game. The implementation of this form would have been relatively complex as I would need to define the colours of each control programmatically rather than using the properties menu in Visual Studio. To do this, each form would have a procedure which would run when it is opened which would set the colours of all the controls. This procedure could also run each time there is a change to the accessibility options made. Tests 1f and 1g were graded as a pass, however I feel they weren't a full pass. I would have liked it more if the buttons actually resized to reflect the size of the form better.

Test set 2 focused on the create new game form `frmNewGame` and the different functions which it can perform. Tests 2a and 2b passed, as expected. Test 2c failed; this was disappointingly expected, as at the beginning of development I didn't think about input validation. This is something which I will strive to include in the next update of the game. Tests 2d and 2e were expected to pass as the random character generation algorithm is relatively robust. The outcome of test 2f didn't surprise me, but did disappoint me. I had hoped that I had set the TabIndex of each of the controls correctly. Evidently I hadn't but this is an easy fix which I can complete for the next update of the app. Test 2g didn't surprise me either as I was planning on adding more to the create new game form so had left the resizing to be completed after then. As it turned out, I didn't make those changes. Test 2i pleased me to see that it had passed as it was a trial of input validation. A further improvement to the current input validation system would be to include some form of error reporting system where the error message was logged in a supporting file to the game and I could view them to improve the game.

Test set 3 focused on the loading game form. Test 3a wasn't a surprise to see it pass, tests 2b and 3c were disappointing to see fail however. To solve these issues, I would need to include a try-catch block for each file that needs opening and if the file isn't present display a suitable message to the user, perhaps listing the files which were missing. Test 3d and 3e performed as expected, to rectify test 3d shouldn't be too complex as I would just need to change some of the properties in the Visual Studio Properties panel.

Test set 4 focused on the main game screen form and on the basic functions of the game. Most of the the tests were graded as a pass, this is unsurprising as they are all features of the game which I have been working with throughout the majority of development. The only test which was graded as a fail was test 4d. This was the form resizing test so was unsurprising.

Test set 5 focused on the family tab. During the analysis stage, I had more ideas for this section

but due to the nature of this project being a prototype, I didn't design or implement any of them, hence why the form is a bit sparse. The two tests 5a and 5b were both graded a pass.

Test set 6 focused on the education system of the game. This also included the choice box as I haven't used it anywhere else. All the tests were graded as a pass. During planning, I had intended to use the choice box more throughout the game, however it ended up only being used in the education section.

Test set 7 focused on the very simple crime section of the game. The commit random crime button (test 7a) works as intended and the crime only event box (test 7b) also works as expected.

Test set 8 focused on the character customisation options which were presented to the user. It is unsurprising to me that these all passed as these options were developed further down the development line when I was focusing more on the robustness of my app. The end life button tests were also a pass, however I think this system could be better. Currently the game just triggers the standard kill main character function, which generates a random reason for death in the same way it would if the main character was to spontaneously die. I would like to improve this algorithm so it can take a reason for death as a parameter rather than generate it within, this wouldn't change anything major as far as the algorithm goes, however it would mean that I could use self-triggered reasons for death if the user presses the end life button.

Test set 9 focused on the employment features of the game. This mostly worked, apart from test 9e. This test looked at applying for a job while the main character currently has a job. To fix this, I would need to include an if statement before the apply for selected job function is run which makes sure the main character in employment is set to false. The other tests passed.

Test set 10 focused on the relationships section of the game. This section was not where I wanted it to be, however due to the nature of this project being a prototype, it would never have been. From reviewing the tests, it is evident that the features which I had implemented were implemented well.

Test set 11 followed on from test set 10. It focused on changes to the relationship system once a change had been made to the main characters gender and/or sexuality. The test passed as I had been thinking about these customisation features all throughout the design and implementation of the game.

The final test set, 12, was a combination of two areas. It focused on the full life and the natural end to a full life (death). This test passed which was expected as I had tested it extensively throughout the development process.

6.4 Further Evaluation

C# was a new language to me for this project, so I feel I have done quite well, using my knowledge of other languages to inform logical decisions I have made throughout this project. There have been instances where I haven't been able to do something on the first try, for example the saving to a file with arrays of objects. However I used my programming knowledge to research and logically solve the problem; hence I found the package Newtonsoft. I had some experience using Visual Studio and developing Windows Forms from using Visual Basic in other projects, this was extremely valuable knowledge to have both during the design (as I knew about what controls I could use and what attributes they had) and implementation (as I knew my way around the software and how to change settings) phases.

This project has also taught me a lot about the software development life cycle, especially how no matter how good the design of the system is, when you come to implement it, things will implement differently. This might have just been my way of working during this project, not worrying about

sticking to my design completely; however it is useful knowledge to have gained from this project. I am not concerned by the fact that I haven't stuck to my design as I think it is an important part of the creative process to adapt and overcome challenges that are presented to you during the project. An example of this is when I was setting up the progress bars for the scores displays, I wanted them to be the theme colours of the app (blue, purple and pink). However, WinForms has a default colour theme which is loaded in the Program.cs file. I hadn't intended to remove the line which loads the default styles, but once I did, my app looked much more how I wanted it to.

Something else which this project has taught me, is that when designing algorithms, you need to have a much better sense of the bigger picture. This is especially true for the algorithms which I designed in the design phase, including the 'age up'algorithm. When writing these, I hadn't thought about how each element fitted into the final solution and the order which they would need to be implemented in. This meant that during implementation, I needed to deviate from my design to enable me to implement it. Now I have completed the implementation, I could go back through and re-order the code however this would be a waste of time.

I am disappointed that I didn't manage to implement the LifeScore function as this was one of the biggest selling points of my game. I had left this until towards the end of development so once I had written it so I wouldn't have to change it to take into account any other values or scores. It wasn't implemented as I had run out of time. Another area which I spent too much time on was the user interface. I should've been more realistic with my ambitions and used the default lighter theme which Visual Studio defaults to; this would have allowed me to spend the time I spent customising the UI on developing the substance of the application.

6.5 Maintenance and Further Development of the Application

The approach I have taken to designing and implementing the solution is a modular approach, allowing me to easily debug and test different elements of the game without having to run all the code every time. This also is beneficial as it allows for future development of the game where subroutines I have written could be easily replaced or modified without having to dig through many lines of code. This will also allow for easy maintenance to the system. A further improvement I could make to this is to use methods belonging to a class more rather than have all my code in one file. This may make my code slightly shorter too.

The code is well-commented and documented in the Implementation chapter of this report, this will enable the application to be maintained easily. The quality of the comments could be improved however as well as removing some of the old lines of code which have been commented out during debugging.

A further improvement I could make to the ease of maintenance of my solution would be to use GitHub, a cloud based version control system which allows the development of the program to be branched. This would have been extremely valuable to the development of the program as I would have been able to create a branch of each of the new features which I am implementing then I would be able to develop them safely without worrying that I am going to break the master working version of the program. Once satisfied with each branch, they can be merged with the master branch.

6.6 Improvements to the current solution

There are a number of improvements I would like to make to my solution should I choose to further develop it after this project.
One of these improvements is to add more of a backstory to the partner, adding maybe a brief bio; information about their jobs, hobbies and preferred pets. This would have added a more personal touch to the game, something I am now realising is a big factor in the games I researched way back

in my analysis.
Another improvement I would like to make is to look into the file storage I am using. Whilst the current solution of using a number of separate JSON files works, it feels inelegant. A potential solution would be to use a database. This would be a lot of extra work with little gain as each system the game is played on would need the same database system to be loaded onto it. This would provide storage space efficiency improvements over the current solution. As seen in appendix B, the `eventArray` and `familyArray` both have many empty slots, this is quite a large waste of space. Another potential solution would be to abstract the saving from the users all together, creating a save data folder in the directory which the game is installed in and the using an inbuilt game function to allow the user to select the game to open. This would be more feasible to implement.

6.7 Personal takeaways from this project

Through completing this project, I have learnt a lot about the software development life cycle, especially how important it is to thoroughly think through each step before you begin documenting it. Furthermore, I have learnt how valuable being comfortable with the language is before you begin to program in it. This has helped me to understand the reasoning behind software developers specialising in a language, as to be able to write a program quickly, you need to know what you are doing in the language.

6.8 Known Issues

There are a number of known issues with the current iteration of the solution. They are listed below

- Age of the partner is incorrect
- Spelling and grammar needs reviewing throughout the application
- The game has stopped generating jobs above band 2

These issues are simple to fix and would be resolved in the next release.

Appendix A

Full Code Listing

A.1 Forms

A.1.1 Main Menu

[FILE]Form1.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace A_level_Computer_Science_Project
{

    public partial class frmMainMenu : Form
    {
        public frmMainMenu()
        {
            InitializeComponent();
        }

        private void btnNewGame_Click(object sender, EventArgs e)
        {
            frmNewGame fNewGame = new frmNewGame();
            fNewGame.Show();
            this.Hide();
        }

        private void btnShowSaved_Click(object sender, EventArgs e)
        {
            frmShowSavedGames fShowSavedGames = new frmShowSavedGames();
            fShowSavedGames.Show();
            this.Hide();
        }

        private void frmMainMenu_Load(object sender, EventArgs e)
        {

        }

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

```
        {


        }

        private void btnAbout_Click(object sender, EventArgs e)
        {
            frmAbout fAbout = new frmAbout();
            fAbout.Show();
        }
    }
}
```

Listing A.1: FILE/Form1.cs

[FILE]Form1.Designer.cs

```

namespace A_level_Computer_Science_Project
{
    partial class frmMainMenu
    {
        /// <summary>
        ///  Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        ///  Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be
    ↪ disposed; otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        ///  Required method for Designer support - do not modify
        ///  the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            System.ComponentModel.ComponentResourceManager resources = new
    ↪ System.ComponentModel.ComponentResourceManager(typeof(frmMainMenu));
            this.panel1 = new System.Windows.Forms.Panel();
            this.pictureBox1 = new System.Windows.Forms.PictureBox();
            this.panel2 = new System.Windows.Forms.Panel();
            this.btnAbout = new System.Windows.Forms.Button();
            this.btnSettings = new System.Windows.Forms.Button();
            this.btnShowSaved = new System.Windows.Forms.Button();
            this.btnNewGame = new System.Windows.Forms.Button();
            this.panel1.SuspendLayout();
            ((System.ComponentModel.ISupportInitialize)(this.pictureBox1)).
    ↪ BeginInit();
            this.panel2.SuspendLayout();
```

```
43              this.SuspendLayout();
44              //
45              // panel1
46              //
47              this.panel1.Anchor = ((System.Windows.Forms.AnchorStyles)((((
    ↪ System.Windows.Forms.AnchorStyles.Top | System.Windows.Forms.
    ↪ AnchorStyles.Bottom)
48                    | System.Windows.Forms.AnchorStyles.Left)
49                    | System.Windows.Forms.AnchorStyles.Right)));
50              this.panel1.Controls.Add(this.pictureBox1);
51              this.panel1.Location = new System.Drawing.Point(12, 12);
52              this.panel1.Name = "panel1";
53              this.panel1.Size = new System.Drawing.Size(780, 139);
54              this.panel1.TabIndex = 0;
55              //
56              // pictureBox1
57              //
58              this.pictureBox1.Anchor = ((System.Windows.Forms.AnchorStyles)
    ↪ (((System.Windows.Forms.AnchorStyles.Top | System.Windows.Forms.
    ↪ AnchorStyles.Left)
59                    | System.Windows.Forms.AnchorStyles.Right)));
60              this.pictureBox1.BackgroundImage = global::
    ↪ A_level_Computer_Science_Project.Properties.Resources.Logo;
61              this.pictureBox1.BackgroundImageLayout = System.Windows.Forms.
    ↪ ImageLayout.None;
62              this.pictureBox1.Image = global::
    ↪ A_level_Computer_Science_Project.Properties.Resources.Logo_cropped;
63              this.pictureBox1.Location = new System.Drawing.Point(0, 0);
64              this.pictureBox1.Name = "pictureBox1";
65              this.pictureBox1.Size = new System.Drawing.Size(780, 139);
66              this.pictureBox1.SizeMode = System.Windows.Forms.
    ↪ PictureBoxSizeMode.Zoom;
67              this.pictureBox1.TabIndex = 0;
68              this.pictureBox1.TabStop = false;
69              //
70              // panel2
71              //
72              this.panel2.Anchor = ((System.Windows.Forms.AnchorStyles)((((
    ↪ System.Windows.Forms.AnchorStyles.Top | System.Windows.Forms.
    ↪ AnchorStyles.Bottom)
73                    | System.Windows.Forms.AnchorStyles.Left)
74                    | System.Windows.Forms.AnchorStyles.Right)));
75              this.panel2.BackColor = System.Drawing.Color.FromArgb(((int)(((
    ↪ byte)(31)))), ((int)(((byte)(31)))), ((int)(((byte)(31)))));
76              this.panel2.Controls.Add(this.btnAbout);
77              this.panel2.Controls.Add(this.btnSettings);
78              this.panel2.Controls.Add(this.btnShowSaved);
79              this.panel2.Controls.Add(this.btnNewGame);
80              this.panel2.Location = new System.Drawing.Point(269, 156);
81              this.panel2.Name = "panel2";
82              this.panel2.Size = new System.Drawing.Size(277, 283);
83              this.panel2.TabIndex = 1;
84              //
85              // btnAbout
86              //
87              this.btnAbout.Anchor = System.Windows.Forms.AnchorStyles.None;
88              this.btnAbout.BackColor = System.Drawing.Color.FromArgb(((int)
    ↪ (((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
89              this.btnAbout.FlatStyle = System.Windows.Forms.FlatStyle.Popup;
90              this.btnAbout.ForeColor = System.Drawing.SystemColors.Control;
91              this.btnAbout.Location = new System.Drawing.Point(3, 135);
```

```
92              this.btnAbout.Name = "btnAbout";
93              this.btnAbout.Size = new System.Drawing.Size(270, 38);
94              this.btnAbout.TabIndex = 3;
95              this.btnAbout.Text = "About";
96              this.btnAbout.UseVisualStyleBackColor = false;
97              this.btnAbout.Click += new System.EventHandler(this.
    ↪ btnAbout_Click);
98              //
99              // btnSettings
100             //
101             this.btnSettings.Anchor = System.Windows.Forms.AnchorStyles.None
    ↪ ;
102             this.btnSettings.BackColor = System.Drawing.Color.FromArgb(((int
    ↪ )(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
103             this.btnSettings.FlatStyle = System.Windows.Forms.FlatStyle.
    ↪ Popup;
104             this.btnSettings.ForeColor = System.Drawing.SystemColors.Control
    ↪ ;
105             this.btnSettings.Location = new System.Drawing.Point(3, 91);
106             this.btnSettings.Name = "btnSettings";
107             this.btnSettings.Size = new System.Drawing.Size(270, 38);
108             this.btnSettings.TabIndex = 2;
109             this.btnSettings.Text = "Settings";
110             this.btnSettings.UseVisualStyleBackColor = false;
111             this.btnSettings.Click += new System.EventHandler(this.
    ↪ btnSettings_Click);
112             //
113             // btnShowSaved
114             //
115             this.btnShowSaved.Anchor = System.Windows.Forms.AnchorStyles.
    ↪ None;
116             this.btnShowSaved.BackColor = System.Drawing.Color.FromArgb(((
    ↪ int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
117             this.btnShowSaved.FlatStyle = System.Windows.Forms.FlatStyle.
    ↪ Popup;
118             this.btnShowSaved.ForeColor = System.Drawing.SystemColors.
    ↪ Control;
119             this.btnShowSaved.Location = new System.Drawing.Point(3, 47);
120             this.btnShowSaved.Name = "btnShowSaved";
121             this.btnShowSaved.Size = new System.Drawing.Size(270, 38);
122             this.btnShowSaved.TabIndex = 1;
123             this.btnShowSaved.Text = "Show saved games";
124             this.btnShowSaved.UseVisualStyleBackColor = false;
125             this.btnShowSaved.Click += new System.EventHandler(this.
    ↪ btnShowSaved_Click);
126             //
127             // btnNewGame
128             //
129             this.btnNewGame.Anchor = System.Windows.Forms.AnchorStyles.None;
130             this.btnNewGame.BackColor = System.Drawing.Color.FromArgb(((int)
    ↪ (((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
131             this.btnNewGame.FlatStyle = System.Windows.Forms.FlatStyle.Popup
    ↪ ;
132             this.btnNewGame.ForeColor = System.Drawing.SystemColors.Control;
133             this.btnNewGame.Location = new System.Drawing.Point(3, 3);
134             this.btnNewGame.Name = "btnNewGame";
135             this.btnNewGame.Size = new System.Drawing.Size(270, 38);
136             this.btnNewGame.TabIndex = 0;
137             this.btnNewGame.Text = "New Game";
138             this.btnNewGame.UseVisualStyleBackColor = false;
139             this.btnNewGame.Click += new System.EventHandler(this.
```

```
    ↪ btnNewGame_Click);
140             //
141             // frmMainMenu
142             //
143             this.AutoScaleDimensions = new System.Drawing.SizeF(7F, 15F);
144             this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
145             this.BackColor = System.Drawing.Color.FromArgb(((int)(((byte)
    ↪ (31)))), ((int)(((byte)(31)))), ((int)(((byte)(31)))));
146             this.ClientSize = new System.Drawing.Size(804, 451);
147             this.Controls.Add(this.panel2);
148             this.Controls.Add(this.panel1);
149             this.Icon = ((System.Drawing.Icon)(resources.GetObject("$this.
    ↪ Icon")));
150             this.MinimumSize = new System.Drawing.Size(820, 490);
151             this.Name = "frmMainMenu";
152             this.Text = "Virtual-Life Main Menu";
153             this.Load += new System.EventHandler(this.frmMainMenu_Load);
154             this.panel1.ResumeLayout(false);
155             ((System.ComponentModel.ISupportInitialize)(this.pictureBox1)).
    ↪ EndInit();
156             this.panel2.ResumeLayout(false);
157             this.ResumeLayout(false);
158
159         }
160
161         #endregion
162
163         private System.Windows.Forms.Panel panel1;
164         private System.Windows.Forms.PictureBox pictureBox1;
165         private System.Windows.Forms.Panel panel2;
166         private System.Windows.Forms.Button btnAbout;
167         private System.Windows.Forms.Button btnSettings;
168         private System.Windows.Forms.Button btnShowSaved;
169         private System.Windows.Forms.Button btnNewGame;
170     }
171 }
```

Listing A.2: FILE/Form1.Designer.cs

A.1.2 About

[FILE]frmAbout.cs

```
1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Text;
7  using System.Windows.Forms;
8
9  namespace A_level_Computer_Science_Project
10 {
11     public partial class frmAbout : Form
12     {
13         public frmAbout()
14         {
15             InitializeComponent();
16         }
17
18     }
```

```
}
```

Listing A.3: FILE/frmAbout.cs

[FILE]frmAbout.Designer.cs

```

namespace A_level_Computer_Science_Project
{
    partial class frmAbout
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            System.ComponentModel.ComponentResourceManager resources = new System.ComponentModel.ComponentResourceManager(typeof(frmAbout));
            this.pictureBox1 = new System.Windows.Forms.PictureBox();
            this.panel2 = new System.Windows.Forms.Panel();
            this.label1 = new System.Windows.Forms.Label();
            this.label2 = new System.Windows.Forms.Label();
            ((System.ComponentModel.ISupportInitialize)(this.pictureBox1)).BeginInit();
            this.panel2.SuspendLayout();
            this.SuspendLayout();
            //
            // pictureBox1
            //
            this.pictureBox1.Anchor = ((System.Windows.Forms.AnchorStyles)(((System.Windows.Forms.AnchorStyles.Top | System.Windows.Forms.AnchorStyles.Left)
            | System.Windows.Forms.AnchorStyles.Right)));
            this.pictureBox1.BackgroundImage = global::A_level_Computer_Science_Project.Properties.Resources.Logo;
            this.pictureBox1.BackgroundImageLayout = System.Windows.Forms.ImageLayout.None;
            this.pictureBox1.Image = global::A_level_Computer_Science_Project.Properties.Resources.Logo_cropped;
            this.pictureBox1.Location = new System.Drawing.Point(10, 6);
```

```
49            this.pictureBox1.Name = "pictureBox1";
50            this.pictureBox1.Size = new System.Drawing.Size(780, 139);
51            this.pictureBox1.SizeMode = System.Windows.Forms.
    ↪ PictureBoxSizeMode.Zoom;
52            this.pictureBox1.TabIndex = 2;
53            this.pictureBox1.TabStop = false;
54            //
55            // panel2
56            //
57            this.panel2.Anchor = ((System.Windows.Forms.AnchorStyles)((((
    ↪ System.Windows.Forms.AnchorStyles.Top | System.Windows.Forms.
    ↪ AnchorStyles.Bottom)
58            | System.Windows.Forms.AnchorStyles.Left)
59            | System.Windows.Forms.AnchorStyles.Right)));
60            this.panel2.BackColor = System.Drawing.Color.FromArgb(((int)(((
    ↪ byte)(31)))), ((int)(((byte)(31)))), ((int)(((byte)(31)))));
61            this.panel2.Controls.Add(this.label2);
62            this.panel2.Controls.Add(this.label1);
63            this.panel2.Location = new System.Drawing.Point(12, 162);
64            this.panel2.Name = "panel2";
65            this.panel2.Size = new System.Drawing.Size(776, 283);
66            this.panel2.TabIndex = 3;
67            //
68            // label1
69            //
70            this.label1.AutoSize = true;
71            this.label1.Font = new System.Drawing.Font("Segoe UI", 12F,
    ↪ System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point);
72            this.label1.ForeColor = System.Drawing.Color.FromArgb(((int)(((
    ↪ byte)(100)))), ((int)(((byte)(164)))), ((int)(((byte)(209)))));
73            this.label1.Location = new System.Drawing.Point(22, 16);
74            this.label1.Name = "label1";
75            this.label1.Size = new System.Drawing.Size(65, 21);
76            this.label1.TabIndex = 4;
77            this.label1.Text = "Images";
78            //
79            // label2
80            //
81            this.label2.AutoSize = true;
82            this.label2.Location = new System.Drawing.Point(22, 37);
83            this.label2.Name = "label2";
84            this.label2.Size = new System.Drawing.Size(161, 15);
85            this.label2.TabIndex = 5;
86            this.label2.Text = "Downloaded from Flaticon.io";
87            //
88            // frmAbout
89            //
90            this.AutoScaleDimensions = new System.Drawing.SizeF(7F, 15F);
91            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
92            this.BackColor = System.Drawing.Color.FromArgb(((int)(((byte)
    ↪ (31)))), ((int)(((byte)(31)))), ((int)(((byte)(31)))));
93            this.ClientSize = new System.Drawing.Size(800, 450);
94            this.Controls.Add(this.pictureBox1);
95            this.Controls.Add(this.panel2);
96            this.ForeColor = System.Drawing.Color.White;
97            this.Icon = ((System.Drawing.Icon)(resources.GetObject("$this.
    ↪ Icon")));
98            this.MaximizeBox = false;
99            this.Name = "frmAbout";
100           this.Text = "About";
101           ((System.ComponentModel.ISupportInitialize)(this.pictureBox1)).
```

```
    ↪ EndInit();
102             this.panel2.ResumeLayout(false);
103             this.panel2.PerformLayout();
104             this.ResumeLayout(false);
105 
106         }
107 
108         #endregion
109 
110         private System.Windows.Forms.PictureBox pictureBox1;
111         private System.Windows.Forms.Panel panel2;
112         private System.Windows.Forms.Label label2;
113         private System.Windows.Forms.Label label1;
114     }
115 }
```

Listing A.4: FILE/frmAbout.Designer.cs

A.1.3 Choice Box

[FILE]frmChoiceBox.cs

```
1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Text;
7  using System.Windows.Forms;
8  
9  namespace A_level_Computer_Science_Project
10 {
11     public partial class frmChoiceBox : Form
12     {
13         public int ReturnValue1 { get; set; }
14         string headerText;
15         string subheaderText;
16         string opt1header;
17         string opt1body;
18         string opt2header;
19         string opt2body;
20         string opt3header;
21         string opt3body;
22         public frmChoiceBox(string header, string subheader, string opt1h,
    ↪ string opt1b, string opt2h, string opt2b, string opt3h, string opt3b)
23         {
24             InitializeComponent();
25             headerText = header;
26             subheaderText = subheader;
27             opt1header = opt1h;
28             opt1body = opt1b;
29             opt2header = opt2h;
30             opt2body = opt2b;
31             opt3header = opt3h;
32             opt3body = opt3b;
33         }
34 
35         private void frmChoiceBox_Load(object sender, EventArgs e)
36         {
37             lblHeader.Text = headerText;
38             lblSubheading.Text = subheaderText;
39             lblChoice1.Text = opt1header;
```

```
            lblChoice1Desc.Text = opt1body;
            lblChoice2.Text = opt2header;
            lblChoice2Desc.Text = opt2body;
            lblChoice3.Text = opt3header;
            lblChoice3Desc.Text = opt3body;

        }

        private void btnChoice1_Click(object sender, EventArgs e)
        {

            this.ReturnValue1 = 1;
            this.DialogResult = DialogResult.OK;
            this.Close();

        }

        private void btnChoice2_Click(object sender, EventArgs e)
        {
            this.ReturnValue1 = 2;
            this.DialogResult = DialogResult.OK;
            this.Close();
        }

        private void btnChoice3_Click(object sender, EventArgs e)
        {
            this.ReturnValue1 = 3;
            this.DialogResult = DialogResult.OK;
            this.Close();
        }

        private void btnRandomChoice_Click(object sender, EventArgs e)
        {
            Random rnd = new Random();
            int random = rnd.Next(1, 4);
            this.ReturnValue1 = random;
            this.DialogResult = DialogResult.OK;
            this.Close();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            this.DialogResult = DialogResult.Cancel;
            this.Close();
        }
    }
}
```

Listing A.5: FILE/frmChoiceBox.cs

[FILE]frmChoiceBox.Designer.cs

```

namespace A_level_Computer_Science_Project
{
    partial class frmChoiceBox
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

```

```
        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            System.ComponentModel.ComponentResourceManager resources = new System.ComponentModel.ComponentResourceManager(typeof(frmChoiceBox));
            this.lblHeader = new System.Windows.Forms.Label();
            this.lblSubheading = new System.Windows.Forms.Label();
            this.panel1 = new System.Windows.Forms.Panel();
            this.btnRandomChoice = new System.Windows.Forms.Button();
            this.btnChoice3 = new System.Windows.Forms.Button();
            this.btnChoice2 = new System.Windows.Forms.Button();
            this.btnChoice1 = new System.Windows.Forms.Button();
            this.panel2 = new System.Windows.Forms.Panel();
            this.lblChoice3 = new System.Windows.Forms.Label();
            this.lblChoice2Desc = new System.Windows.Forms.Label();
            this.lblChoice3Desc = new System.Windows.Forms.Label();
            this.lblChoice2 = new System.Windows.Forms.Label();
            this.lblChoice1Desc = new System.Windows.Forms.Label();
            this.lblChoice1 = new System.Windows.Forms.Label();
            this.panel3 = new System.Windows.Forms.Panel();
            this.panel4 = new System.Windows.Forms.Panel();
            this.button1 = new System.Windows.Forms.Button();
            this.panel1.SuspendLayout();
            this.panel2.SuspendLayout();
            this.panel3.SuspendLayout();
            this.panel4.SuspendLayout();
            this.SuspendLayout();
            //
            // lblHeader
            //
            this.lblHeader.AutoSize = true;
            this.lblHeader.Font = new System.Drawing.Font("Segoe UI", 21.75F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point);
            this.lblHeader.ForeColor = System.Drawing.Color.White;
            this.lblHeader.Location = new System.Drawing.Point(0, 0);
            this.lblHeader.Name = "lblHeader";
            this.lblHeader.Size = new System.Drawing.Size(100, 40);
            this.lblHeader.TabIndex = 0;
            this.lblHeader.Text = "label1";
            //
            // lblSubheading
            //
```

```
69              this.lblSubheading.AutoSize = true;
70              this.lblSubheading.ForeColor = System.Drawing.Color.White;
71              this.lblSubheading.Location = new System.Drawing.Point(3, 51);
72              this.lblSubheading.Name = "lblSubheading";
73              this.lblSubheading.Size = new System.Drawing.Size(38, 15);
74              this.lblSubheading.TabIndex = 1;
75              this.lblSubheading.Text = "label2";
76              //
77              // panel1
78              //
79              this.panel1.Controls.Add(this.btnRandomChoice);
80              this.panel1.Controls.Add(this.btnChoice3);
81              this.panel1.Controls.Add(this.btnChoice2);
82              this.panel1.Controls.Add(this.btnChoice1);
83              this.panel1.Location = new System.Drawing.Point(12, 380);
84              this.panel1.Name = "panel1";
85              this.panel1.Size = new System.Drawing.Size(776, 58);
86              this.panel1.TabIndex = 2;
87              //
88              // btnRandomChoice
89              //
90              this.btnRandomChoice.BackColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))
    ↪ ));
91              this.btnRandomChoice.FlatStyle = System.Windows.Forms.FlatStyle.
    ↪ Popup;
92              this.btnRandomChoice.ForeColor = System.Drawing.Color.White;
93              this.btnRandomChoice.Location = new System.Drawing.Point(464, 0)
    ↪ ;
94              this.btnRandomChoice.Name = "btnRandomChoice";
95              this.btnRandomChoice.Size = new System.Drawing.Size(75, 58);
96              this.btnRandomChoice.TabIndex = 3;
97              this.btnRandomChoice.Text = "Random Choice";
98              this.btnRandomChoice.UseVisualStyleBackColor = false;
99              this.btnRandomChoice.Click += new System.EventHandler(this.
    ↪ btnRandomChoice_Click);
100             //
101             // btnChoice3
102             //
103             this.btnChoice3.BackColor = System.Drawing.Color.FromArgb(((int)
    ↪ (((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
104             this.btnChoice3.FlatStyle = System.Windows.Forms.FlatStyle.Popup
    ↪ ;
105             this.btnChoice3.ForeColor = System.Drawing.Color.White;
106             this.btnChoice3.Location = new System.Drawing.Point(383, 0);
107             this.btnChoice3.Name = "btnChoice3";
108             this.btnChoice3.Size = new System.Drawing.Size(75, 58);
109             this.btnChoice3.TabIndex = 2;
110             this.btnChoice3.Text = "Choice 3";
111             this.btnChoice3.UseVisualStyleBackColor = false;
112             this.btnChoice3.Click += new System.EventHandler(this.
    ↪ btnChoice3_Click);
113             //
114             // btnChoice2
115             //
116             this.btnChoice2.BackColor = System.Drawing.Color.FromArgb(((int)
    ↪ (((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
117             this.btnChoice2.FlatStyle = System.Windows.Forms.FlatStyle.Popup
    ↪ ;
118             this.btnChoice2.ForeColor = System.Drawing.Color.White;
119             this.btnChoice2.Location = new System.Drawing.Point(302, 0);
```

```
120             this.btnChoice2.Name = "btnChoice2";
121             this.btnChoice2.Size = new System.Drawing.Size(75, 58);
122             this.btnChoice2.TabIndex = 1;
123             this.btnChoice2.Text = "Choice 2";
124             this.btnChoice2.UseVisualStyleBackColor = false;
125             this.btnChoice2.Click += new System.EventHandler(this.
    ↪ btnChoice2_Click);
126             //
127             // btnChoice1
128             //
129             this.btnChoice1.BackColor = System.Drawing.Color.FromArgb(((int)
    ↪ (((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
130             this.btnChoice1.FlatStyle = System.Windows.Forms.FlatStyle.Popup
    ↪ ;
131             this.btnChoice1.ForeColor = System.Drawing.Color.White;
132             this.btnChoice1.Location = new System.Drawing.Point(221, 0);
133             this.btnChoice1.Name = "btnChoice1";
134             this.btnChoice1.Size = new System.Drawing.Size(75, 58);
135             this.btnChoice1.TabIndex = 0;
136             this.btnChoice1.Text = "Choice 1";
137             this.btnChoice1.UseVisualStyleBackColor = false;
138             this.btnChoice1.Click += new System.EventHandler(this.
    ↪ btnChoice1_Click);
139             //
140             // panel2
141             //
142             this.panel2.Controls.Add(this.lblChoice3);
143             this.panel2.Controls.Add(this.lblChoice2Desc);
144             this.panel2.Controls.Add(this.lblChoice3Desc);
145             this.panel2.Controls.Add(this.lblChoice2);
146             this.panel2.Controls.Add(this.lblChoice1Desc);
147             this.panel2.Controls.Add(this.lblChoice1);
148             this.panel2.Location = new System.Drawing.Point(12, 183);
149             this.panel2.Name = "panel2";
150             this.panel2.Size = new System.Drawing.Size(776, 191);
151             this.panel2.TabIndex = 3;
152             //
153             // lblChoice3
154             //
155             this.lblChoice3.AutoSize = true;
156             this.lblChoice3.Font = new System.Drawing.Font("Segoe UI", 11.25
    ↪ F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point);
157             this.lblChoice3.ForeColor = System.Drawing.Color.FromArgb(((int)
    ↪ (((byte)(100)))), ((int)(((byte)(164)))), ((int)(((byte)(209)))));
158             this.lblChoice3.Location = new System.Drawing.Point(37, 126);
159             this.lblChoice3.Name = "lblChoice3";
160             this.lblChoice3.Size = new System.Drawing.Size(51, 20);
161             this.lblChoice3.TabIndex = 7;
162             this.lblChoice3.Text = "label1";
163             //
164             // lblChoice2Desc
165             //
166             this.lblChoice2Desc.AutoSize = true;
167             this.lblChoice2Desc.ForeColor = System.Drawing.Color.White;
168             this.lblChoice2Desc.Location = new System.Drawing.Point(37, 90);
169             this.lblChoice2Desc.Name = "lblChoice2Desc";
170             this.lblChoice2Desc.Size = new System.Drawing.Size(38, 15);
171             this.lblChoice2Desc.TabIndex = 6;
172             this.lblChoice2Desc.Text = "label6";
173             //
174             // lblChoice3Desc
```

```
175             //
176             this.lblChoice3Desc.AutoSize = true;
177             this.lblChoice3Desc.ForeColor = System.Drawing.Color.White;
178             this.lblChoice3Desc.Location = new System.Drawing.Point(37, 146)
    ↪ ;
179             this.lblChoice3Desc.Name = "lblChoice3Desc";
180             this.lblChoice3Desc.Size = new System.Drawing.Size(38, 15);
181             this.lblChoice3Desc.TabIndex = 5;
182             this.lblChoice3Desc.Text = "label5";
183             //
184             // lblChoice2
185             //
186             this.lblChoice2.AutoSize = true;
187             this.lblChoice2.Font = new System.Drawing.Font("Segoe UI", 11.25
    ↪ F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point);
188             this.lblChoice2.ForeColor = System.Drawing.Color.FromArgb(((int)
    ↪ (((byte)(100)))), ((int)(((byte)(164)))), ((int)(((byte)(209)))));
189             this.lblChoice2.Location = new System.Drawing.Point(37, 70);
190             this.lblChoice2.Name = "lblChoice2";
191             this.lblChoice2.Size = new System.Drawing.Size(51, 20);
192             this.lblChoice2.TabIndex = 4;
193             this.lblChoice2.Text = "label4";
194             //
195             // lblChoice1Desc
196             //
197             this.lblChoice1Desc.AutoSize = true;
198             this.lblChoice1Desc.ForeColor = System.Drawing.Color.White;
199             this.lblChoice1Desc.Location = new System.Drawing.Point(37, 41);
200             this.lblChoice1Desc.Name = "lblChoice1Desc";
201             this.lblChoice1Desc.Size = new System.Drawing.Size(38, 15);
202             this.lblChoice1Desc.TabIndex = 3;
203             this.lblChoice1Desc.Text = "label3";
204             //
205             // lblChoice1
206             //
207             this.lblChoice1.AutoSize = true;
208             this.lblChoice1.Font = new System.Drawing.Font("Segoe UI", 11.25
    ↪ F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point);
209             this.lblChoice1.ForeColor = System.Drawing.Color.FromArgb(((int)
    ↪ (((byte)(100)))), ((int)(((byte)(164)))), ((int)(((byte)(209)))));
210             this.lblChoice1.Location = new System.Drawing.Point(37, 21);
211             this.lblChoice1.Name = "lblChoice1";
212             this.lblChoice1.Size = new System.Drawing.Size(51, 20);
213             this.lblChoice1.TabIndex = 2;
214             this.lblChoice1.Text = "label1";
215             //
216             // panel3
217             //
218             this.panel3.Controls.Add(this.lblHeader);
219             this.panel3.Controls.Add(this.lblSubheading);
220             this.panel3.Location = new System.Drawing.Point(12, 85);
221             this.panel3.Name = "panel3";
222             this.panel3.Size = new System.Drawing.Size(776, 92);
223             this.panel3.TabIndex = 4;
224             //
225             // panel4
226             //
227             this.panel4.BackgroundImage = global::
    ↪ A_level_Computer_Science_Project.Properties.Resources.Logo_cropped;
228             this.panel4.BackgroundImageLayout = System.Windows.Forms.
    ↪ ImageLayout.Zoom;
```

```
229             this.panel4.Controls.Add(this.button1);
230             this.panel4.Location = new System.Drawing.Point(12, 12);
231             this.panel4.Name = "panel4";
232             this.panel4.Size = new System.Drawing.Size(776, 67);
233             this.panel4.TabIndex = 5;
234             //
235             // button1
236             //
237             this.button1.BackColor = System.Drawing.Color.FromArgb(((int)(((
    ↪ byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
238             this.button1.BackgroundImage = global::
    ↪ A_level_Computer_Science_Project.Properties.Resources.cancel__1_;
239             this.button1.BackgroundImageLayout = System.Windows.Forms.
    ↪ ImageLayout.Zoom;
240             this.button1.FlatStyle = System.Windows.Forms.FlatStyle.Popup;
241             this.button1.Location = new System.Drawing.Point(0, 0);
242             this.button1.Name = "button1";
243             this.button1.Size = new System.Drawing.Size(75, 67);
244             this.button1.TabIndex = 0;
245             this.button1.UseVisualStyleBackColor = false;
246             this.button1.Click += new System.EventHandler(this.button1_Click
    ↪ );
247             //
248             // frmChoiceBox
249             //
250             this.AcceptButton = this.btnRandomChoice;
251             this.AutoScaleDimensions = new System.Drawing.SizeF(7F, 15F);
252             this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
253             this.BackColor = System.Drawing.Color.FromArgb(((int)(((byte)
    ↪ (31)))), ((int)(((byte)(31)))), ((int)(((byte)(31)))));
254             this.ClientSize = new System.Drawing.Size(800, 450);
255             this.Controls.Add(this.panel4);
256             this.Controls.Add(this.panel3);
257             this.Controls.Add(this.panel2);
258             this.Controls.Add(this.panel1);
259             this.Icon = ((System.Drawing.Icon)(resources.GetObject("$this.
    ↪ Icon")));
260             this.Name = "frmChoiceBox";
261             this.Text = "Virtual-Life Choice Box";
262             this.Load += new System.EventHandler(this.frmChoiceBox_Load);
263             this.panel1.ResumeLayout(false);
264             this.panel2.ResumeLayout(false);
265             this.panel2.PerformLayout();
266             this.panel3.ResumeLayout(false);
267             this.panel3.PerformLayout();
268             this.panel4.ResumeLayout(false);
269             this.ResumeLayout(false);
270 
271         }
272 
273         #endregion
274 
275         private System.Windows.Forms.Label lblHeader;
276         private System.Windows.Forms.Label lblSubheading;
277         private System.Windows.Forms.Panel panel1;
278         private System.Windows.Forms.Button btnRandomChoice;
279         private System.Windows.Forms.Button btnChoice3;
280         private System.Windows.Forms.Button btnChoice2;
281         private System.Windows.Forms.Button btnChoice1;
282         private System.Windows.Forms.Panel panel2;
283         private System.Windows.Forms.Label lblChoice3;
```

```
        private System.Windows.Forms.Label lblChoice2Desc;
        private System.Windows.Forms.Label lblChoice3Desc;
        private System.Windows.Forms.Label lblChoice2;
        private System.Windows.Forms.Label lblChoice1Desc;
        private System.Windows.Forms.Label lblChoice1;
        private System.Windows.Forms.Panel panel3;
        private System.Windows.Forms.Panel panel4;
        private System.Windows.Forms.Button button1;
    }
}
```

Listing A.6: FILE/frmChoiceBox.Designer.cs

A.1.4 Loading Bar

[FILE]frmLoadingBar.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;

namespace A_level_Computer_Science_Project
{
    public partial class frmLoadingBar : Form
    {
        public frmLoadingBar()
        {
            InitializeComponent();
        }

        private void frmLoadingBar_Load(object sender, EventArgs e)
        {
            timer1.Start();

        }

        private void label1_Click(object sender, EventArgs e)
        {

        }
    }
}
```

Listing A.7: FILE/frmLoadingBar.cs

[FILE]frmLoadingBar.Designer.cs

```

namespace A_level_Computer_Science_Project
{
    partial class frmLoadingBar
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
```

```
12          /// Clean up any resources being used.
13          /// </summary>
14          /// <param name="disposing">true if managed resources should be
    ↪ disposed; otherwise, false.</param>
15          protected override void Dispose(bool disposing)
16          {
17              if (disposing && (components != null))
18              {
19                  components.Dispose();
20              }
21              base.Dispose(disposing);
22          }
23
24          #region Windows Form Designer generated code
25
26          /// <summary>
27          /// Required method for Designer support - do not modify
28          /// the contents of this method with the code editor.
29          /// </summary>
30          private void InitializeComponent()
31          {
32              this.components = new System.ComponentModel.Container();
33              System.ComponentModel.ComponentResourceManager resources = new
    ↪ System.ComponentModel.ComponentResourceManager(typeof(frmLoadingBar));
34              this.progressBar1 = new System.Windows.Forms.ProgressBar();
35              this.timer1 = new System.Windows.Forms.Timer(this.components);
36              this.label1 = new System.Windows.Forms.Label();
37              this.SuspendLayout();
38              //
39              // progressBar1
40              //
41              this.progressBar1.ForeColor = System.Drawing.Color.FromArgb(((
    ↪ int)(((byte)(100)))), ((int)(((byte)(164)))), ((int)(((byte)(209)))));
42              this.progressBar1.Location = new System.Drawing.Point(12, 114);
43              this.progressBar1.MarqueeAnimationSpeed = 10;
44              this.progressBar1.Name = "progressBar1";
45              this.progressBar1.Size = new System.Drawing.Size(376, 38);
46              this.progressBar1.Style = System.Windows.Forms.ProgressBarStyle.
    ↪ Marquee;
47              this.progressBar1.TabIndex = 0;
48              //
49              // label1
50              //
51              this.label1.AutoSize = true;
52              this.label1.BackColor = System.Drawing.Color.Transparent;
53              this.label1.Font = new System.Drawing.Font("Segoe UI", 21.75F,
    ↪ System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point);
54              this.label1.ForeColor = System.Drawing.Color.White;
55              this.label1.Location = new System.Drawing.Point(111, 43);
56              this.label1.Name = "label1";
57              this.label1.Size = new System.Drawing.Size(158, 40);
58              this.label1.TabIndex = 1;
59              this.label1.Text = "LOADING...";
60              this.label1.Click += new System.EventHandler(this.label1_Click);
61              //
62              // frmLoadingBar
63              //
64              this.AutoScaleDimensions = new System.Drawing.SizeF(7F, 15F);
65              this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
66              this.BackColor = System.Drawing.Color.FromArgb(((int)(((byte)
    ↪ (31)))), ((int)(((byte)(31)))), ((int)(((byte)(31)))));
```

```
            this.ClientSize = new System.Drawing.Size(400, 225);
            this.Controls.Add(this.label1);
            this.Controls.Add(this.progressBar1);
            this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.
    ↪ FixedSingle;
            this.Icon = ((System.Drawing.Icon)(resources.GetObject("$this.
    ↪ Icon")));
            this.MaximizeBox = false;
            this.MaximumSize = new System.Drawing.Size(416, 264);
            this.MinimizeBox = false;
            this.Name = "frmLoadingBar";
            this.StartPosition = System.Windows.Forms.FormStartPosition.
    ↪ CenterScreen;
            this.Text = "Loading";
            this.TopMost = true;
            this.WindowState = System.Windows.Forms.FormWindowState.
    ↪ Maximized;
            this.Load += new System.EventHandler(this.frmLoadingBar_Load);
            this.ResumeLayout(false);
            this.PerformLayout();

        }

        #endregion

        private System.Windows.Forms.ProgressBar progressBar1;
        private System.Windows.Forms.Timer timer1;
        private System.Windows.Forms.Label label1;
    }
}
```

Listing A.8: FILE/frmLoadingBar.Designer.cs

A.1.5 Main Game Screen

[FILE]frmMainGameScreen.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using Newtonsoft.Json.Linq;
using System.IO;
using Newtonsoft.Json;

namespace A_level_Computer_Science_Project
{
    public partial class frmMainGameScreen : Form
    {
        MainCharacter mainCharacter; //setup the object to put main
    ↪ character into when its transferred
        Event[] eventArray;
        ControlClass controlClass;
        GenericFamilyMember[] familyArray;
        Score mainCharacterScores;
        public Job[] availableJobs = new Job[100];
        public Partner[] potentialPartners = new Partner[15];
        Partner partner = new Partner();

```

```
25          public frmMainGameScreen(
26              MainCharacter mainCharacterTransfer,
27              Event[] eventArrayTransfer,
28              ControlClass controlClassTransfer,
29              GenericFamilyMember[] familyArrayTransfer,
30              Score mainCharacterScoresTransfer,
31              Partner partnerTransfer
32              )
33          {
34              InitializeComponent();
35              mainCharacter = mainCharacterTransfer; //Set the contents of
    ↪ mainCharacter to mainCharacterTransfer which has come from previous
    ↪ form
36              eventArray = eventArrayTransfer;
37              controlClass = controlClassTransfer;
38              familyArray = familyArrayTransfer;
39              mainCharacterScores = mainCharacterScoresTransfer;
40              partner = partnerTransfer;
41          }
42
43
44          private void btnSaveToFile_Click(object sender, EventArgs e)
45          {
46              Job newJob = new Job();
47              newJob.generateJob(1);
48              MessageBox.Show(newJob.JobTitle);
49
50          }
51
52          public void saveToFile()
53          {
54              //taken from Stack Overflow - it seems to work, don't question
    ↪ it!
55              File.WriteAllText(controlClass.Filepath + "\\mainCharacter.json"
    ↪ , JsonConvert.SerializeObject(mainCharacter));
56              File.WriteAllText(controlClass.Filepath + "\\eventArray.json",
    ↪ JsonConvert.SerializeObject(eventArray));
57              File.WriteAllText(controlClass.Filepath + "\\controlClass.json",
    ↪  JsonConvert.SerializeObject(controlClass));
58              File.WriteAllText(controlClass.Filepath + "\\familyArray.json",
    ↪ JsonConvert.SerializeObject(familyArray));
59              File.WriteAllText(controlClass.Filepath + "\\mainCharacterScores
    ↪ .json", JsonConvert.SerializeObject(mainCharacterScores));
60              File.WriteAllText(controlClass.Filepath + "\\partner.json",
    ↪ JsonConvert.SerializeObject(partner));
61          }
62
63          private void btnSaveAndHome_Click(object sender, EventArgs e)
64          {
65              frmLoadingBar fLoadingBar = new frmLoadingBar();
66              fLoadingBar.Show();
67              saveToFile();
68              frmMainMenu fMainMenu = new frmMainMenu();
69              fMainMenu.Show();
70              fLoadingBar.Close();
71              this.Hide();
72              fLoadingBar.Close();
73          }
74
75          private void frmMainGameScreen_Load(object sender, EventArgs e)
76          {
```

```
77              saveToFile();
78              //populate the main character info on the first tab
79              populateForms();
80              yearlyCheck();
81          }
82
83          private void tabControl1_SelectedIndexChanged(object sender,
    ↪ EventArgs e)
84          {
85
86          }
87
88          public void populateMainCharacterInfo()
89          {
90              lblMPFirstName.Text = "First name: " + mainCharacter.FirstName;
91              lblMPLastName.Text = "Last name: " + mainCharacter.LastName;
92              lblMPGender.Text = "Gender: " + mainCharacter.Gender;
93              lblMPSexuality.Text = "Sexuality: " + mainCharacter.Sexuality;
94              lblMPAge.Text = "Age: " + mainCharacter.Age.ToString();
95              lblMPDateOfBirth.Text = "Date Of Birth: " + mainCharacter.
    ↪ DateOfBirth.ToShortDateString();
96          }
97          public void populateEventBox()
98          {
99              txtEventBox.ResetText();
100             for(int i=0; i< controlClass.NextEvent; i++)
101             {
102                 //txtEventBox.Text = txtEventBox.Text + Environment.NewLine
    ↪ + eventArray[i].DateHappened.ToShortDateString() + " - " +  eventArray
    ↪ [i].Description;
103                 txtEventBox.AppendText(Environment.NewLine + eventArray[i].
    ↪ DateHappened.ToShortDateString() + " - " + eventArray[i].Description);
104             }
105         }
106
107         public void populateForms()
108         {
109             //populate all forms, this can replace multiple lines in both
    ↪ age up and load.
110
111             populateMainCharacterInfo();
112             updateEducationPage();
113             updateCrimes();
114             updateInGameDate();
115             populateDGVFamily();
116             populateEventBox();
117             updateCharCustomisationOptions();
118             updateCurrentJobInformation();
119             updatePartnerInformation();
120
121             updateScoresPB(); // this will need to be the last function
    ↪ called so all scores have had a chance to update before they are
    ↪ displayed.
122         }
123
124         //============================AGE UP
    ↪ ========================================
125         private void btnAgeUp_Click(object sender, EventArgs e)
126         {
127             //Increase main character age by 1 and create event for it.
128             mainCharacter.Age = mainCharacter.Age + 1;
```

```
129             controlClass.InGameDate = controlClass.InGameDate.AddYears(1);
130             eventArray[controlClass.NextEvent].DateHappened = controlClass.
    ↪ InGameDate;
131             eventArray[controlClass.NextEvent].Description = "You turned " +
    ↪  mainCharacter.Age.ToString();
132             eventArray[controlClass.NextEvent].Category = "Main character
    ↪ birthday";
133             controlClass.NextEvent++;
134             if(mainCharacter.InRelationship == true)
135             {
136                 partner.Age = partner.Age + 1;
137             }
138             //decide if main character should die or not.
139             bool returnedDeathMC = deathCheck(mainCharacter.Age);
140             if(returnedDeathMC == true)
141             {
142                 killMainCharacter();
143             }
144             else
145             {
146                 //if maincharacter hasn't just died, progress with the aging
    ↪  up
147 
148                 //first check for any key life events (can drive, able to go
    ↪  to school etc)
149                 lifeEvents();
150                 yearlyCheck();
151 
152 
153                 //Age up all familiy members
154                 ageUpFamilyArray();
155                 //check kill family members
156                 checkKillGenericFamilyMember();
157 
158 
159                 //education
160                 checkEducationStatus();
161                 generateMedicalScore();
162 
163 
164                 //job
165                 if(mainCharacter.InJob == true)
166                 {
167                     //increase job duration
168                     mainCharacter.JobCurrentDuration = mainCharacter.
    ↪ JobCurrentDuration + 1;
169                     calculateSalary();
170                 }
171 
172                 updateHappinessScore(0);
173             }
174             //some bits need to happen regardless of if the main charcter
    ↪ dead or not, as will need info from them.
175             populateForms();
176 
177 
178             saveToFile();
179         }
180         //
    ↪ ================================================================================
    ↪
```

```
181         public void updateInGameDate()
182         {
183             lblInGameDate.Text = "Date in game: " + controlClass.InGameDate.
    ↪ ToShortDateString();
184         }
185 
186         public void populateDGVFamily()
187         {
188             //first have to clear it
189             dgvFamily.Rows.Clear();
190             dgvFamily.Refresh();
191 
192             //dgvFamily.DataSource = familyArray;
193 
194             //use code from openldbws app to populate dgvFamily
195 
196             //first, set var for which column contains which data
197             int relationshipCol = 0;
198             int firstNameCol = 1;
199             int lastNameCol = 2;
200 
201             //now loop through familyArray and populate dgvFamily
202             // can use a for loop as number of family members are in the
    ↪ controlClass
203 
204             for(int x=0; x<controlClass.NumberOfFamily; x++)
205             {
206                 if (familyArray[x].FirstName != null)
207                 {
208                     int i = dgvFamily.Rows.Add();
209                     dgvFamily.Rows[i].Cells[relationshipCol].Value =
    ↪ familyArray[x].RelationshipToMain;
210                     dgvFamily.Rows[i].Cells[firstNameCol].Value =
    ↪ familyArray[x].FirstName;
211                     dgvFamily.Rows[i].Cells[lastNameCol].Value = familyArray
    ↪ [x].LastName;
212                 }
213             }
214         }
215 
216         private void dgvFamily_CellContentClick(object sender,
    ↪ DataGridViewCellEventArgs e)
217         {
218            // MessageBox.Show(dgvFamily.CurrentCell.RowIndex.ToString());
219             int currentIndex = dgvFamily.CurrentCell.RowIndex;
220             //MessageBox.Show(familyArray[currentIndex].Age.ToString());
221 
222             //populate info on form
223             lblFamRealtionshipToMain.Text = mainCharacter.FirstName + "'s "
    ↪ + familyArray[currentIndex].RelationshipToMain;
224             lblFamFirstName.Text = "First Name: " + familyArray[currentIndex
    ↪ ].FirstName;
225             lblFamLastName.Text = "Last Name: " + familyArray[currentIndex].
    ↪ LastName;
226             lblFamGender.Text = "Gender: " + familyArray[currentIndex].
    ↪ Gender;
227             lblFamSexuality.Text = "Sexuality: " + familyArray[currentIndex
    ↪ ].Sexuality;
228             lblFamAge.Text = "Age: " + familyArray[currentIndex].Age.
    ↪ ToString();
229             lblFamLivingStatus.Text = "Living Status: " + familyArray[
```

```
currentIndex].LivingStatus.ToString();
            lblFamDateOfBirth.Text = "Date Of Birth: " + familyArray[
currentIndex].DateOfBirth.ToShortDateString();
            lblFamInRelationship.Text = "In Relationship: " + familyArray[
currentIndex].InRelationship.ToString();
            if(familyArray[currentIndex].LivingStatus == false)
            {
                //person is dead so can show death date and reason for death
                lblFamDateOfDeath.Text = "Date Of Death: " + familyArray[
currentIndex].DateOfDeath.ToShortDateString();
                lblFamReasonForDeath.Text = "Reason For Death: " +
familyArray[currentIndex].ReasonForDeath;
            }
            else
            {
                //person is not dead - don't show death date and reason for
death
                lblFamDateOfDeath.Text = "Date Of Death: N/A (haven't died
yet!)";
                lblFamReasonForDeath.Text = "Reason For Death: N/A (haven't
died yet!)";
            }
        }

        private void ageUpFamilyArray()
        {
            //loop through all the members of the family array and add 1 to
their age if livingstatus is true
            for(int x=0; x<controlClass.NumberOfFamily; x++)
            {
                if(familyArray[x].LivingStatus == true)
                {
                    //can age them up
                    familyArray[x].Age = familyArray[x].Age + 1;
                }
            }
        }


        public void killMainCharacter()
        {
            //very rough for now, will need to polish at some point

            disableClickables();
            mainCharacter.DateOfDeath = controlClass.InGameDate.AddDays(
randomNumber(1,300));
            mainCharacter.LivingStatus = false;
            mainCharacter.ReasonForDeath = generateReasonForDeath();
            //generate event showing this
            eventArray[controlClass.NextEvent].Category = "Death";
            eventArray[controlClass.NextEvent].DateHappened = controlClass.
InGameDate;
            eventArray[controlClass.NextEvent].Description = "You died due
to " + mainCharacter.ReasonForDeath;
            controlClass.NextEvent++;
            MessageBox.Show(mainCharacter.FirstName + " has died a tragic
death due to " + mainCharacter.ReasonForDeath, mainCharacter.FirstName
 + " has died", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            saveToFile();
        }

```

```
277         public string generateReasonForDeath()
278         {
279             //Generate a reason for death
280 
281             string[] reasonsForDeath = { "a heart attack", "eating too many
    ↪ oreos", "being clawed at by cat" , "slipping on a banana peel", "
    ↪ falling in the otter enclosure at a zoo and being licked to death", "
    ↪ dropping a toaster into the bath", "being crushed by a falling grand
    ↪ piano", "being strangled by a fez tassel", "walking into a cactus", "
    ↪ getting hit by a train", "being clubbed by old lady in the street with
    ↪  her walker", "being crushed by a stampede of hungry campers" };
282             Random rnd = new Random();
283             int deathReasonIndex = rnd.Next(0, (reasonsForDeath.Length));
284             string reasonForDeathReturn = reasonsForDeath[deathReasonIndex];
285             return reasonForDeathReturn;
286         }
287 
288         public void checkKillGenericFamilyMember()
289         {
290             //loop through all members of familyArray, passing their age
    ↪ into deathCheck. if return=true then kill, else move onto next.
291 
292             for(int x=0; x<controlClass.NextFamily; x++)
293             {
294                 if(familyArray[x].LivingStatus == true)
295                 {
296                     bool deathCheckReturn = deathCheck(familyArray[x].Age);
297                     if(deathCheckReturn == true)
298                     {
299                         //RIP - put death code here
300                         //MessageBox.Show("DEATH" + x.ToString());
301                         familyArray[x].DateOfDeath = controlClass.InGameDate
    ↪ .AddDays(randomNumber(1, 300));
302                         familyArray[x].LivingStatus = false;
303                         familyArray[x].ReasonForDeath =
    ↪ generateReasonForDeath();
304                         //Now gen event
305                         eventArray[controlClass.NextEvent].Category = "Death
    ↪ ";
306                         eventArray[controlClass.NextEvent].DateHappened =
    ↪ controlClass.InGameDate;
307                         eventArray[controlClass.NextEvent].Description =
    ↪ familyArray[x].FirstName + " (your " + familyArray[x].
    ↪ RelationshipToMain.ToLower() + ") died due to " + familyArray[x].
    ↪ ReasonForDeath;
308                         controlClass.NextEvent++;
309                         //now reduce mc happiness score
310                         mainCharacterScores.HappinessScore =
    ↪ mainCharacterScores.HappinessScore - randomNumber(1, 100);
311                         saveToFile();
312 
313 
314                     }
315                 }
316 
317             }
318 
319         }
320 
321         public bool deathCheck(int age)
322         {
```

```
323             bool deathBool = false;
324             int deathChance = 1;
325             //for as main char gets older, their death chance increases
326             /* 00-10 1/30
327              * 11-16 1/25
328              * 17-30 1/15
329              * 31-50 1/25
330              * 51-67 1/20
331              * 68-80 1/15
332              * 81-100 1/8
333              * >100 1/3
334              */
335             if (age <= 10)
336             {
337                 deathChance = 30;
338             }
339             else if (age >= 11 && age <= 16)
340             {
341                 deathChance = 25;
342             }
343             else if (age >= 17 && age <= 30)
344             {
345                 deathChance = 15;
346             }
347             else if (age >= 31 && age <= 50)
348             {
349                 deathChance = 25;
350             }
351             else if (age >= 51 && age <= 67)
352             {
353                 deathChance = 20;
354             }
355             else if (age >= 68 && age <= 80)
356             {
357                 deathChance = 15;
358             }
359             else if (age >= 81 && age <= 99)
360             {
361                 deathChance = 8;
362             }
363             else if (age >= 100)
364             {
365                 deathChance = 3;
366             }
367             //now generate random number based off of 1 and deathChance
368             Random rnd = new Random();
369             int randomReturn = rnd.Next(1, deathChance);
370             if (randomReturn == 2)
371             {
372                 //DEATH
373                 deathBool = true;
374             }
375             else
376             {
377                 deathBool = false;
378             }
379 
380             //return deathBool to main program where it will be processed.
381             return deathBool;
382         }
383 
```

```
384         public void updateScoresPB()
385         {
386             if (mainCharacterScores.EducationScore < prbEducationScore.
    ↪ Maximum && mainCharacterScores.EducationScore > prbEducationScore.
    ↪ Minimum)
387             {
388                 prbEducationScore.Value = mainCharacterScores.EducationScore
    ↪ ;
389             }
390             if(mainCharacterScores.HappinessScore < prbHappinessScore.
    ↪ Maximum && mainCharacterScores.HappinessScore > prbHappinessScore.
    ↪ Minimum)
391             {
392                 prbHappinessScore.Value = mainCharacterScores.HappinessScore
    ↪ ;
393             }
394             if (mainCharacterScores.MedicalScore < prbMedicalScore.Maximum
    ↪ && mainCharacterScores.MedicalScore > prbMedicalScore.Minimum)
395             {
396                 prbMedicalScore.Value = mainCharacterScores.MedicalScore;
397             }
398             if (mainCharacterScores.LifeScore < prbLifeScore.Maximum &&
    ↪ mainCharacterScores.LifeScore > prbLifeScore.Minimum)
399             {
400                 prbLifeScore.Value = mainCharacterScores.LifeScore;
401             }
402             ttMainToolTip.SetToolTip(prbEducationScore, prbEducationScore.
    ↪ Value.ToString());
403             ttMainToolTip.SetToolTip(prbHappinessScore, prbHappinessScore.
    ↪ Value.ToString());
404             ttMainToolTip.SetToolTip(prbJobScore, prbJobScore.Value.ToString
    ↪ ());
405             ttMainToolTip.SetToolTip(prbLifeScore, prbLifeScore.Value.
    ↪ ToString());
406             ttMainToolTip.SetToolTip(prbMedicalScore, prbMedicalScore.Value.
    ↪ ToString());
407         }
408 
409         public void checkEducationStatus()
410         {
411             if(mainCharacter.InEducation == true)
412             {
413                 int age = mainCharacter.Age;
414                 DateTime mcBirthday = mainCharacter.DateOfBirth;
415                 DateTime dateOfEvent;
416                 if (mcBirthday.DayOfYear < 245)
417                 {
418                     //mc is born before 2nd sept therefor events need to be
    ↪ for this september
419                     int yearOfEvent = controlClass.InGameDate.Year;
420                     int monthOfEvent = 9;
421                     int dayOfEvent = 1;
422                     dateOfEvent = new DateTime(yearOfEvent, monthOfEvent,
    ↪ dayOfEvent);
423                 }
424                 else
425                 {
426                     //mc is born after 2nd sept therfore events need to be
    ↪ for the following september
427                     int yearOfEvent = (controlClass.InGameDate.Year) + 1;
428                     int monthOfEvent = 9;
```

```
429                     int dayOfEvent = 1;
430                     dateOfEvent = new DateTime(yearOfEvent, monthOfEvent,
    ↪ dayOfEvent);
431                 }
432                 //need to get date of 1st september after mcs birthday
433                 switch (age)
434                 {
435                     case 4:
436                         //move into reception
437                         string nameOfSchool = genNameOfSchool(1);
438                         eventArray[controlClass.NextEvent].Category = "
    ↪ Education";
439                         eventArray[controlClass.NextEvent].DateHappened =
    ↪ dateOfEvent;
440                         eventArray[controlClass.NextEvent].Description = "
    ↪ Start infant school in Reception. Your parents chose " + nameOfSchool
    ↪ + " for you.";
441                         controlClass.NextEvent++;
442                         controlClass.EduScorePlus = 100;
443                         calculateEducationScore();
444                         controlClass.CurrentSchoolYear = 0;
445                         break;
446                     case 5:
447                         //move into yr 1
448                         eventArray[controlClass.NextEvent].Category = "
    ↪ Education";
449                         eventArray[controlClass.NextEvent].DateHappened =
    ↪ dateOfEvent;
450                         eventArray[controlClass.NextEvent].Description = "
    ↪ Move into Year 1 at infant school.";
451                         controlClass.NextEvent++;
452                         calculateEducationScore();
453                         controlClass.CurrentSchoolYear = 1;
454                         break;
455                     case 6:
456                         //move into yr 2
457                         eventArray[controlClass.NextEvent].Category = "
    ↪ Education";
458                         eventArray[controlClass.NextEvent].DateHappened =
    ↪ dateOfEvent;
459                         eventArray[controlClass.NextEvent].Description = "
    ↪ Move into year 2 at infant school. This is your last year there. Your
    ↪ parents are frantically trying to find a place at Junior school for
    ↪ you.";
460                         controlClass.NextEvent++;
461                         calculateEducationScore();
462                         controlClass.CurrentSchoolYear = 2;
463                         break;
464                     case 7:
465                         //move into yr 3(need to pick new school here)
466                         string nameOfSchool3 = genNameOfSchool(2);
467                         eventArray[controlClass.NextEvent].Category = "
    ↪ Education";
468                         eventArray[controlClass.NextEvent].DateHappened =
    ↪ dateOfEvent;
469                         eventArray[controlClass.NextEvent].Description = "
    ↪ Start year 3 at Junior school. You make lots of friends very quickly.
    ↪ Your parents managed to find you a place at " + nameOfSchool3 + ".";
470                         controlClass.NextEvent++;
471                         calculateEducationScore();
472                         controlClass.CurrentSchoolYear = 3;
```

```
473                         break;
474                     case 8:
475                         //move into yr 4
476                         eventArray[controlClass.NextEvent].Category = "
    ↪ Education";
477                         eventArray[controlClass.NextEvent].DateHappened =
    ↪ dateOfEvent;
478                         eventArray[controlClass.NextEvent].Description = "
    ↪ Move into year 4 at Junior School.";
479                         controlClass.NextEvent++;
480                         calculateEducationScore();
481                         controlClass.CurrentSchoolYear = 4;
482                         break;
483                     case 9:
484                         //move into yr 5
485                         eventArray[controlClass.NextEvent].Category = "
    ↪ Education";
486                         eventArray[controlClass.NextEvent].DateHappened =
    ↪ dateOfEvent;
487                         eventArray[controlClass.NextEvent].Description = "
    ↪ Move into year 5 at Junior School.";
488                         controlClass.NextEvent++;
489                         calculateEducationScore();
490                         controlClass.CurrentSchoolYear = 5;
491                         break;
492                     case 10:
493                         //move into yr 6
494                         eventArray[controlClass.NextEvent].Category = "
    ↪ Education";
495                         eventArray[controlClass.NextEvent].DateHappened =
    ↪ dateOfEvent;
496                         eventArray[controlClass.NextEvent].Description = "
    ↪ Move into year 6 at Junior School. You have SATS exams at the end of
    ↪ this year.";
497                         controlClass.NextEvent++;
498                         calculateEducationScore();
499                         controlClass.CurrentSchoolYear = 6;
500                         break;
501                     case 11:
502                         //move into yr 7
503                         string nameOfSchool7 = genNameOfSchool(3);
504                         eventArray[controlClass.NextEvent].Category = "
    ↪ Education";
505                         eventArray[controlClass.NextEvent].DateHappened =
    ↪ dateOfEvent;
506                         eventArray[controlClass.NextEvent].Description = "
    ↪ Start year 7 at Secondary School. You don't make many friends to begin
    ↪  with. You are lucky enough to be attending " + nameOfSchool7 + ",
    ↪ without that persuasive letter your dad sent, this wouldn't have
    ↪ happned.";
507                         controlClass.NextEvent++;
508                         calculateEducationScore();
509                         controlClass.CurrentSchoolYear = 7;
510                         break;
511                     case 12:
512                         //move into yr 8
513                         eventArray[controlClass.NextEvent].Category = "
    ↪ Education";
514                         eventArray[controlClass.NextEvent].DateHappened =
    ↪ dateOfEvent;
515                         eventArray[controlClass.NextEvent].Description = "
```

```
    ↪ Start yer 8 at Secondary School. You now have lots of friends..";
516                     controlClass.NextEvent++;
517                     calculateEducationScore();
518                     controlClass.CurrentSchoolYear = 8;
519                     break;
520                 case 13:
521                     //more into yr 9
522                     eventArray[controlClass.NextEvent].Category = "
    ↪ Education";
523                     eventArray[controlClass.NextEvent].DateHappened =
    ↪ dateOfEvent;
524                     eventArray[controlClass.NextEvent].Description = "
    ↪ Start year 9 at Secondary school.";
525                     controlClass.NextEvent++;
526                     calculateEducationScore();
527                     controlClass.CurrentSchoolYear = 9;
528                     break;
529                 case 14:
530                     //move into yr 10
531                     eventArray[controlClass.NextEvent].Category = "
    ↪ Education";
532                     eventArray[controlClass.NextEvent].DateHappened =
    ↪ dateOfEvent;
533                     eventArray[controlClass.NextEvent].Description = "
    ↪ Start year 10 at Secondary school.";
534                     controlClass.NextEvent++;
535                     calculateEducationScore();
536                     controlClass.CurrentSchoolYear = 10;
537                     break;
538                 case 15:
539                     //move into yr 11
540                     eventArray[controlClass.NextEvent].Category = "
    ↪ Education";
541                     eventArray[controlClass.NextEvent].DateHappened =
    ↪ dateOfEvent;
542                     eventArray[controlClass.NextEvent].Description = "
    ↪ Start year 11 at Secondary school. You have GCSE exams at the end of
    ↪ the year.";
543                     controlClass.NextEvent++;
544                     calculateEducationScore();
545                     controlClass.CurrentSchoolYear = 11;
546                     break;
547                 case 16:
548                     //move into yr 12
549                     string nameOfSchool12 = genNameOfSchool(4);
550                     eventArray[controlClass.NextEvent].Category = "
    ↪ Education";
551                     eventArray[controlClass.NextEvent].DateHappened =
    ↪ dateOfEvent;
552                     eventArray[controlClass.NextEvent].Description = "
    ↪ Start year 12 at College. Your parents wanted you to take English,
    ↪ Maths and Chemistry but you are taking Drama, music and Physics. You'
    ↪ re now attending " + nameOfSchool12 + ".";
553                     controlClass.NextEvent++;
554                     calculateEducationScore();
555                     controlClass.CurrentSchoolYear = 12;
556                     break;
557                 case 17:
558                     //move into yr 13
559                     eventArray[controlClass.NextEvent].Category = "
    ↪ Education";
```

```
560                         eventArray[controlClass.NextEvent].DateHappened =
    ↪ dateOfEvent;
561                         eventArray[controlClass.NextEvent].Description = "
    ↪ Start year 13 at College. You have your A-levels at the end of this
    ↪ year.";
562                         controlClass.NextEvent++;
563                         calculateEducationScore();
564                         controlClass.CurrentSchoolYear = 13;
565                         break;
566                     default:
567                         //no change to school situation
568                         break;
569                 }
570             }
571 
572         }
573         public void updateEducationPage()
574         {
575             //first update the events box
576             txtEduEducationEvents.ResetText();
577             for (int i = 0; i < controlClass.NextEvent; i++)
578             {
579                 if(eventArray[i].Category == "Education")
580                 {
581                     txtEduEducationEvents.Text = txtEduEducationEvents.Text
    ↪ + Environment.NewLine + eventArray[i].DateHappened.ToShortDateString()
    ↪  + " - " + eventArray[i].Description;
582                 }
583 
584             }
585 
586             //now update the lables
587             lblEduCurrentEducationPoints.Text = "Current Education Points: "
    ↪  + mainCharacterScores.EducationScore.ToString();
588             lblEduEducationSituation.Text = "Current Education Situation: At
    ↪  school - " + controlClass.NameOfCurrentSchool;
589             lblEduCurrentYear.Text = "Current Year: " + controlClass.
    ↪ CurrentSchoolYear.ToString();
590         }
591 
592         public int randomNumber(int lower, int upper)
593         {
594             Random rnd = new Random();
595             int random = rnd.Next(lower, upper);
596             return random;
597         }
598 
599         public int choiceBox(int typeOfChoice, string opt1, string opt2,
    ↪ string opt3)
600         {
601             /*
602              * Will need to pass the following into this function and into
    ↪ the choice box form.
603              * -type of choice (this will determine the header and subheader
    ↪  text)
604              * --> using this rather than passing all information in to
    ↪ simplifiy calls to this function.
605              * -opt1
606              * -opt2
607              * -opt3
608              */
```

```
609
610             //declare vars used for passing data to choice box
611             string headerText = "HeaderText";
612             string subheaderText = "SubheaderText";
613             string opt1header = "Choice 1";
614             string opt1body = opt1;
615             string opt2header = "Choice 2";
616             string opt2body = opt2;
617             string opt3header = "Choice 3";
618             string opt3body = opt3;
619
620
621             //first setup the typeOfChoice
622             switch (typeOfChoice)
623             {
624                 case 1:
625                     //choose infant school
626                     headerText = "Infant school";
627                     subheaderText = "Select the infant school you want to
    ↪ spend the next 3 years at";
628                     break;
629                 case 2:
630                     //choose junior school
631                     headerText = "Junior school";
632                     subheaderText = "Select the junior school you want to
    ↪ spend the next 4 years at";
633                     break;
634                 case 3:
635                     //choose secondary school
636                     headerText = "Secondary school";
637                     subheaderText = "Select the secondary school you want to
    ↪  spend the next 5 years at";
638                     break;
639                 case 4:
640                     //choose college
641                     headerText = "College";
642                     subheaderText = "Select the college you want to spend
    ↪ the next 2 years at";
643                     break;
644                 default:
645                     //no option found
646                     break;
647             }
648
649             using (var form = new frmChoiceBox(headerText, subheaderText,
    ↪ opt1header, opt1body, opt2header, opt2body, opt3header, opt3body))
650             {
651
652                 this.Hide();
653                 var result = form.ShowDialog();
654
655                 this.Show();
656                 if (result == DialogResult.OK)
657                 {
658                     int val = form.ReturnValue1;            //values
    ↪ preserved after close
659                     //Do something here with these values
660
661                     //for example
662
663                     return (val);
```

```
664                 }
665
666             }
667             return 0;
668         }
669
670         public string genNameOfSchool(int typeOfSchool)
671         {
672             Random rnd = new Random();
673             //first declare array contianing different names of schools,
    ↪ then randomly generate three to be used.
674             string[] namesOfSchools = {"Westwood comprehensive", "Oak Ridge
    ↪ academy", "Eastview Institute", "Westview Institute", "Lone Oak all-
    ↪ through", "Sunset valley school", "Oceanside", "Cape Coral technical
    ↪ school", "Big pine institute", "Mapel Hills academy" };
675
676             //first set the options for return
677             string option1 = namesOfSchools[rnd.Next(1, 9)];
678             string option2 = namesOfSchools[rnd.Next(1, 9)];
679             string option3 = namesOfSchools[rnd.Next(1, 9)];
680             int returnedValue = choiceBox(typeOfSchool, option1, option2,
    ↪ option3);
681             switch (returnedValue)
682             {
683                 case 1:
684                     controlClass.NameOfCurrentSchool = option1;
685                     break;
686                 case 2:
687                     controlClass.NameOfCurrentSchool = option2;
688                     break;
689                 case 3:
690                     controlClass.NameOfCurrentSchool = option3;
691                     break;
692             }
693
694             return controlClass.NameOfCurrentSchool;
695
696         }
697
698         private void btnEduWithdraw_Click(object sender, EventArgs e)
699         {
700             mainCharacter.InEducation = false;
701             // eventArray[controlClass.NextEvent].Category = "Education";
702             //eventArray[controlClass.NextEvent].DateHappened = controlClass
    ↪ .InGameDate;
703             //eventArray[controlClass.NextEvent].Description = "You
    ↪ permanently withdrew from education. You said it was too boring.";
704             //controlClass.NextEvent++;
705             generateEvent("Education", "You permanently withdrew from
    ↪ education. You said it was too boring.", controlClass.InGameDate);
706             mainCharacterScores.EducationScore = 0;
707
708         }
709
710         public void lifeEvents()
711         {
712             switch (mainCharacter.Age)
713             {
714                 case 2:
715                     controlClass.EduScorePlus = 12;
716                     break;
```

```
717                 case 4:
718                     mainCharacter.InEducation = true;
719                     break;
720             }
721         }
722 
723         public void calculateEducationScore()
724         {
725             //Calculate the education score based off of the current score
    ↪ and the modifier
726             //1 in 8 chance of the score being decreased rather than being
    ↪ increased. When decreased, needs to go down a fair chunk.
727 
728             int randomReturn = randomNumber(1, 30);
729             if (randomReturn == 6)
730             {
731                 //Decrease
732                 mainCharacterScores.EducationScore = mainCharacterScores.
    ↪ EducationScore - controlClass.EduScorePlus;
733             }
734             else
735             {
736                 //increase
737                 //1 in 2 chance of modifier being upped by 0.3 of the
    ↪ current modifier.
738                 int randomReturn1 = randomNumber(1, 3);
739                 if(randomReturn1 == 1)
740                 {
741                     //normal modifier
742                     mainCharacterScores.EducationScore = mainCharacterScores
    ↪ .EducationScore + controlClass.EduScorePlus;
743                 }
744                 else if (randomReturn1 == 2)
745                 {
746                     //increased modifier
747                     int plus = (int)Math.Round(controlClass.EduScorePlus *
    ↪ 0.3);
748                     mainCharacterScores.EducationScore = mainCharacterScores
    ↪ .EducationScore + plus;
749                 }
750             }
751 
752         }
753 
754         public void generateEvent(string category, string content, DateTime
    ↪ date)
755         {
756             eventArray[controlClass.NextEvent].Category = category;
757             eventArray[controlClass.NextEvent].DateHappened = date;
758             eventArray[controlClass.NextEvent].Description = content;
759             controlClass.NextEvent++;
760         }
761 
762         public void updateHappinessScore(int externalModifier)
763         {
764             //add to the current happiness score the generic value to
    ↪ increse by plus a modifier
765             mainCharacterScores.HappinessScore = mainCharacterScores.
    ↪ HappinessScore + controlClass.HappinessScorePlus + externalModifier;
766 
767         }
```

```
768
769         public void disableClickables()
770         {
771             //procedure to disable all clickable elements on this form, for
    ↪ use when the main char dies.
772             //There is no way to reverse this.
773 
774             btnAgeUp.Enabled = false;
775             btnEduWithdraw.Enabled = false;
776             btnEndLife.Enabled = false;
777             btnJobApplyForJob.Enabled = false;
778             btnJobQuitJob.Enabled = false;
779         }
780 
781         private void btnCriCommitCrime_Click(object sender, EventArgs e)
782         {
783             //first generate new crime then time of community service for it
    ↪ .
784 
785             string[] crimes = { "Burgle a shop", "Steal a car", "Pirate a
    ↪ DVD" };
786             int crimesLength = crimes.Length;
787             int rand = randomNumber(0, crimesLength);
788             int years = randomNumber(1, 12);
789             string crimeText = "You committed a crime " + crimes[rand] + "
    ↪ and you recieved " + years.ToString() + " years community service.";
790             generateEvent("Crime", crimes[rand], controlClass.InGameDate);
791             updateCrimes();
792         }
793 
794         public void updateCrimes()
795         {
796             txtCriCrimeEvents.ResetText();
797             for (int i = 0; i < controlClass.NextEvent; i++)
798             {
799                 if (eventArray[i].Category == "Crime")
800                 {
801                     txtCriCrimeEvents.Text = txtCriCrimeEvents.Text +
    ↪ Environment.NewLine + eventArray[i].DateHappened.ToShortDateString() +
    ↪  " - " + eventArray[i].Description;
802                 }
803 
804             }
805         }
806 
807         public void yearlyCheck()
808         {
809             //this is a place which will run every time the mc ages up. If
    ↪ they are older than the specified age then will run code in relevant
    ↪ section.
810 
811             if(mainCharacter.Age >= 13)
812             {
813                 //'unlock' character customisation screen.
814                 txtEditFirstName.Enabled = true;
815                 txtEditLastName.Enabled = true;
816                 cboEditGender.Enabled = true;
817                 cboEditSexuality.Enabled = true;
818                 cboEditEyeColour.Enabled = true;
819                 cboEditHairColour.Enabled = true;
820             }
```

```
821             if (mainCharacter.Age >= 18)
822             {
823                 //'unlock' character job options
824                 btnJobApplyForJob.Enabled = true;
825                 btnJobQuitJob.Enabled = true;
826                 btnJobReshuffleJobBoard.Enabled = true;
827             }
828         }
829 
830         public void updateCharCustomisationOptions()
831         {
832             //fill all the Character Customisation option controls with
    ↪ current information from main char object
833             txtEditFirstName.Text = mainCharacter.FirstName;
834             txtEditLastName.Text = mainCharacter.LastName;
835             cboEditGender.Text = mainCharacter.Gender;
836             cboEditSexuality.Text = mainCharacter.Sexuality;
837             cboEditHairColour.Text = mainCharacter.HairColour;
838             cboEditEyeColour.Text = mainCharacter.EyeColour;
839         }
840 
841         public void saveCharCustomisationOptions()
842         {
843             //if the contents of the text boxes etc is NOT null and is
    ↪ different to the current contents of the main character object, then
    ↪ it needs to be written into the object
844 
845             //start with the first name
846             if(txtEditFirstName.Text != "" && txtEditFirstName.Text !=
    ↪ mainCharacter.FirstName)
847             {
848                 mainCharacter.FirstName = txtEditFirstName.Text;
849             }
850             //next - do the last name
851             if(txtEditLastName.Text != "" && txtEditLastName.Text !=
    ↪ mainCharacter.LastName)
852             {
853                 mainCharacter.LastName = txtEditLastName.Text;
854             }
855             if(cboEditGender.Text != "" && cboEditGender.Text !=
    ↪ mainCharacter.Gender)
856             {
857                 mainCharacter.Gender = cboEditGender.Text;
858             }
859             if(cboEditSexuality.Text != "" && cboEditSexuality.Text !=
    ↪ mainCharacter.Sexuality)
860             {
861                 mainCharacter.Sexuality = cboEditSexuality.Text;
862             }
863             if(cboEditHairColour.Text != "" && cboEditHairColour.Text !=
    ↪ mainCharacter.HairColour)
864             {
865                 mainCharacter.HairColour = cboEditHairColour.Text;
866             }
867             if(cboEditEyeColour.Text != "" && cboEditEyeColour.Text !=
    ↪ mainCharacter.EyeColour)
868             {
869                 mainCharacter.EyeColour = cboEditEyeColour.Text;
870             }
871         }
872 
```

```
873         private void btnEditSaveChanges_Click(object sender, EventArgs e)
874         {
875             saveCharCustomisationOptions();
876             populateForms();
877         }
878 
879         private void btnEndLife_Click(object sender, EventArgs e)
880         {
881             DialogResult dialogResult = MessageBox.Show("Are you sure you
    ↪ want to end this life?", "End life", MessageBoxButtons.YesNo);
882             if (dialogResult == DialogResult.Yes)
883             {
884                 killMainCharacter();
885             }
886             else if (dialogResult == DialogResult.No)
887             {
888                 //do something else
889             }
890 
891         }
892 
893         private void btnJobReshuffleJobBoard_Click(object sender, EventArgs
    ↪ e)
894         {
895             if(mainCharacter.JobCurrentDuration <= 5)
896             {
897                 //can only get band 1 job
898                 reshuffleJobs(1);
899             }
900             else if (mainCharacter.JobCurrentDuration >=6 && mainCharacter.
    ↪ JobCurrentDuration <= 14)
901             {
902                 //get band 2 job
903                 reshuffleJobs(2);
904             }
905             else if (mainCharacter.JobCurrentDuration >= 15 && mainCharacter
    ↪ .JobCurrentDuration <= 26)
906             {
907                 //get band 3 job
908                 reshuffleJobs(3);
909             }
910             else if (mainCharacter.JobCurrentDuration >= 27 && mainCharacter
    ↪ .JobCurrentDuration <= 44)
911             {
912                 //get band 4 job
913                 reshuffleJobs(4);
914             }
915             else if (mainCharacter.JobCurrentDuration >= 45)
916             {
917                 //get band 5 job
918                 reshuffleJobs(5);
919             }
920         }
921 
922         public void reshuffleJobs(int band)
923         {
924             //first generate the jobs
925             int maxNumb = 10;
926             for (int i = 0; i < maxNumb; i++)
927             {
928                 Job tempJob = new Job(); //make new instance of the object
```

```
929                 tempJob.generateJob(band);
930                 availableJobs[i] = tempJob; //transfer the temp object into
    ↪ the array.
931             }
932 
933             //now populate data grid view
934             //first have to clear it
935             dgvEmpAvailableJobs.Rows.Clear();
936             dgvEmpAvailableJobs.Refresh();
937 
938             //use code from openldbws app to populate dgvEmpAvailableJobs
939 
940             //first, set var for which column contains which data
941             int jtCol = 0;
942             int salCol = 1;
943             int bandCol = 2;
944 
945             //now loop through familyArray and populate dgvEmpAvailableJobs
946             //as jobs are only ever generated and populated here - can use
    ↪ max number from job generation loop for loop here
947 
948             for (int x = 0; x < maxNumb; x++)
949             {
950                 if (availableJobs[x].JobTitle != null)
951                 {
952                     int i = dgvEmpAvailableJobs.Rows.Add();
953                     dgvEmpAvailableJobs.Rows[i].Cells[jtCol].Value =
    ↪ availableJobs[x].JobTitle;
954                     dgvEmpAvailableJobs.Rows[i].Cells[salCol].Value =
    ↪ availableJobs[x].Salary;
955                     dgvEmpAvailableJobs.Rows[i].Cells[bandCol].Value =
    ↪ availableJobs[x].Band;
956                 }
957             }
958         }
959         public void updateCurrentJobInformation()
960         {
961             lblJobCurrentJobBand.Text = "Current Job Band: " + mainCharacter
    ↪ .JobBand.ToString(); ;
962             lblJobCurrentJobTitle.Text = "Current Job Title: " +
    ↪ mainCharacter.JobTitle;
963             lblJobCurrentJobSalary.Text = "Current Job Salary: " +
    ↪ mainCharacter.JobSalary.ToString();
964             lblJobCurrentJobDuration.Text = "Current Job Duration: " +
    ↪ mainCharacter.JobCurrentDuration.ToString();
965             mainCharacterScores.JobScore = mainCharacter.JobCurrentDuration
    ↪ *4;
966             prbJobScore.Value = mainCharacterScores.JobScore;
967         }
968 
969         private void btnJobApplyForJob_Click(object sender, EventArgs e)
970         {
971             try
972             {
973                 mainCharacter.JobTitle = dgvEmpAvailableJobs.SelectedCells
    ↪ [0].Value.ToString();
974                 mainCharacter.JobSalary = Int32.Parse(dgvEmpAvailableJobs.
    ↪ SelectedCells[1].Value.ToString());
975                 mainCharacter.JobBand = Int32.Parse(dgvEmpAvailableJobs.
    ↪ SelectedCells[2].Value.ToString());
976                 mainCharacter.InJob = true;
```

```
977                 generateEvent("Job", "You applied for and got the job: " +
    ↪ mainCharacter.JobTitle, controlClass.InGameDate);
978                 updateCurrentJobInformation();
979             } catch(Exception)
980             {
981                 MessageBox.Show("Please select a job before applying for it.
    ↪ ");
982             }
983 
984 
985         }
986 
987         private void button1_Click(object sender, EventArgs e) //quit job
988         {
989             try
990             {
991                 mainCharacter.JobTitle = "";
992                 mainCharacter.JobSalary = 0;
993                 mainCharacter.JobBand = 0;
994                 mainCharacter.InJob = false;
995                 mainCharacter.JobCurrentDuration = 0;
996                 generateEvent("Job", "You quit your job", controlClass.
    ↪ InGameDate);
997             }
998             catch (Exception)
999             {
1000                MessageBox.Show("Please make sure you have a job before you
    ↪ try to quit it.");
1001            }
1002            updateCurrentJobInformation();
1003        }
1004 
1005        public void calculateSalary()
1006        {
1007            Random rnd = new Random();
1008            //algorithm to generate the salary of the main character
1009            //needs to take into consideration the current band of job, base
    ↪  band of that salary and how long the mc has been in that job
1010            int salary = mainCharacter.JobSalary;
1011            int band = mainCharacter.JobBand;
1012            int duration = mainCharacter.JobCurrentDuration;
1013 
1014            if(mainCharacter.InJob == true)
1015            {
1016                //mc is in job so can now calc salary.
1017                int increaseAmount = rnd.Next(0, (band*duration)*50);
1018                mainCharacter.JobSalary = salary + increaseAmount;
1019            }
1020 
1021        }
1022 
1023        public void generateMedicalScore()
1024        {
1025            Random rnd = new Random();
1026            int currentMedScore = mainCharacterScores.MedicalScore;
1027            if(currentMedScore == 100)
1028            {
1029                //has perfect health, 1/10 chance of it being lowered so can
    ↪  then be diseased.
1030                int chance = rnd.Next(0, 11);
1031                if(chance == 5)
```

```
                {
                    currentMedScore = 99;
                }

            }
            if(currentMedScore < 100)
            {
                //can be diseased. 1 in 20 chance of getting disease
                int chance1 = rnd.Next(0, 21);
                if(chance1 == 12)
                {
                    //is gonna be diseased
                    string condition = getMedicalCondition();

                    generateEvent("Medical","You " + condition, controlClass
    ↪ .InGameDate);

                    mainCharacterScores.MedicalScore = mainCharacterScores.
    ↪ MedicalScore - rnd.Next(0, 50);

                }
            }
        }

        public string getMedicalCondition()
        {
            string[] medicalConditions = {"broke your arm", "broke your leg"
    ↪ , "stubbed your toe" };
            Random rnd = new Random();

            string returnString = medicalConditions[rnd.Next(0,
    ↪ medicalConditions.Length)];
            return returnString;
        }

        private void btnParGenMore_Click(object sender, EventArgs e)
        {
            generateAndPopulateMorePartners();
        }

        public void generateAndPopulateMorePartners()
        {
            //Now create family members
            for (int i = 0; i < 10; i++)
            {
                Partner tempFamily = new Partner();
                tempFamily.generateDetailedChar(mainCharacter.Gender,
    ↪ mainCharacter.Sexuality, controlClass.InGameDate);

                //add to the array
                potentialPartners[i] = tempFamily;
            }
            //now fill dgv. Code taken from dgvFamily

            //first have to clear it
            dgvPartPotential.Rows.Clear();
            dgvPartPotential.Refresh();


            //use code from openldbws app to populate dgvFamily

```

```
1088            //first, set var for which column contains which data
1089            int firstNameCol = 0;
1090            int lastNameCol = 1;
1091            int genderCol = 2;
1092            int dobCol = 3;
1093            int indexCol = 4;
1094
1095            //now loop through familyArray and populate dgvFamily
1096            // can use a for loop as number of family members are in the
    ↪ controlClass
1097
1098            for (int x = 0; x < 10; x++)
1099            {
1100
1101                int i = dgvPartPotential.Rows.Add();
1102                dgvPartPotential.Rows[i].Cells[firstNameCol].Value =
    ↪ potentialPartners[x].FirstName;
1103                dgvPartPotential.Rows[i].Cells[lastNameCol].Value =
    ↪ potentialPartners[x].LastName;
1104                dgvPartPotential.Rows[i].Cells[genderCol].Value =
    ↪ potentialPartners[x].Gender;
1105                dgvPartPotential.Rows[i].Cells[dobCol].Value =
    ↪ potentialPartners[x].DateOfBirth.ToShortDateString();
1106                dgvPartPotential.Rows[i].Cells[indexCol].Value = x;
1107
1108            }
1109        }
1110
1111        private void btnAskOnDate_Click(object sender, EventArgs e)
1112        {
1113            //basically the same as applying for a job
1114            try
1115            {
1116                int index = Int32.Parse(dgvPartPotential.SelectedCells[4].
    ↪ Value.ToString());
1117                partner = potentialPartners[index];
1118                MessageBox.Show(partner.FirstName);
1119                generateEvent("Relationships", "You asked " + partner.
    ↪ FirstName + " on a date and they said yes!", controlClass.InGameDate);
1120                partner.WhenMet = controlClass.InGameDate;
1121                mainCharacter.InRelationship = true;
1122            }
1123
1124            catch(Exception)
1125            {
1126                MessageBox.Show("Please select a partner before trying to go
    ↪  on a date with them.");
1127            }
1128            updatePartnerInformation();
1129        }
1130
1131        public void updatePartnerInformation()
1132        {
1133            if (mainCharacter.InRelationship == true)
1134            {
1135                lblPartFirstName.Text = "First Name: " + partner.FirstName;
1136                lblPartLastName.Text = "Last Name: " + partner.LastName;
1137                lblPartDateOfBirth.Text = "Date of Birth: " + partner.
    ↪ DateOfBirth.ToShortDateString();
1138                lblPartAge.Text = "Age: " + partner.Age.ToString();
1139            }
```

```
            else
            {
                lblPartFirstName.Text = "First Name: ";
                lblPartLastName.Text = "Last Name: ";
                lblPartDateOfBirth.Text = "Date of Birth: ";
                lblPartAge.Text = "Age: ";
            }
        }

        private void btnPartDump_Click(object sender, EventArgs e)
        {
            mainCharacter.InRelationship = false;
            partner.FirstName = null;
            partner.LastName = null;
            partner.Sexuality = null;
            partner.Gender = null;
            partner.LivingStatus = false;
            partner.Age = 0;
            updatePartnerInformation();
        }
    }
}
```

Listing A.9: FILE/frmMainGameScreen.cs

[FILE]frmMainGameScreen.Designer.cs

```

namespace A_level_Computer_Science_Project
{
    partial class frmMainGameScreen
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be
    ↪ disposed; otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.components = new System.ComponentModel.Container();
            System.Windows.Forms.DataGridViewCellStyle
    ↪ dataGridViewCellStyle1 = new System.Windows.Forms.
```

```
    ↪ DataGridViewCellStyle();
34          System.Windows.Forms.DataGridViewCellStyle
    ↪ dataGridViewCellStyle2 = new System.Windows.Forms.
    ↪ DataGridViewCellStyle();
35          System.Windows.Forms.DataGridViewCellStyle
    ↪ dataGridViewCellStyle3 = new System.Windows.Forms.
    ↪ DataGridViewCellStyle();
36          System.Windows.Forms.DataGridViewCellStyle
    ↪ dataGridViewCellStyle4 = new System.Windows.Forms.
    ↪ DataGridViewCellStyle();
37          System.Windows.Forms.DataGridViewCellStyle
    ↪ dataGridViewCellStyle5 = new System.Windows.Forms.
    ↪ DataGridViewCellStyle();
38          System.Windows.Forms.DataGridViewCellStyle
    ↪ dataGridViewCellStyle6 = new System.Windows.Forms.
    ↪ DataGridViewCellStyle();
39          System.Windows.Forms.DataGridViewCellStyle
    ↪ dataGridViewCellStyle7 = new System.Windows.Forms.
    ↪ DataGridViewCellStyle();
40          System.Windows.Forms.DataGridViewCellStyle
    ↪ dataGridViewCellStyle8 = new System.Windows.Forms.
    ↪ DataGridViewCellStyle();
41          System.Windows.Forms.DataGridViewCellStyle
    ↪ dataGridViewCellStyle9 = new System.Windows.Forms.
    ↪ DataGridViewCellStyle();
42          System.ComponentModel.ComponentResourceManager resources = new
    ↪ System.ComponentModel.ComponentResourceManager(typeof(
    ↪ frmMainGameScreen));
43          this.btnSaveToFile = new System.Windows.Forms.Button();
44          this.panel1 = new System.Windows.Forms.Panel();
45          this.lblInGameDate = new System.Windows.Forms.Label();
46          this.btnSaveAndHome = new System.Windows.Forms.Button();
47          this.pictureBox1 = new System.Windows.Forms.PictureBox();
48          this.panel2 = new System.Windows.Forms.Panel();
49          this.tabControl1 = new System.Windows.Forms.TabControl();
50          this.tabPage1 = new System.Windows.Forms.TabPage();
51          this.panel7 = new System.Windows.Forms.Panel();
52          this.btnAgeUp = new System.Windows.Forms.Button();
53          this.panel6 = new System.Windows.Forms.Panel();
54          this.lblMPDateOfBirth = new System.Windows.Forms.Label();
55          this.lblMPAge = new System.Windows.Forms.Label();
56          this.lblMPSexuality = new System.Windows.Forms.Label();
57          this.label2 = new System.Windows.Forms.Label();
58          this.lblMPGender = new System.Windows.Forms.Label();
59          this.lblMPLastName = new System.Windows.Forms.Label();
60          this.lblMPFirstName = new System.Windows.Forms.Label();
61          this.panel5 = new System.Windows.Forms.Panel();
62          this.label18 = new System.Windows.Forms.Label();
63          this.prbLifeScore = new System.Windows.Forms.ProgressBar();
64          this.prbMedicalScore = new System.Windows.Forms.ProgressBar();
65          this.prbHappinessScore = new System.Windows.Forms.ProgressBar();
66          this.prbJobScore = new System.Windows.Forms.ProgressBar();
67          this.label17 = new System.Windows.Forms.Label();
68          this.label16 = new System.Windows.Forms.Label();
69          this.label15 = new System.Windows.Forms.Label();
70          this.label14 = new System.Windows.Forms.Label();
71          this.prbEducationScore = new System.Windows.Forms.ProgressBar();
72          this.panel3 = new System.Windows.Forms.Panel();
73          this.txtEventBox = new System.Windows.Forms.TextBox();
74          this.label1 = new System.Windows.Forms.Label();
75          this.tabPage2 = new System.Windows.Forms.TabPage();
```

```
76              this.panel12 = new System.Windows.Forms.Panel();
77              this.panel13 = new System.Windows.Forms.Panel();
78              this.lblFamRealtionshipToMain = new System.Windows.Forms.Label()
    ↪ ;
79              this.lblFamLivingStatus = new System.Windows.Forms.Label();
80              this.lblFamReasonForDeath = new System.Windows.Forms.Label();
81              this.lblFamDateOfDeath = new System.Windows.Forms.Label();
82              this.lblFamInRelationship = new System.Windows.Forms.Label();
83              this.lblFamDateOfBirth = new System.Windows.Forms.Label();
84              this.lblFamAge = new System.Windows.Forms.Label();
85              this.lblFamSexuality = new System.Windows.Forms.Label();
86              this.lblFamGender = new System.Windows.Forms.Label();
87              this.lblFamLastName = new System.Windows.Forms.Label();
88              this.lblFamFirstName = new System.Windows.Forms.Label();
89              this.panel4 = new System.Windows.Forms.Panel();
90              this.dgvFamily = new System.Windows.Forms.DataGridView();
91              this.relationshipToMain = new System.Windows.Forms.
    ↪ DataGridViewTextBoxColumn();
92              this.firstName = new System.Windows.Forms.
    ↪ DataGridViewTextBoxColumn();
93              this.lastName = new System.Windows.Forms.
    ↪ DataGridViewTextBoxColumn();
94              this.tabPage3 = new System.Windows.Forms.TabPage();
95              this.panel17 = new System.Windows.Forms.Panel();
96              this.txtEduEducationEvents = new System.Windows.Forms.TextBox();
97              this.label20 = new System.Windows.Forms.Label();
98              this.panel14 = new System.Windows.Forms.Panel();
99              this.btnEduWithdraw = new System.Windows.Forms.Button();
100             this.panel15 = new System.Windows.Forms.Panel();
101             this.lblEduCurrentYear = new System.Windows.Forms.Label();
102             this.lblEduEducationSituation = new System.Windows.Forms.Label()
    ↪ ;
103             this.lblEduCurrentEducationPoints = new System.Windows.Forms.
    ↪ Label();
104             this.panel16 = new System.Windows.Forms.Panel();
105             this.tabPage5 = new System.Windows.Forms.TabPage();
106             this.panel18 = new System.Windows.Forms.Panel();
107             this.txtCriCrimeEvents = new System.Windows.Forms.TextBox();
108             this.label19 = new System.Windows.Forms.Label();
109             this.panel19 = new System.Windows.Forms.Panel();
110             this.btnCriCommitCrime = new System.Windows.Forms.Button();
111             this.panel21 = new System.Windows.Forms.Panel();
112             this.tabPage6 = new System.Windows.Forms.TabPage();
113             this.panel25 = new System.Windows.Forms.Panel();
114             this.panel23 = new System.Windows.Forms.Panel();
115             this.btnEditSaveChanges = new System.Windows.Forms.Button();
116             this.panel24 = new System.Windows.Forms.Panel();
117             this.panel22 = new System.Windows.Forms.Panel();
118             this.btnEndLife = new System.Windows.Forms.Button();
119             this.label11 = new System.Windows.Forms.Label();
120             this.label10 = new System.Windows.Forms.Label();
121             this.panel20 = new System.Windows.Forms.Panel();
122             this.label9 = new System.Windows.Forms.Label();
123             this.cboEditSexuality = new System.Windows.Forms.ComboBox();
124             this.cboEditEyeColour = new System.Windows.Forms.ComboBox();
125             this.label6 = new System.Windows.Forms.Label();
126             this.txtEditFirstName = new System.Windows.Forms.TextBox();
127             this.cboEditHairColour = new System.Windows.Forms.ComboBox();
128             this.label8 = new System.Windows.Forms.Label();
129             this.label5 = new System.Windows.Forms.Label();
130             this.txtEditLastName = new System.Windows.Forms.TextBox();
```

```
131             this.label4 = new System.Windows.Forms.Label();
132             this.label7 = new System.Windows.Forms.Label();
133             this.label3 = new System.Windows.Forms.Label();
134             this.cboEditGender = new System.Windows.Forms.ComboBox();
135             this.tabPage7 = new System.Windows.Forms.TabPage();
136             this.panel10 = new System.Windows.Forms.Panel();
137             this.label21 = new System.Windows.Forms.Label();
138             this.btnJobApplyForJob = new System.Windows.Forms.Button();
139             this.label13 = new System.Windows.Forms.Label();
140             this.btnJobReshuffleJobBoard = new System.Windows.Forms.Button()
    ↪ ;
141             this.panel8 = new System.Windows.Forms.Panel();
142             this.btnJobQuitJob = new System.Windows.Forms.Button();
143             this.panel9 = new System.Windows.Forms.Panel();
144             this.lblJobCurrentJobDuration = new System.Windows.Forms.Label()
    ↪ ;
145             this.lblJobCurrentJobBand = new System.Windows.Forms.Label();
146             this.lblJobCurrentJobSalary = new System.Windows.Forms.Label();
147             this.lblJobCurrentJobTitle = new System.Windows.Forms.Label();
148             this.label12 = new System.Windows.Forms.Label();
149             this.panel11 = new System.Windows.Forms.Panel();
150             this.dgvEmpAvailableJobs = new System.Windows.Forms.DataGridView
    ↪ ();
151             this.dataGridViewTextBoxColumn1 = new System.Windows.Forms.
    ↪ DataGridViewTextBoxColumn();
152             this.dataGridViewTextBoxColumn2 = new System.Windows.Forms.
    ↪ DataGridViewTextBoxColumn();
153             this.Column1 = new System.Windows.Forms.
    ↪ DataGridViewTextBoxColumn();
154             this.tabPage8 = new System.Windows.Forms.TabPage();
155             this.panel29 = new System.Windows.Forms.Panel();
156             this.panel26 = new System.Windows.Forms.Panel();
157             this.btnAskOnDate = new System.Windows.Forms.Button();
158             this.btnParGenMore = new System.Windows.Forms.Button();
159             this.panel27 = new System.Windows.Forms.Panel();
160             this.lblPartAge = new System.Windows.Forms.Label();
161             this.lblPartDateOfBirth = new System.Windows.Forms.Label();
162             this.lblPartLastName = new System.Windows.Forms.Label();
163             this.lblPartFirstName = new System.Windows.Forms.Label();
164             this.panel28 = new System.Windows.Forms.Panel();
165             this.dgvPartPotential = new System.Windows.Forms.DataGridView();
166             this.dataGridViewTextBoxColumn3 = new System.Windows.Forms.
    ↪ DataGridViewTextBoxColumn();
167             this.dataGridViewTextBoxColumn4 = new System.Windows.Forms.
    ↪ DataGridViewTextBoxColumn();
168             this.dataGridViewTextBoxColumn5 = new System.Windows.Forms.
    ↪ DataGridViewTextBoxColumn();
169             this.DateOfBirth = new System.Windows.Forms.
    ↪ DataGridViewTextBoxColumn();
170             this.index = new System.Windows.Forms.DataGridViewTextBoxColumn
    ↪ ();
171             this.ttMainToolTip = new System.Windows.Forms.ToolTip(this.
    ↪ components);
172             this.btnPartDump = new System.Windows.Forms.Button();
173             this.panel1.SuspendLayout();
174             ((System.ComponentModel.ISupportInitialize)(this.pictureBox1)).
    ↪ BeginInit();
175             this.panel2.SuspendLayout();
176             this.tabControl1.SuspendLayout();
177             this.tabPage1.SuspendLayout();
178             this.panel7.SuspendLayout();
```

```
179             this.panel6.SuspendLayout();
180             this.panel5.SuspendLayout();
181             this.panel3.SuspendLayout();
182             this.tabPage2.SuspendLayout();
183             this.panel13.SuspendLayout();
184             this.panel4.SuspendLayout();
185             ((System.ComponentModel.ISupportInitialize)(this.dgvFamily)).
    ↪ BeginInit();
186             this.tabPage3.SuspendLayout();
187             this.panel17.SuspendLayout();
188             this.panel14.SuspendLayout();
189             this.panel15.SuspendLayout();
190             this.tabPage5.SuspendLayout();
191             this.panel18.SuspendLayout();
192             this.panel19.SuspendLayout();
193             this.tabPage6.SuspendLayout();
194             this.panel23.SuspendLayout();
195             this.panel22.SuspendLayout();
196             this.panel20.SuspendLayout();
197             this.tabPage7.SuspendLayout();
198             this.panel10.SuspendLayout();
199             this.panel8.SuspendLayout();
200             this.panel9.SuspendLayout();
201             this.panel11.SuspendLayout();
202             ((System.ComponentModel.ISupportInitialize)(this.
    ↪ dgvEmpAvailableJobs)).BeginInit();
203             this.tabPage8.SuspendLayout();
204             this.panel26.SuspendLayout();
205             this.panel27.SuspendLayout();
206             this.panel28.SuspendLayout();
207             ((System.ComponentModel.ISupportInitialize)(this.
    ↪ dgvPartPotential)).BeginInit();
208             this.SuspendLayout();
209             //
210             // btnSaveToFile
211             //
212             this.btnSaveToFile.Location = new System.Drawing.Point(163, 0);
213             this.btnSaveToFile.Name = "btnSaveToFile";
214             this.btnSaveToFile.Size = new System.Drawing.Size(12, 26);
215             this.btnSaveToFile.TabIndex = 1;
216             this.btnSaveToFile.UseVisualStyleBackColor = true;
217             this.btnSaveToFile.Visible = false;
218             this.btnSaveToFile.Click += new System.EventHandler(this.
    ↪ btnSaveToFile_Click);
219             //
220             // panel1
221             //
222             this.panel1.BackColor = System.Drawing.Color.FromArgb(((int)(((
    ↪ byte)(31)))), ((int)(((byte)(31)))), ((int)(((byte)(31)))));
223             this.panel1.Controls.Add(this.lblInGameDate);
224             this.panel1.Controls.Add(this.btnSaveAndHome);
225             this.panel1.Controls.Add(this.pictureBox1);
226             this.panel1.Controls.Add(this.btnSaveToFile);
227             this.panel1.Location = new System.Drawing.Point(12, 12);
228             this.panel1.Name = "panel1";
229             this.panel1.Size = new System.Drawing.Size(776, 86);
230             this.panel1.TabIndex = 2;
231             //
232             // lblInGameDate
233             //
234             this.lblInGameDate.AutoSize = true;
```

```
235             this.lblInGameDate.ForeColor = System.Drawing.Color.White;
236             this.lblInGameDate.Location = new System.Drawing.Point(7, 27);
237             this.lblInGameDate.Name = "lblInGameDate";
238             this.lblInGameDate.Size = new System.Drawing.Size(84, 15);
239             this.lblInGameDate.TabIndex = 4;
240             this.lblInGameDate.Text = "In Game Date: ";
241             //
242             // btnSaveAndHome
243             //
244             this.btnSaveAndHome.BackColor = System.Drawing.Color.FromArgb(((
    ↪ int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
245             this.btnSaveAndHome.BackgroundImage = global::
    ↪ A_level_Computer_Science_Project.Properties.Resources.
    ↪ Save_and_home_icon;
246             this.btnSaveAndHome.BackgroundImageLayout = System.Windows.Forms
    ↪ .ImageLayout.Zoom;
247             this.btnSaveAndHome.FlatStyle = System.Windows.Forms.FlatStyle.
    ↪ Popup;
248             this.btnSaveAndHome.Location = new System.Drawing.Point(713, 2);
249             this.btnSaveAndHome.Name = "btnSaveAndHome";
250             this.btnSaveAndHome.Size = new System.Drawing.Size(63, 84);
251             this.btnSaveAndHome.TabIndex = 3;
252             this.btnSaveAndHome.UseVisualStyleBackColor = false;
253             this.btnSaveAndHome.Click += new System.EventHandler(this.
    ↪ btnSaveAndHome_Click);
254             //
255             // pictureBox1
256             //
257             this.pictureBox1.BackgroundImage = global::
    ↪ A_level_Computer_Science_Project.Properties.Resources.Logo_cropped;
258             this.pictureBox1.BackgroundImageLayout = System.Windows.Forms.
    ↪ ImageLayout.Zoom;
259             this.pictureBox1.Location = new System.Drawing.Point(181, 0);
260             this.pictureBox1.Name = "pictureBox1";
261             this.pictureBox1.Size = new System.Drawing.Size(423, 86);
262             this.pictureBox1.TabIndex = 2;
263             this.pictureBox1.TabStop = false;
264             //
265             // panel2
266             //
267             this.panel2.BackColor = System.Drawing.Color.FromArgb(((int)(((
    ↪ byte)(31)))), ((int)(((byte)(31)))), ((int)(((byte)(31)))));
268             this.panel2.Controls.Add(this.tabControl1);
269             this.panel2.Location = new System.Drawing.Point(12, 104);
270             this.panel2.Name = "panel2";
271             this.panel2.Size = new System.Drawing.Size(776, 334);
272             this.panel2.TabIndex = 3;
273             //
274             // tabControl1
275             //
276             this.tabControl1.Controls.Add(this.tabPage1);
277             this.tabControl1.Controls.Add(this.tabPage2);
278             this.tabControl1.Controls.Add(this.tabPage3);
279             this.tabControl1.Controls.Add(this.tabPage5);
280             this.tabControl1.Controls.Add(this.tabPage6);
281             this.tabControl1.Controls.Add(this.tabPage7);
282             this.tabControl1.Controls.Add(this.tabPage8);
283             this.tabControl1.Dock = System.Windows.Forms.DockStyle.Bottom;
284             this.tabControl1.Location = new System.Drawing.Point(0, 0);
285             this.tabControl1.Name = "tabControl1";
286             this.tabControl1.SelectedIndex = 0;
```

```
287             this.tabControl1.Size = new System.Drawing.Size(776, 334);
288             this.tabControl1.TabIndex = 0;
289             this.tabControl1.SelectedIndexChanged += new System.EventHandler
    ↪ (this.tabControl1_SelectedIndexChanged);
290             //
291             // tabPage1
292             //
293             this.tabPage1.BackColor = System.Drawing.Color.FromArgb(((int)
    ↪ (((byte)(31)))), ((int)(((byte)(31)))), ((int)(((byte)(31)))));
294             this.tabPage1.Controls.Add(this.panel7);
295             this.tabPage1.Controls.Add(this.panel6);
296             this.tabPage1.Controls.Add(this.panel5);
297             this.tabPage1.Controls.Add(this.panel3);
298             this.tabPage1.Location = new System.Drawing.Point(4, 24);
299             this.tabPage1.Name = "tabPage1";
300             this.tabPage1.Padding = new System.Windows.Forms.Padding(3);
301             this.tabPage1.Size = new System.Drawing.Size(768, 306);
302             this.tabPage1.TabIndex = 0;
303             this.tabPage1.Text = "Main page";
304             //
305             // panel7
306             //
307             this.panel7.Controls.Add(this.btnAgeUp);
308             this.panel7.Location = new System.Drawing.Point(536, 215);
309             this.panel7.Name = "panel7";
310             this.panel7.Size = new System.Drawing.Size(229, 88);
311             this.panel7.TabIndex = 4;
312             //
313             // btnAgeUp
314             //
315             this.btnAgeUp.BackColor = System.Drawing.Color.FromArgb(((int)
    ↪ (((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
316             this.btnAgeUp.FlatStyle = System.Windows.Forms.FlatStyle.Popup;
317             this.btnAgeUp.ForeColor = System.Drawing.Color.White;
318             this.btnAgeUp.Location = new System.Drawing.Point(0, 0);
319             this.btnAgeUp.Name = "btnAgeUp";
320             this.btnAgeUp.Size = new System.Drawing.Size(229, 88);
321             this.btnAgeUp.TabIndex = 0;
322             this.btnAgeUp.Text = "Age Up";
323             this.btnAgeUp.UseVisualStyleBackColor = false;
324             this.btnAgeUp.Click += new System.EventHandler(this.
    ↪ btnAgeUp_Click);
325             //
326             // panel6
327             //
328             this.panel6.Controls.Add(this.lblMPDateOfBirth);
329             this.panel6.Controls.Add(this.lblMPAge);
330             this.panel6.Controls.Add(this.lblMPSexuality);
331             this.panel6.Controls.Add(this.label2);
332             this.panel6.Controls.Add(this.lblMPGender);
333             this.panel6.Controls.Add(this.lblMPLastName);
334             this.panel6.Controls.Add(this.lblMPFirstName);
335             this.panel6.Location = new System.Drawing.Point(243, 3);
336             this.panel6.Name = "panel6";
337             this.panel6.Size = new System.Drawing.Size(522, 100);
338             this.panel6.TabIndex = 4;
339             //
340             // lblMPDateOfBirth
341             //
342             this.lblMPDateOfBirth.AutoSize = true;
343             this.lblMPDateOfBirth.ForeColor = System.Drawing.Color.White;
```

```
344             this.lblMPDateOfBirth.Location = new System.Drawing.Point(246,
    ↪ 30);
345             this.lblMPDateOfBirth.Name = "lblMPDateOfBirth";
346             this.lblMPDateOfBirth.Size = new System.Drawing.Size(81, 15);
347             this.lblMPDateOfBirth.TabIndex = 6;
348             this.lblMPDateOfBirth.Text = "Date Of Birth: ";
349             //
350             // lblMPAge
351             //
352             this.lblMPAge.AutoSize = true;
353             this.lblMPAge.ForeColor = System.Drawing.Color.White;
354             this.lblMPAge.Location = new System.Drawing.Point(246, 15);
355             this.lblMPAge.Name = "lblMPAge";
356             this.lblMPAge.Size = new System.Drawing.Size(34, 15);
357             this.lblMPAge.TabIndex = 5;
358             this.lblMPAge.Text = "Age: ";
359             //
360             // lblMPSexuality
361             //
362             this.lblMPSexuality.AutoSize = true;
363             this.lblMPSexuality.ForeColor = System.Drawing.Color.White;
364             this.lblMPSexuality.Location = new System.Drawing.Point(15, 58);
365             this.lblMPSexuality.Name = "lblMPSexuality";
366             this.lblMPSexuality.Size = new System.Drawing.Size(60, 15);
367             this.lblMPSexuality.TabIndex = 4;
368             this.lblMPSexuality.Text = "Sexuality: ";
369             //
370             // label2
371             //
372             this.label2.AutoSize = true;
373             this.label2.ForeColor = System.Drawing.Color.White;
374             this.label2.Location = new System.Drawing.Point(725, 482);
375             this.label2.Name = "label2";
376             this.label2.Size = new System.Drawing.Size(51, 15);
377             this.label2.TabIndex = 3;
378             this.label2.Text = "Gender: ";
379             //
380             // lblMPGender
381             //
382             this.lblMPGender.AutoSize = true;
383             this.lblMPGender.ForeColor = System.Drawing.Color.White;
384             this.lblMPGender.Location = new System.Drawing.Point(15, 43);
385             this.lblMPGender.Name = "lblMPGender";
386             this.lblMPGender.Size = new System.Drawing.Size(51, 15);
387             this.lblMPGender.TabIndex = 2;
388             this.lblMPGender.Text = "Gender: ";
389             //
390             // lblMPLastName
391             //
392             this.lblMPLastName.AutoSize = true;
393             this.lblMPLastName.ForeColor = System.Drawing.Color.White;
394             this.lblMPLastName.Location = new System.Drawing.Point(15, 30);
395             this.lblMPLastName.Name = "lblMPLastName";
396             this.lblMPLastName.Size = new System.Drawing.Size(69, 15);
397             this.lblMPLastName.TabIndex = 1;
398             this.lblMPLastName.Text = "Last Name: ";
399             //
400             // lblMPFirstName
401             //
402             this.lblMPFirstName.AutoSize = true;
403             this.lblMPFirstName.ForeColor = System.Drawing.Color.White;
```

```
404             this.lblMPFirstName.Location = new System.Drawing.Point(15, 15);
405             this.lblMPFirstName.Name = "lblMPFirstName";
406             this.lblMPFirstName.Size = new System.Drawing.Size(70, 15);
407             this.lblMPFirstName.TabIndex = 0;
408             this.lblMPFirstName.Text = "First Name: ";
409             //
410             // panel5
411             //
412             this.panel5.Controls.Add(this.label18);
413             this.panel5.Controls.Add(this.prbLifeScore);
414             this.panel5.Controls.Add(this.prbMedicalScore);
415             this.panel5.Controls.Add(this.prbHappinessScore);
416             this.panel5.Controls.Add(this.prbJobScore);
417             this.panel5.Controls.Add(this.label17);
418             this.panel5.Controls.Add(this.label16);
419             this.panel5.Controls.Add(this.label15);
420             this.panel5.Controls.Add(this.label14);
421             this.panel5.Controls.Add(this.prbEducationScore);
422             this.panel5.Location = new System.Drawing.Point(243, 109);
423             this.panel5.Name = "panel5";
424             this.panel5.Size = new System.Drawing.Size(522, 100);
425             this.panel5.TabIndex = 3;
426             //
427             // label18
428             //
429             this.label18.AutoSize = true;
430             this.label18.ForeColor = System.Drawing.Color.White;
431             this.label18.Location = new System.Drawing.Point(395, 3);
432             this.label18.Name = "label18";
433             this.label18.Size = new System.Drawing.Size(58, 15);
434             this.label18.TabIndex = 17;
435             this.label18.Text = "Life Score";
436             //
437             // prbLifeScore
438             //
439             this.prbLifeScore.BackColor = System.Drawing.Color.FromArgb(((
    ↪ int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
440             this.prbLifeScore.ForeColor = System.Drawing.Color.FromArgb(((
    ↪ int)(((byte)(100)))), ((int)(((byte)(164)))), ((int)(((byte)(209)))));
441             this.prbLifeScore.Location = new System.Drawing.Point(395, 21);
442             this.prbLifeScore.Maximum = 1000;
443             this.prbLifeScore.Name = "prbLifeScore";
444             this.prbLifeScore.Size = new System.Drawing.Size(92, 23);
445             this.prbLifeScore.Style = System.Windows.Forms.ProgressBarStyle.
    ↪ Continuous;
446             this.prbLifeScore.TabIndex = 16;
447             //
448             // prbMedicalScore
449             //
450             this.prbMedicalScore.BackColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))
    ↪ ));
451             this.prbMedicalScore.ForeColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(100)))), ((int)(((byte)(164)))), ((int)(((byte)(209)))
    ↪ ));
452             this.prbMedicalScore.Location = new System.Drawing.Point(297,
    ↪ 21);
453             this.prbMedicalScore.Maximum = 1000;
454             this.prbMedicalScore.Name = "prbMedicalScore";
455             this.prbMedicalScore.Size = new System.Drawing.Size(92, 23);
456             this.prbMedicalScore.Style = System.Windows.Forms.
```

```
    ↪ ProgressBarStyle.Continuous;
457             this.prbMedicalScore.TabIndex = 15;
458             //
459             // prbHappinessScore
460             //
461             this.prbHappinessScore.BackColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))
    ↪ ));
462             this.prbHappinessScore.ForeColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(100)))), ((int)(((byte)(164)))), ((int)(((byte)(209)))
    ↪ ));
463             this.prbHappinessScore.Location = new System.Drawing.Point(199,
    ↪ 21);
464             this.prbHappinessScore.Maximum = 1000;
465             this.prbHappinessScore.Name = "prbHappinessScore";
466             this.prbHappinessScore.Size = new System.Drawing.Size(92, 23);
467             this.prbHappinessScore.Style = System.Windows.Forms.
    ↪ ProgressBarStyle.Continuous;
468             this.prbHappinessScore.TabIndex = 14;
469             //
470             // prbJobScore
471             //
472             this.prbJobScore.BackColor = System.Drawing.Color.FromArgb(((int
    ↪ )(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
473             this.prbJobScore.ForeColor = System.Drawing.Color.FromArgb(((int
    ↪ )(((byte)(100)))), ((int)(((byte)(164)))), ((int)(((byte)(209)))));
474             this.prbJobScore.Location = new System.Drawing.Point(101, 21);
475             this.prbJobScore.Maximum = 1000;
476             this.prbJobScore.Name = "prbJobScore";
477             this.prbJobScore.Size = new System.Drawing.Size(92, 23);
478             this.prbJobScore.Style = System.Windows.Forms.ProgressBarStyle.
    ↪ Continuous;
479             this.prbJobScore.TabIndex = 13;
480             //
481             // label17
482             //
483             this.label17.AutoSize = true;
484             this.label17.ForeColor = System.Drawing.Color.White;
485             this.label17.Location = new System.Drawing.Point(299, 3);
486             this.label17.Name = "label17";
487             this.label17.Size = new System.Drawing.Size(81, 15);
488             this.label17.TabIndex = 12;
489             this.label17.Text = "Medical Score";
490             //
491             // label16
492             //
493             this.label16.AutoSize = true;
494             this.label16.ForeColor = System.Drawing.Color.White;
495             this.label16.Location = new System.Drawing.Point(199, 3);
496             this.label16.Name = "label16";
497             this.label16.Size = new System.Drawing.Size(94, 15);
498             this.label16.TabIndex = 10;
499             this.label16.Text = "Happiness Score";
500             //
501             // label15
502             //
503             this.label15.AutoSize = true;
504             this.label15.ForeColor = System.Drawing.Color.White;
505             this.label15.Location = new System.Drawing.Point(116, 3);
506             this.label15.Name = "label15";
507             this.label15.Size = new System.Drawing.Size(57, 15);
```

```
508             this.label15.TabIndex = 8;
509             this.label15.Text = "Job Score";
510             //
511             // label14
512             //
513             this.label14.AutoSize = true;
514             this.label14.ForeColor = System.Drawing.Color.White;
515             this.label14.Location = new System.Drawing.Point(3, 3);
516             this.label14.Name = "label14";
517             this.label14.Size = new System.Drawing.Size(92, 15);
518             this.label14.TabIndex = 6;
519             this.label14.Text = "Education Score";
520             //
521             // prbEducationScore
522             //
523             this.prbEducationScore.BackColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))
    ↪ ));
524             this.prbEducationScore.ForeColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(100)))), ((int)(((byte)(164)))), ((int)(((byte)(209)))
    ↪ ));
525             this.prbEducationScore.Location = new System.Drawing.Point(3,
    ↪ 21);
526             this.prbEducationScore.Maximum = 1000;
527             this.prbEducationScore.Name = "prbEducationScore";
528             this.prbEducationScore.Size = new System.Drawing.Size(92, 23);
529             this.prbEducationScore.Style = System.Windows.Forms.
    ↪ ProgressBarStyle.Continuous;
530             this.prbEducationScore.TabIndex = 5;
531             //
532             // panel3
533             //
534             this.panel3.Controls.Add(this.txtEventBox);
535             this.panel3.Controls.Add(this.label1);
536             this.panel3.Location = new System.Drawing.Point(3, 3);
537             this.panel3.Name = "panel3";
538             this.panel3.Size = new System.Drawing.Size(234, 300);
539             this.panel3.TabIndex = 2;
540             //
541             // txtEventBox
542             //
543             this.txtEventBox.BackColor = System.Drawing.Color.FromArgb(((int
    ↪ )(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
544             this.txtEventBox.ForeColor = System.Drawing.Color.White;
545             this.txtEventBox.Location = new System.Drawing.Point(12, 43);
546             this.txtEventBox.Multiline = true;
547             this.txtEventBox.Name = "txtEventBox";
548             this.txtEventBox.ReadOnly = true;
549             this.txtEventBox.ScrollBars = System.Windows.Forms.ScrollBars.
    ↪ Vertical;
550             this.txtEventBox.Size = new System.Drawing.Size(200, 257);
551             this.txtEventBox.TabIndex = 0;
552             //
553             // label1
554             //
555             this.label1.AutoSize = true;
556             this.label1.Font = new System.Drawing.Font("Segoe UI", 14.25F,
    ↪ System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point);
557             this.label1.ForeColor = System.Drawing.Color.White;
558             this.label1.Location = new System.Drawing.Point(24, 15);
559             this.label1.Name = "label1";
```

```
560             this.label1.Size = new System.Drawing.Size(172, 25);
561             this.label1.TabIndex = 1;
562             this.label1.Text = "Whats Happening?";
563             //
564             // tabPage2
565             //
566             this.tabPage2.BackColor = System.Drawing.Color.FromArgb(((int)
    ↪ (((byte)(31)))), ((int)(((byte)(31)))), ((int)(((byte)(31)))));
567             this.tabPage2.Controls.Add(this.panel12);
568             this.tabPage2.Controls.Add(this.panel13);
569             this.tabPage2.Controls.Add(this.panel4);
570             this.tabPage2.Location = new System.Drawing.Point(4, 24);
571             this.tabPage2.Name = "tabPage2";
572             this.tabPage2.Padding = new System.Windows.Forms.Padding(3);
573             this.tabPage2.Size = new System.Drawing.Size(768, 306);
574             this.tabPage2.TabIndex = 1;
575             this.tabPage2.Text = "Family";
576             //
577             // panel12
578             //
579             this.panel12.Location = new System.Drawing.Point(536, 215);
580             this.panel12.Name = "panel12";
581             this.panel12.Size = new System.Drawing.Size(229, 88);
582             this.panel12.TabIndex = 7;
583             //
584             // panel13
585             //
586             this.panel13.Controls.Add(this.lblFamRealtionshipToMain);
587             this.panel13.Controls.Add(this.lblFamLivingStatus);
588             this.panel13.Controls.Add(this.lblFamReasonForDeath);
589             this.panel13.Controls.Add(this.lblFamDateOfDeath);
590             this.panel13.Controls.Add(this.lblFamInRelationship);
591             this.panel13.Controls.Add(this.lblFamDateOfBirth);
592             this.panel13.Controls.Add(this.lblFamAge);
593             this.panel13.Controls.Add(this.lblFamSexuality);
594             this.panel13.Controls.Add(this.lblFamGender);
595             this.panel13.Controls.Add(this.lblFamLastName);
596             this.panel13.Controls.Add(this.lblFamFirstName);
597             this.panel13.Location = new System.Drawing.Point(379, 3);
598             this.panel13.Name = "panel13";
599             this.panel13.Size = new System.Drawing.Size(386, 206);
600             this.panel13.TabIndex = 6;
601             //
602             // lblFamRealtionshipToMain
603             //
604             this.lblFamRealtionshipToMain.AutoSize = true;
605             this.lblFamRealtionshipToMain.Font = new System.Drawing.Font("
    ↪ Segoe UI", 9.75F, System.Drawing.FontStyle.Regular, System.Drawing.
    ↪ GraphicsUnit.Point);
606             this.lblFamRealtionshipToMain.ForeColor = System.Drawing.Color.
    ↪ White;
607             this.lblFamRealtionshipToMain.Location = new System.Drawing.
    ↪ Point(14, 11);
608             this.lblFamRealtionshipToMain.Name = "lblFamRealtionshipToMain";
609             this.lblFamRealtionshipToMain.Size = new System.Drawing.Size(0,
    ↪ 17);
610             this.lblFamRealtionshipToMain.TabIndex = 17;
611             //
612             // lblFamLivingStatus
613             //
614             this.lblFamLivingStatus.AutoSize = true;
```

```
615             this.lblFamLivingStatus.ForeColor = System.Drawing.Color.White;
616             this.lblFamLivingStatus.Location = new System.Drawing.Point(14,
    ↪ 101);
617             this.lblFamLivingStatus.Name = "lblFamLivingStatus";
618             this.lblFamLivingStatus.Size = new System.Drawing.Size(80, 15);
619             this.lblFamLivingStatus.TabIndex = 16;
620             this.lblFamLivingStatus.Text = "Living Status: ";
621             //
622             // lblFamReasonForDeath
623             //
624             this.lblFamReasonForDeath.AutoSize = true;
625             this.lblFamReasonForDeath.ForeColor = System.Drawing.Color.White
    ↪ ;
626             this.lblFamReasonForDeath.Location = new System.Drawing.Point
    ↪ (14, 177);
627             this.lblFamReasonForDeath.Name = "lblFamReasonForDeath";
628             this.lblFamReasonForDeath.Size = new System.Drawing.Size(105,
    ↪ 15);
629             this.lblFamReasonForDeath.TabIndex = 15;
630             this.lblFamReasonForDeath.Text = "Reason For Death: ";
631             //
632             // lblFamDateOfDeath
633             //
634             this.lblFamDateOfDeath.AutoSize = true;
635             this.lblFamDateOfDeath.ForeColor = System.Drawing.Color.White;
636             this.lblFamDateOfDeath.Location = new System.Drawing.Point(14,
    ↪ 162);
637             this.lblFamDateOfDeath.Name = "lblFamDateOfDeath";
638             this.lblFamDateOfDeath.Size = new System.Drawing.Size(87, 15);
639             this.lblFamDateOfDeath.TabIndex = 14;
640             this.lblFamDateOfDeath.Text = "Date Of Death: ";
641             //
642             // lblFamInRelationship
643             //
644             this.lblFamInRelationship.AutoSize = true;
645             this.lblFamInRelationship.ForeColor = System.Drawing.Color.White
    ↪ ;
646             this.lblFamInRelationship.Location = new System.Drawing.Point
    ↪ (14, 147);
647             this.lblFamInRelationship.Name = "lblFamInRelationship";
648             this.lblFamInRelationship.Size = new System.Drawing.Size(91, 15)
    ↪ ;
649             this.lblFamInRelationship.TabIndex = 13;
650             this.lblFamInRelationship.Text = "In Relationship: ";
651             //
652             // lblFamDateOfBirth
653             //
654             this.lblFamDateOfBirth.AutoSize = true;
655             this.lblFamDateOfBirth.ForeColor = System.Drawing.Color.White;
656             this.lblFamDateOfBirth.Location = new System.Drawing.Point(14,
    ↪ 132);
657             this.lblFamDateOfBirth.Name = "lblFamDateOfBirth";
658             this.lblFamDateOfBirth.Size = new System.Drawing.Size(81, 15);
659             this.lblFamDateOfBirth.TabIndex = 12;
660             this.lblFamDateOfBirth.Text = "Date Of Birth: ";
661             //
662             // lblFamAge
663             //
664             this.lblFamAge.AutoSize = true;
665             this.lblFamAge.ForeColor = System.Drawing.Color.White;
666             this.lblFamAge.Location = new System.Drawing.Point(14, 117);
```

```
667             this.lblFamAge.Name = "lblFamAge";
668             this.lblFamAge.Size = new System.Drawing.Size(34, 15);
669             this.lblFamAge.TabIndex = 11;
670             this.lblFamAge.Text = "Age: ";
671             //
672             // lblFamSexuality
673             //
674             this.lblFamSexuality.AutoSize = true;
675             this.lblFamSexuality.ForeColor = System.Drawing.Color.White;
676             this.lblFamSexuality.Location = new System.Drawing.Point(14, 86)
    ↪ ;
677             this.lblFamSexuality.Name = "lblFamSexuality";
678             this.lblFamSexuality.Size = new System.Drawing.Size(60, 15);
679             this.lblFamSexuality.TabIndex = 10;
680             this.lblFamSexuality.Text = "Sexuality: ";
681             //
682             // lblFamGender
683             //
684             this.lblFamGender.AutoSize = true;
685             this.lblFamGender.ForeColor = System.Drawing.Color.White;
686             this.lblFamGender.Location = new System.Drawing.Point(14, 71);
687             this.lblFamGender.Name = "lblFamGender";
688             this.lblFamGender.Size = new System.Drawing.Size(51, 15);
689             this.lblFamGender.TabIndex = 9;
690             this.lblFamGender.Text = "Gender: ";
691             //
692             // lblFamLastName
693             //
694             this.lblFamLastName.AutoSize = true;
695             this.lblFamLastName.ForeColor = System.Drawing.Color.White;
696             this.lblFamLastName.Location = new System.Drawing.Point(14, 58);
697             this.lblFamLastName.Name = "lblFamLastName";
698             this.lblFamLastName.Size = new System.Drawing.Size(69, 15);
699             this.lblFamLastName.TabIndex = 8;
700             this.lblFamLastName.Text = "Last Name: ";
701             //
702             // lblFamFirstName
703             //
704             this.lblFamFirstName.AutoSize = true;
705             this.lblFamFirstName.ForeColor = System.Drawing.Color.White;
706             this.lblFamFirstName.Location = new System.Drawing.Point(14, 43)
    ↪ ;
707             this.lblFamFirstName.Name = "lblFamFirstName";
708             this.lblFamFirstName.Size = new System.Drawing.Size(70, 15);
709             this.lblFamFirstName.TabIndex = 7;
710             this.lblFamFirstName.Text = "First Name: ";
711             //
712             // panel4
713             //
714             this.panel4.Controls.Add(this.dgvFamily);
715             this.panel4.Location = new System.Drawing.Point(3, 3);
716             this.panel4.Name = "panel4";
717             this.panel4.Size = new System.Drawing.Size(370, 300);
718             this.panel4.TabIndex = 3;
719             //
720             // dgvFamily
721             //
722             this.dgvFamily.AllowUserToAddRows = false;
723             this.dgvFamily.AllowUserToDeleteRows = false;
724             this.dgvFamily.AllowUserToResizeColumns = false;
725             this.dgvFamily.AllowUserToResizeRows = false;
```

```
726             dataGridViewCellStyle1.BackColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))
    ↪ ));
727             this.dgvFamily.AlternatingRowsDefaultCellStyle =
    ↪ dataGridViewCellStyle1;
728             this.dgvFamily.BackgroundColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(31)))), ((int)(((byte)(31)))), ((int)(((byte)(31)))));
729             this.dgvFamily.CellBorderStyle = System.Windows.Forms.
    ↪ DataGridViewCellBorderStyle.SingleHorizontal;
730             dataGridViewCellStyle2.Alignment = System.Windows.Forms.
    ↪ DataGridViewContentAlignment.MiddleCenter;
731             dataGridViewCellStyle2.BackColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(100)))), ((int)(((byte)(164)))), ((int)(((byte)(209)))
    ↪ ));
732             dataGridViewCellStyle2.Font = new System.Drawing.Font("Segoe UI"
    ↪ , 9.75F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.
    ↪ Point);
733             dataGridViewCellStyle2.ForeColor = System.Drawing.Color.White;
734             dataGridViewCellStyle2.SelectionBackColor = System.Drawing.Color
    ↪ .FromArgb(((int)(((byte)(100)))), ((int)(((byte)(164)))), ((int)(((
    ↪ byte)(209)))));
735             dataGridViewCellStyle2.SelectionForeColor = System.Drawing.
    ↪ SystemColors.HighlightText;
736             dataGridViewCellStyle2.WrapMode = System.Windows.Forms.
    ↪ DataGridViewTriState.True;
737             this.dgvFamily.ColumnHeadersDefaultCellStyle =
    ↪ dataGridViewCellStyle2;
738             this.dgvFamily.ColumnHeadersHeightSizeMode = System.Windows.
    ↪ Forms.DataGridViewColumnHeadersHeightSizeMode.AutoSize;
739             this.dgvFamily.Columns.AddRange(new System.Windows.Forms.
    ↪ DataGridViewColumn[] {
740             this.relationshipToMain,
741             this.firstName,
742             this.lastName});
743             dataGridViewCellStyle3.Alignment = System.Windows.Forms.
    ↪ DataGridViewContentAlignment.MiddleLeft;
744             dataGridViewCellStyle3.BackColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))
    ↪ ));
745             dataGridViewCellStyle3.Font = new System.Drawing.Font("Segoe UI"
    ↪ , 9F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.
    ↪ Point);
746             dataGridViewCellStyle3.ForeColor = System.Drawing.Color.White;
747             dataGridViewCellStyle3.SelectionBackColor = System.Drawing.Color
    ↪ .FromArgb(((int)(((byte)(167)))), ((int)(((byte)(108)))), ((int)(((
    ↪ byte)(179)))));
748             dataGridViewCellStyle3.SelectionForeColor = System.Drawing.
    ↪ SystemColors.HighlightText;
749             dataGridViewCellStyle3.WrapMode = System.Windows.Forms.
    ↪ DataGridViewTriState.True;
750             this.dgvFamily.DefaultCellStyle = dataGridViewCellStyle3;
751             this.dgvFamily.Dock = System.Windows.Forms.DockStyle.Fill;
752             this.dgvFamily.GridColor = System.Drawing.Color.White;
753             this.dgvFamily.Location = new System.Drawing.Point(0, 0);
754             this.dgvFamily.Name = "dgvFamily";
755             this.dgvFamily.ReadOnly = true;
756             this.dgvFamily.RowHeadersVisible = false;
757             this.dgvFamily.RowTemplate.Height = 25;
758             this.dgvFamily.SelectionMode = System.Windows.Forms.
    ↪ DataGridViewSelectionMode.FullRowSelect;
759             this.dgvFamily.Size = new System.Drawing.Size(370, 300);
```

```
760             this.dgvFamily.TabIndex = 0;
761             this.dgvFamily.CellContentClick += new System.Windows.Forms.
    ↪ DataGridViewCellEventHandler(this.dgvFamily_CellContentClick);
762             //
763             // relationshipToMain
764             //
765             this.relationshipToMain.HeaderText = "Relationship";
766             this.relationshipToMain.Name = "relationshipToMain";
767             this.relationshipToMain.ReadOnly = true;
768             this.relationshipToMain.SortMode = System.Windows.Forms.
    ↪ DataGridViewColumnSortMode.NotSortable;
769             //
770             // firstName
771             //
772             this.firstName.HeaderText = "First Name";
773             this.firstName.Name = "firstName";
774             this.firstName.ReadOnly = true;
775             this.firstName.SortMode = System.Windows.Forms.
    ↪ DataGridViewColumnSortMode.NotSortable;
776             //
777             // lastName
778             //
779             this.lastName.HeaderText = "Last Name";
780             this.lastName.Name = "lastName";
781             this.lastName.ReadOnly = true;
782             this.lastName.SortMode = System.Windows.Forms.
    ↪ DataGridViewColumnSortMode.NotSortable;
783             //
784             // tabPage3
785             //
786             this.tabPage3.BackColor = System.Drawing.Color.FromArgb(((int)
    ↪ (((byte)(31)))), ((int)(((byte)(31)))), ((int)(((byte)(31)))));
787             this.tabPage3.Controls.Add(this.panel17);
788             this.tabPage3.Controls.Add(this.panel14);
789             this.tabPage3.Controls.Add(this.panel15);
790             this.tabPage3.Controls.Add(this.panel16);
791             this.tabPage3.Location = new System.Drawing.Point(4, 24);
792             this.tabPage3.Name = "tabPage3";
793             this.tabPage3.Size = new System.Drawing.Size(768, 306);
794             this.tabPage3.TabIndex = 4;
795             this.tabPage3.Text = "Education";
796             //
797             // panel17
798             //
799             this.panel17.Controls.Add(this.txtEduEducationEvents);
800             this.panel17.Controls.Add(this.label20);
801             this.panel17.Location = new System.Drawing.Point(3, 2);
802             this.panel17.Name = "panel17";
803             this.panel17.Size = new System.Drawing.Size(370, 300);
804             this.panel17.TabIndex = 11;
805             //
806             // txtEduEducationEvents
807             //
808             this.txtEduEducationEvents.BackColor = System.Drawing.Color.
    ↪ FromArgb(((int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte
    ↪ )(102)))));
809             this.txtEduEducationEvents.ForeColor = System.Drawing.Color.
    ↪ White;
810             this.txtEduEducationEvents.Location = new System.Drawing.Point
    ↪ (3, 44);
811             this.txtEduEducationEvents.Multiline = true;
```

```
812             this.txtEduEducationEvents.Name = "txtEduEducationEvents";
813             this.txtEduEducationEvents.ReadOnly = true;
814             this.txtEduEducationEvents.ScrollBars = System.Windows.Forms.
    ↪ ScrollBars.Vertical;
815             this.txtEduEducationEvents.Size = new System.Drawing.Size(364,
    ↪ 253);
816             this.txtEduEducationEvents.TabIndex = 1;
817             //
818             // label20
819             //
820             this.label20.AutoSize = true;
821             this.label20.Font = new System.Drawing.Font("Segoe UI", 15.75F,
    ↪ System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point);
822             this.label20.ForeColor = System.Drawing.Color.White;
823             this.label20.Location = new System.Drawing.Point(84, 15);
824             this.label20.Name = "label20";
825             this.label20.Size = new System.Drawing.Size(171, 30);
826             this.label20.TabIndex = 0;
827             this.label20.Text = "Education events";
828             //
829             // panel14
830             //
831             this.panel14.Controls.Add(this.btnEduWithdraw);
832             this.panel14.Location = new System.Drawing.Point(536, 214);
833             this.panel14.Name = "panel14";
834             this.panel14.Size = new System.Drawing.Size(229, 88);
835             this.panel14.TabIndex = 10;
836             //
837             // btnEduWithdraw
838             //
839             this.btnEduWithdraw.BackColor = System.Drawing.Color.FromArgb(((
    ↪ int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
840             this.btnEduWithdraw.FlatStyle = System.Windows.Forms.FlatStyle.
    ↪ Popup;
841             this.btnEduWithdraw.ForeColor = System.Drawing.Color.White;
842             this.btnEduWithdraw.Location = new System.Drawing.Point(142, 0);
843             this.btnEduWithdraw.Name = "btnEduWithdraw";
844             this.btnEduWithdraw.Size = new System.Drawing.Size(84, 85);
845             this.btnEduWithdraw.TabIndex = 0;
846             this.btnEduWithdraw.Text = "Permanently Withdraw from Education"
    ↪ ;
847             this.btnEduWithdraw.UseVisualStyleBackColor = false;
848             this.btnEduWithdraw.Click += new System.EventHandler(this.
    ↪ btnEduWithdraw_Click);
849             //
850             // panel15
851             //
852             this.panel15.Controls.Add(this.lblEduCurrentYear);
853             this.panel15.Controls.Add(this.lblEduEducationSituation);
854             this.panel15.Controls.Add(this.lblEduCurrentEducationPoints);
855             this.panel15.Location = new System.Drawing.Point(379, 2);
856             this.panel15.Name = "panel15";
857             this.panel15.Size = new System.Drawing.Size(386, 100);
858             this.panel15.TabIndex = 9;
859             //
860             // lblEduCurrentYear
861             //
862             this.lblEduCurrentYear.AutoSize = true;
863             this.lblEduCurrentYear.ForeColor = System.Drawing.Color.White;
864             this.lblEduCurrentYear.Location = new System.Drawing.Point(16,
    ↪ 42);
```

```
865             this.lblEduCurrentYear.Name = "lblEduCurrentYear";
866             this.lblEduCurrentYear.Size = new System.Drawing.Size(75, 15);
867             this.lblEduCurrentYear.TabIndex = 2;
868             this.lblEduCurrentYear.Text = "Current Year:";
869             //
870             // lblEduEducationSituation
871             //
872             this.lblEduEducationSituation.AutoSize = true;
873             this.lblEduEducationSituation.ForeColor = System.Drawing.Color.
    ↪ White;
874             this.lblEduEducationSituation.Location = new System.Drawing.
    ↪ Point(16, 27);
875             this.lblEduEducationSituation.Name = "lblEduEducationSituation";
876             this.lblEduEducationSituation.Size = new System.Drawing.Size
    ↪ (159, 15);
877             this.lblEduEducationSituation.TabIndex = 1;
878             this.lblEduEducationSituation.Text = "Current Education
    ↪ Situation: ";
879             //
880             // lblEduCurrentEducationPoints
881             //
882             this.lblEduCurrentEducationPoints.AutoSize = true;
883             this.lblEduCurrentEducationPoints.ForeColor = System.Drawing.
    ↪ Color.White;
884             this.lblEduCurrentEducationPoints.Location = new System.Drawing.
    ↪ Point(16, 12);
885             this.lblEduCurrentEducationPoints.Name = "
    ↪ lblEduCurrentEducationPoints";
886             this.lblEduCurrentEducationPoints.Size = new System.Drawing.Size
    ↪ (145, 15);
887             this.lblEduCurrentEducationPoints.TabIndex = 0;
888             this.lblEduCurrentEducationPoints.Text = "Current Education
    ↪ Points: ";
889             //
890             // panel16
891             //
892             this.panel16.Location = new System.Drawing.Point(379, 108);
893             this.panel16.Name = "panel16";
894             this.panel16.Size = new System.Drawing.Size(386, 100);
895             this.panel16.TabIndex = 8;
896             //
897             // tabPage5
898             //
899             this.tabPage5.BackColor = System.Drawing.Color.FromArgb(((int)
    ↪ (((byte)(31)))), ((int)(((byte)(31)))), ((int)(((byte)(31)))));
900             this.tabPage5.Controls.Add(this.panel18);
901             this.tabPage5.Controls.Add(this.panel19);
902             this.tabPage5.Controls.Add(this.panel21);
903             this.tabPage5.Location = new System.Drawing.Point(4, 24);
904             this.tabPage5.Name = "tabPage5";
905             this.tabPage5.Padding = new System.Windows.Forms.Padding(3);
906             this.tabPage5.Size = new System.Drawing.Size(768, 306);
907             this.tabPage5.TabIndex = 5;
908             this.tabPage5.Text = "Crime";
909             //
910             // panel18
911             //
912             this.panel18.Controls.Add(this.txtCriCrimeEvents);
913             this.panel18.Controls.Add(this.label19);
914             this.panel18.Location = new System.Drawing.Point(3, 3);
915             this.panel18.Name = "panel18";
```

```
916             this.panel18.Size = new System.Drawing.Size(370, 300);
917             this.panel18.TabIndex = 15;
918             //
919             // txtCriCrimeEvents
920             //
921             this.txtCriCrimeEvents.BackColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))
    ↪ ));
922             this.txtCriCrimeEvents.ForeColor = System.Drawing.Color.White;
923             this.txtCriCrimeEvents.Location = new System.Drawing.Point(3,
    ↪ 44);
924             this.txtCriCrimeEvents.Multiline = true;
925             this.txtCriCrimeEvents.Name = "txtCriCrimeEvents";
926             this.txtCriCrimeEvents.ReadOnly = true;
927             this.txtCriCrimeEvents.ScrollBars = System.Windows.Forms.
    ↪ ScrollBars.Vertical;
928             this.txtCriCrimeEvents.Size = new System.Drawing.Size(364, 253);
929             this.txtCriCrimeEvents.TabIndex = 1;
930             //
931             // label19
932             //
933             this.label19.AutoSize = true;
934             this.label19.Font = new System.Drawing.Font("Segoe UI", 15.75F,
    ↪ System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point);
935             this.label19.ForeColor = System.Drawing.Color.White;
936             this.label19.Location = new System.Drawing.Point(84, 15);
937             this.label19.Name = "label19";
938             this.label19.Size = new System.Drawing.Size(133, 30);
939             this.label19.TabIndex = 0;
940             this.label19.Text = "Crime Events";
941             //
942             // panel19
943             //
944             this.panel19.Controls.Add(this.btnCriCommitCrime);
945             this.panel19.Location = new System.Drawing.Point(536, 215);
946             this.panel19.Name = "panel19";
947             this.panel19.Size = new System.Drawing.Size(229, 88);
948             this.panel19.TabIndex = 14;
949             //
950             // btnCriCommitCrime
951             //
952             this.btnCriCommitCrime.BackColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))
    ↪ ));
953             this.btnCriCommitCrime.FlatStyle = System.Windows.Forms.
    ↪ FlatStyle.Popup;
954             this.btnCriCommitCrime.ForeColor = System.Drawing.Color.White;
955             this.btnCriCommitCrime.Location = new System.Drawing.Point(142,
    ↪ 0);
956             this.btnCriCommitCrime.Name = "btnCriCommitCrime";
957             this.btnCriCommitCrime.Size = new System.Drawing.Size(84, 85);
958             this.btnCriCommitCrime.TabIndex = 0;
959             this.btnCriCommitCrime.Text = "Commit Random Crime";
960             this.btnCriCommitCrime.UseVisualStyleBackColor = false;
961             this.btnCriCommitCrime.Click += new System.EventHandler(this.
    ↪ btnCriCommitCrime_Click);
962             //
963             // panel21
964             //
965             this.panel21.Location = new System.Drawing.Point(379, 109);
966             this.panel21.Name = "panel21";
```

```
967             this.panel21.Size = new System.Drawing.Size(386, 100);
968             this.panel21.TabIndex = 12;
969             //
970             // tabPage6
971             //
972             this.tabPage6.BackColor = System.Drawing.Color.FromArgb(((int)
    ↪ (((byte)(31)))), ((int)(((byte)(31)))), ((int)(((byte)(31)))));
973             this.tabPage6.Controls.Add(this.panel25);
974             this.tabPage6.Controls.Add(this.panel23);
975             this.tabPage6.Controls.Add(this.panel22);
976             this.tabPage6.Controls.Add(this.panel20);
977             this.tabPage6.Location = new System.Drawing.Point(4, 24);
978             this.tabPage6.Name = "tabPage6";
979             this.tabPage6.Padding = new System.Windows.Forms.Padding(3);
980             this.tabPage6.Size = new System.Drawing.Size(768, 306);
981             this.tabPage6.TabIndex = 6;
982             this.tabPage6.Text = "Edit Character";
983             //
984             // panel25
985             //
986             this.panel25.Location = new System.Drawing.Point(433, 227);
987             this.panel25.Name = "panel25";
988             this.panel25.Size = new System.Drawing.Size(332, 76);
989             this.panel25.TabIndex = 2;
990             //
991             // panel23
992             //
993             this.panel23.Controls.Add(this.btnEditSaveChanges);
994             this.panel23.Controls.Add(this.panel24);
995             this.panel23.Location = new System.Drawing.Point(3, 227);
996             this.panel23.Name = "panel23";
997             this.panel23.Size = new System.Drawing.Size(424, 76);
998             this.panel23.TabIndex = 1;
999             //
1000            // btnEditSaveChanges
1001            //
1002            this.btnEditSaveChanges.BackColor = System.Drawing.Color.
    ↪ FromArgb(((int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte
    ↪ )(102)))));
1003            this.btnEditSaveChanges.BackgroundImage = global::
    ↪ A_level_Computer_Science_Project.Properties.Resources.save;
1004            this.btnEditSaveChanges.BackgroundImageLayout = System.Windows.
    ↪ Forms.ImageLayout.Zoom;
1005            this.btnEditSaveChanges.FlatStyle = System.Windows.Forms.
    ↪ FlatStyle.Flat;
1006            this.btnEditSaveChanges.Location = new System.Drawing.Point(0,
    ↪ 0);
1007            this.btnEditSaveChanges.Name = "btnEditSaveChanges";
1008            this.btnEditSaveChanges.Size = new System.Drawing.Size(81, 76);
1009            this.btnEditSaveChanges.TabIndex = 0;
1010            this.btnEditSaveChanges.UseVisualStyleBackColor = false;
1011            this.btnEditSaveChanges.Click += new System.EventHandler(this.
    ↪ btnEditSaveChanges_Click);
1012            //
1013            // panel24
1014            //
1015            this.panel24.Location = new System.Drawing.Point(430, 0);
1016            this.panel24.Name = "panel24";
1017            this.panel24.Size = new System.Drawing.Size(332, 76);
1018            this.panel24.TabIndex = 2;
1019            //
```

```
1020            // panel22
1021            //
1022            this.panel22.Controls.Add(this.btnEndLife);
1023            this.panel22.Controls.Add(this.label11);
1024            this.panel22.Controls.Add(this.label10);
1025            this.panel22.Location = new System.Drawing.Point(433, 3);
1026            this.panel22.Name = "panel22";
1027            this.panel22.Size = new System.Drawing.Size(332, 218);
1028            this.panel22.TabIndex = 1;
1029            //
1030            // btnEndLife
1031            //
1032            this.btnEndLife.BackColor = System.Drawing.Color.FromArgb(((int)
    ↪ (((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
1033            this.btnEndLife.FlatStyle = System.Windows.Forms.FlatStyle.Popup
    ↪ ;
1034            this.btnEndLife.ForeColor = System.Drawing.Color.White;
1035            this.btnEndLife.Location = new System.Drawing.Point(29, 114);
1036            this.btnEndLife.Name = "btnEndLife";
1037            this.btnEndLife.Size = new System.Drawing.Size(261, 39);
1038            this.btnEndLife.TabIndex = 30;
1039            this.btnEndLife.Text = "End life";
1040            this.btnEndLife.UseVisualStyleBackColor = false;
1041            this.btnEndLife.Click += new System.EventHandler(this.
    ↪ btnEndLife_Click);
1042            //
1043            // label11
1044            //
1045            this.label11.AutoSize = true;
1046            this.label11.ForeColor = System.Drawing.Color.White;
1047            this.label11.Location = new System.Drawing.Point(29, 50);
1048            this.label11.Name = "label11";
1049            this.label11.Size = new System.Drawing.Size(272, 45);
1050            this.label11.TabIndex = 29;
1051            this.label11.Text = "The button below will terminate the current
    ↪  life.\r\n(This is irreversible and will" +
1052     " result in you not being \r\nable to play this life again).";
1053            //
1054            // label10
1055            //
1056            this.label10.AutoSize = true;
1057            this.label10.Font = new System.Drawing.Font("Segoe UI", 14.25F,
    ↪ System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point);
1058            this.label10.ForeColor = System.Drawing.Color.White;
1059            this.label10.Location = new System.Drawing.Point(123, 11);
1060            this.label10.Name = "label10";
1061            this.label10.Size = new System.Drawing.Size(78, 25);
1062            this.label10.TabIndex = 28;
1063            this.label10.Text = "End life";
1064            this.label10.TextAlign = System.Drawing.ContentAlignment.
    ↪ TopCenter;
1065            this.ttMainToolTip.SetToolTip(this.label10, "Unlocks when Main
    ↪ Character turns 13");
1066            //
1067            // panel20
1068            //
1069            this.panel20.Controls.Add(this.label9);
1070            this.panel20.Controls.Add(this.cboEditSexuality);
1071            this.panel20.Controls.Add(this.cboEditEyeColour);
1072            this.panel20.Controls.Add(this.label6);
1073            this.panel20.Controls.Add(this.txtEditFirstName);
```

```
1074            this.panel20.Controls.Add(this.cboEditHairColour);
1075            this.panel20.Controls.Add(this.label8);
1076            this.panel20.Controls.Add(this.label5);
1077            this.panel20.Controls.Add(this.txtEditLastName);
1078            this.panel20.Controls.Add(this.label4);
1079            this.panel20.Controls.Add(this.label7);
1080            this.panel20.Controls.Add(this.label3);
1081            this.panel20.Controls.Add(this.cboEditGender);
1082            this.panel20.Location = new System.Drawing.Point(3, 3);
1083            this.panel20.Name = "panel20";
1084            this.panel20.Size = new System.Drawing.Size(424, 218);
1085            this.panel20.TabIndex = 0;
1086            //
1087            // label9
1088            //
1089            this.label9.AutoSize = true;
1090            this.label9.Font = new System.Drawing.Font("Segoe UI", 14.25F,
    ↪ System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point);
1091            this.label9.ForeColor = System.Drawing.Color.White;
1092            this.label9.Location = new System.Drawing.Point(18, 11);
1093            this.label9.Name = "label9";
1094            this.label9.Size = new System.Drawing.Size(243, 25);
1095            this.label9.TabIndex = 27;
1096            this.label9.Text = "Customise Main Character";
1097            this.label9.TextAlign = System.Drawing.ContentAlignment.
    ↪ TopCenter;
1098            this.ttMainToolTip.SetToolTip(this.label9, "Unlocks when Main
    ↪ Character turns 13");
1099            //
1100            // cboEditSexuality
1101            //
1102            this.cboEditSexuality.BackColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))
    ↪ ));
1103            this.cboEditSexuality.DropDownStyle = System.Windows.Forms.
    ↪ ComboBoxStyle.DropDownList;
1104            this.cboEditSexuality.Enabled = false;
1105            this.cboEditSexuality.ForeColor = System.Drawing.Color.White;
1106            this.cboEditSexuality.FormattingEnabled = true;
1107            this.cboEditSexuality.Items.AddRange(new object[] {
1108            "Straight",
1109            "Homosexual"});
1110            this.cboEditSexuality.Location = new System.Drawing.Point(93,
    ↪ 123);
1111            this.cboEditSexuality.Name = "cboEditSexuality";
1112            this.cboEditSexuality.Size = new System.Drawing.Size(121, 23);
1113            this.cboEditSexuality.TabIndex = 26;
1114            //
1115            // cboEditEyeColour
1116            //
1117            this.cboEditEyeColour.BackColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))
    ↪ ));
1118            this.cboEditEyeColour.DropDownStyle = System.Windows.Forms.
    ↪ ComboBoxStyle.DropDownList;
1119            this.cboEditEyeColour.Enabled = false;
1120            this.cboEditEyeColour.ForeColor = System.Drawing.Color.White;
1121            this.cboEditEyeColour.FormattingEnabled = true;
1122            this.cboEditEyeColour.Items.AddRange(new object[] {
1123            "Brown",
1124            "Blue",
```

```
1125            "Green",
1126            "Gray",
1127            "Amber"});
1128            this.cboEditEyeColour.Location = new System.Drawing.Point(93,
    ↪ 181);
1129            this.cboEditEyeColour.Name = "cboEditEyeColour";
1130            this.cboEditEyeColour.Size = new System.Drawing.Size(121, 23);
1131            this.cboEditEyeColour.TabIndex = 19;
1132            //
1133            // label6
1134            //
1135            this.label6.AutoSize = true;
1136            this.label6.ForeColor = System.Drawing.Color.White;
1137            this.label6.Location = new System.Drawing.Point(21, 184);
1138            this.label6.Name = "label6";
1139            this.label6.Size = new System.Drawing.Size(62, 15);
1140            this.label6.TabIndex = 25;
1141            this.label6.Text = "Eye colour";
1142            //
1143            // txtEditFirstName
1144            //
1145            this.txtEditFirstName.BackColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))
    ↪ ));
1146            this.txtEditFirstName.BorderStyle = System.Windows.Forms.
    ↪ BorderStyle.None;
1147            this.txtEditFirstName.Enabled = false;
1148            this.txtEditFirstName.ForeColor = System.Drawing.Color.White;
1149            this.txtEditFirstName.Location = new System.Drawing.Point(93,
    ↪ 50);
1150            this.txtEditFirstName.Name = "txtEditFirstName";
1151            this.txtEditFirstName.Size = new System.Drawing.Size(121, 16);
1152            this.txtEditFirstName.TabIndex = 14;
1153            //
1154            // cboEditHairColour
1155            //
1156            this.cboEditHairColour.BackColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))
    ↪ ));
1157            this.cboEditHairColour.DropDownStyle = System.Windows.Forms.
    ↪ ComboBoxStyle.DropDownList;
1158            this.cboEditHairColour.Enabled = false;
1159            this.cboEditHairColour.ForeColor = System.Drawing.Color.White;
1160            this.cboEditHairColour.FormattingEnabled = true;
1161            this.cboEditHairColour.Items.AddRange(new object[] {
1162            "Blonde",
1163            "Brown",
1164            "Red",
1165            "Black"});
1166            this.cboEditHairColour.Location = new System.Drawing.Point(93,
    ↪ 152);
1167            this.cboEditHairColour.Name = "cboEditHairColour";
1168            this.cboEditHairColour.Size = new System.Drawing.Size(121, 23);
1169            this.cboEditHairColour.TabIndex = 18;
1170            //
1171            // label8
1172            //
1173            this.label8.AutoSize = true;
1174            this.label8.ForeColor = System.Drawing.Color.White;
1175            this.label8.Location = new System.Drawing.Point(20, 50);
1176            this.label8.Name = "label8";
```

```
1177            this.label8.Size = new System.Drawing.Size(64, 15);
1178            this.label8.TabIndex = 17;
1179            this.label8.Text = "First Name";
1180            //
1181            // label5
1182            //
1183            this.label5.AutoSize = true;
1184            this.label5.ForeColor = System.Drawing.Color.White;
1185            this.label5.Location = new System.Drawing.Point(18, 155);
1186            this.label5.Name = "label5";
1187            this.label5.Size = new System.Drawing.Size(66, 15);
1188            this.label5.TabIndex = 24;
1189            this.label5.Text = "Hair colour";
1190            //
1191            // txtEditLastName
1192            //
1193            this.txtEditLastName.BackColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))
    ↪ ));
1194            this.txtEditLastName.BorderStyle = System.Windows.Forms.
    ↪ BorderStyle.None;
1195            this.txtEditLastName.Enabled = false;
1196            this.txtEditLastName.ForeColor = System.Drawing.Color.White;
1197            this.txtEditLastName.Location = new System.Drawing.Point(93, 72)
    ↪ ;
1198            this.txtEditLastName.Name = "txtEditLastName";
1199            this.txtEditLastName.Size = new System.Drawing.Size(121, 16);
1200            this.txtEditLastName.TabIndex = 15;
1201            //
1202            // label4
1203            //
1204            this.label4.AutoSize = true;
1205            this.label4.ForeColor = System.Drawing.Color.White;
1206            this.label4.Location = new System.Drawing.Point(30, 126);
1207            this.label4.Name = "label4";
1208            this.label4.Size = new System.Drawing.Size(54, 15);
1209            this.label4.TabIndex = 23;
1210            this.label4.Text = "Sexuality";
1211            //
1212            // label7
1213            //
1214            this.label7.AutoSize = true;
1215            this.label7.ForeColor = System.Drawing.Color.White;
1216            this.label7.Location = new System.Drawing.Point(20, 72);
1217            this.label7.Name = "label7";
1218            this.label7.Size = new System.Drawing.Size(63, 15);
1219            this.label7.TabIndex = 20;
1220            this.label7.Text = "Last Name";
1221            //
1222            // label3
1223            //
1224            this.label3.AutoSize = true;
1225            this.label3.ForeColor = System.Drawing.Color.White;
1226            this.label3.Location = new System.Drawing.Point(39, 97);
1227            this.label3.Name = "label3";
1228            this.label3.Size = new System.Drawing.Size(45, 15);
1229            this.label3.TabIndex = 21;
1230            this.label3.Text = "Gender";
1231            //
1232            // cboEditGender
1233            //
```

```
1234            this.cboEditGender.BackColor = System.Drawing.Color.FromArgb(((
    ↪ int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
1235            this.cboEditGender.DropDownStyle = System.Windows.Forms.
    ↪ ComboBoxStyle.DropDownList;
1236            this.cboEditGender.Enabled = false;
1237            this.cboEditGender.ForeColor = System.Drawing.Color.White;
1238            this.cboEditGender.FormattingEnabled = true;
1239            this.cboEditGender.Items.AddRange(new object[] {
1240            "Male",
1241            "Female"});
1242            this.cboEditGender.Location = new System.Drawing.Point(93, 94);
1243            this.cboEditGender.Name = "cboEditGender";
1244            this.cboEditGender.Size = new System.Drawing.Size(121, 23);
1245            this.cboEditGender.TabIndex = 16;
1246            //
1247            // tabPage7
1248            //
1249            this.tabPage7.BackColor = System.Drawing.Color.FromArgb(((int)
    ↪ (((byte)(31)))), ((int)(((byte)(31)))), ((int)(((byte)(31)))));
1250            this.tabPage7.Controls.Add(this.panel10);
1251            this.tabPage7.Controls.Add(this.panel8);
1252            this.tabPage7.Controls.Add(this.panel9);
1253            this.tabPage7.Controls.Add(this.panel11);
1254            this.tabPage7.Location = new System.Drawing.Point(4, 24);
1255            this.tabPage7.Name = "tabPage7";
1256            this.tabPage7.Size = new System.Drawing.Size(768, 306);
1257            this.tabPage7.TabIndex = 7;
1258            this.tabPage7.Text = "Jobs";
1259            //
1260            // panel10
1261            //
1262            this.panel10.Controls.Add(this.label21);
1263            this.panel10.Controls.Add(this.btnJobApplyForJob);
1264            this.panel10.Controls.Add(this.label13);
1265            this.panel10.Controls.Add(this.btnJobReshuffleJobBoard);
1266            this.panel10.Location = new System.Drawing.Point(376, 118);
1267            this.panel10.Name = "panel10";
1268            this.panel10.Size = new System.Drawing.Size(386, 91);
1269            this.panel10.TabIndex = 13;
1270            //
1271            // label21
1272            //
1273            this.label21.AutoSize = true;
1274            this.label21.ForeColor = System.Drawing.Color.White;
1275            this.label21.Location = new System.Drawing.Point(142, 58);
1276            this.label21.Name = "label21";
1277            this.label21.Size = new System.Drawing.Size(122, 15);
1278            this.label21.TabIndex = 35;
1279            this.label21.Text = "Apply for selected job";
1280            //
1281            // btnJobApplyForJob
1282            //
1283            this.btnJobApplyForJob.BackColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))
    ↪ ));
1284            this.btnJobApplyForJob.BackgroundImage = global::
    ↪ A_level_Computer_Science_Project.Properties.Resources.added;
1285            this.btnJobApplyForJob.BackgroundImageLayout = System.Windows.
    ↪ Forms.ImageLayout.Zoom;
1286            this.btnJobApplyForJob.Enabled = false;
1287            this.btnJobApplyForJob.FlatStyle = System.Windows.Forms.
```

```
    ↪ FlatStyle.Popup;
1288            this.btnJobApplyForJob.Location = new System.Drawing.Point(142,
    ↪ 3);
1289            this.btnJobApplyForJob.Name = "btnJobApplyForJob";
1290            this.btnJobApplyForJob.Size = new System.Drawing.Size(117, 52);
1291            this.btnJobApplyForJob.TabIndex = 34;
1292            this.btnJobApplyForJob.UseVisualStyleBackColor = false;
1293            this.btnJobApplyForJob.Click += new System.EventHandler(this.
    ↪ btnJobApplyForJob_Click);
1294            //
1295            // label13
1296            //
1297            this.label13.AutoSize = true;
1298            this.label13.ForeColor = System.Drawing.Color.White;
1299            this.label13.Location = new System.Drawing.Point(3, 58);
1300            this.label13.Name = "label13";
1301            this.label13.Size = new System.Drawing.Size(120, 15);
1302            this.label13.TabIndex = 33;
1303            this.label13.Text = "Refresh available jobs";
1304            //
1305            // btnJobReshuffleJobBoard
1306            //
1307            this.btnJobReshuffleJobBoard.BackColor = System.Drawing.Color.
    ↪ FromArgb(((int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte
    ↪ )(102)))));
1308            this.btnJobReshuffleJobBoard.BackgroundImage = global::
    ↪ A_level_Computer_Science_Project.Properties.Resources.refresh;
1309            this.btnJobReshuffleJobBoard.BackgroundImageLayout = System.
    ↪ Windows.Forms.ImageLayout.Zoom;
1310            this.btnJobReshuffleJobBoard.Enabled = false;
1311            this.btnJobReshuffleJobBoard.FlatStyle = System.Windows.Forms.
    ↪ FlatStyle.Popup;
1312            this.btnJobReshuffleJobBoard.Location = new System.Drawing.Point
    ↪ (3, 3);
1313            this.btnJobReshuffleJobBoard.Name = "btnJobReshuffleJobBoard";
1314            this.btnJobReshuffleJobBoard.Size = new System.Drawing.Size(117,
    ↪  52);
1315            this.btnJobReshuffleJobBoard.TabIndex = 12;
1316            this.btnJobReshuffleJobBoard.UseVisualStyleBackColor = false;
1317            this.btnJobReshuffleJobBoard.Click += new System.EventHandler(
    ↪ this.btnJobReshuffleJobBoard_Click);
1318            //
1319            // panel8
1320            //
1321            this.panel8.Controls.Add(this.btnJobQuitJob);
1322            this.panel8.Location = new System.Drawing.Point(536, 215);
1323            this.panel8.Name = "panel8";
1324            this.panel8.Size = new System.Drawing.Size(229, 88);
1325            this.panel8.TabIndex = 11;
1326            //
1327            // btnJobQuitJob
1328            //
1329            this.btnJobQuitJob.BackColor = System.Drawing.Color.FromArgb(((
    ↪ int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
1330            this.btnJobQuitJob.Enabled = false;
1331            this.btnJobQuitJob.FlatStyle = System.Windows.Forms.FlatStyle.
    ↪ Popup;
1332            this.btnJobQuitJob.ForeColor = System.Drawing.Color.White;
1333            this.btnJobQuitJob.Location = new System.Drawing.Point(43, 46);
1334            this.btnJobQuitJob.Name = "btnJobQuitJob";
1335            this.btnJobQuitJob.Size = new System.Drawing.Size(157, 39);
```

```
1336            this.btnJobQuitJob.TabIndex = 31;
1337            this.btnJobQuitJob.Text = "Quit Job";
1338            this.btnJobQuitJob.UseVisualStyleBackColor = false;
1339            this.btnJobQuitJob.Click += new System.EventHandler(this.
    ↪ button1_Click);
1340            //
1341            // panel9
1342            //
1343            this.panel9.Controls.Add(this.lblJobCurrentJobDuration);
1344            this.panel9.Controls.Add(this.lblJobCurrentJobBand);
1345            this.panel9.Controls.Add(this.lblJobCurrentJobSalary);
1346            this.panel9.Controls.Add(this.lblJobCurrentJobTitle);
1347            this.panel9.Controls.Add(this.label12);
1348            this.panel9.Location = new System.Drawing.Point(379, 3);
1349            this.panel9.Name = "panel9";
1350            this.panel9.Size = new System.Drawing.Size(386, 109);
1351            this.panel9.TabIndex = 10;
1352            //
1353            // lblJobCurrentJobDuration
1354            //
1355            this.lblJobCurrentJobDuration.AutoSize = true;
1356            this.lblJobCurrentJobDuration.ForeColor = System.Drawing.Color.
    ↪ White;
1357            this.lblJobCurrentJobDuration.Location = new System.Drawing.
    ↪ Point(6, 83);
1358            this.lblJobCurrentJobDuration.Name = "lblJobCurrentJobDuration";
1359            this.lblJobCurrentJobDuration.Size = new System.Drawing.Size
    ↪ (120, 15);
1360            this.lblJobCurrentJobDuration.TabIndex = 33;
1361            this.lblJobCurrentJobDuration.Text = "Current Job Duration:";
1362            //
1363            // lblJobCurrentJobBand
1364            //
1365            this.lblJobCurrentJobBand.AutoSize = true;
1366            this.lblJobCurrentJobBand.ForeColor = System.Drawing.Color.White
    ↪ ;
1367            this.lblJobCurrentJobBand.Location = new System.Drawing.Point
    ↪ (25, 68);
1368            this.lblJobCurrentJobBand.Name = "lblJobCurrentJobBand";
1369            this.lblJobCurrentJobBand.Size = new System.Drawing.Size(104,
    ↪ 15);
1370            this.lblJobCurrentJobBand.TabIndex = 32;
1371            this.lblJobCurrentJobBand.Text = "Current Job Band: ";
1372            //
1373            // lblJobCurrentJobSalary
1374            //
1375            this.lblJobCurrentJobSalary.AutoSize = true;
1376            this.lblJobCurrentJobSalary.ForeColor = System.Drawing.Color.
    ↪ White;
1377            this.lblJobCurrentJobSalary.Location = new System.Drawing.Point
    ↪ (21, 53);
1378            this.lblJobCurrentJobSalary.Name = "lblJobCurrentJobSalary";
1379            this.lblJobCurrentJobSalary.Size = new System.Drawing.Size(108,
    ↪ 15);
1380            this.lblJobCurrentJobSalary.TabIndex = 31;
1381            this.lblJobCurrentJobSalary.Text = "Current Job Salary: ";
1382            //
1383            // lblJobCurrentJobTitle
1384            //
1385            this.lblJobCurrentJobTitle.AutoSize = true;
1386            this.lblJobCurrentJobTitle.ForeColor = System.Drawing.Color.
```

```
    ↪ White;
1387            this.lblJobCurrentJobTitle.Location = new System.Drawing.Point
    ↪ (30, 38);
1388            this.lblJobCurrentJobTitle.Name = "lblJobCurrentJobTitle";
1389            this.lblJobCurrentJobTitle.Size = new System.Drawing.Size(99,
    ↪ 15);
1390            this.lblJobCurrentJobTitle.TabIndex = 30;
1391            this.lblJobCurrentJobTitle.Text = "Current Job Title: ";
1392            //
1393            // label12
1394            //
1395            this.label12.AutoSize = true;
1396            this.label12.Font = new System.Drawing.Font("Segoe UI", 14.25F,
    ↪ System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point);
1397            this.label12.ForeColor = System.Drawing.Color.White;
1398            this.label12.Location = new System.Drawing.Point(129, 10);
1399            this.label12.Name = "label12";
1400            this.label12.Size = new System.Drawing.Size(118, 25);
1401            this.label12.TabIndex = 29;
1402            this.label12.Text = "Current Job";
1403            this.label12.TextAlign = System.Drawing.ContentAlignment.
    ↪ TopCenter;
1404            this.ttMainToolTip.SetToolTip(this.label12, "Unlocks when Main
    ↪ Character turns 13");
1405            //
1406            // panel11
1407            //
1408            this.panel11.Controls.Add(this.dgvEmpAvailableJobs);
1409            this.panel11.Location = new System.Drawing.Point(3, 3);
1410            this.panel11.Name = "panel11";
1411            this.panel11.Size = new System.Drawing.Size(370, 300);
1412            this.panel11.TabIndex = 8;
1413            //
1414            // dgvEmpAvailableJobs
1415            //
1416            this.dgvEmpAvailableJobs.AllowUserToAddRows = false;
1417            this.dgvEmpAvailableJobs.AllowUserToDeleteRows = false;
1418            this.dgvEmpAvailableJobs.AllowUserToResizeColumns = false;
1419            this.dgvEmpAvailableJobs.AllowUserToResizeRows = false;
1420            dataGridViewCellStyle4.BackColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))
    ↪ ));
1421            this.dgvEmpAvailableJobs.AlternatingRowsDefaultCellStyle =
    ↪ dataGridViewCellStyle4;
1422            this.dgvEmpAvailableJobs.BackgroundColor = System.Drawing.Color.
    ↪ FromArgb(((int)(((byte)(31)))), ((int)(((byte)(31)))), ((int)(((byte)
    ↪ (31)))));
1423            this.dgvEmpAvailableJobs.CellBorderStyle = System.Windows.Forms.
    ↪ DataGridViewCellBorderStyle.SingleHorizontal;
1424            dataGridViewCellStyle5.Alignment = System.Windows.Forms.
    ↪ DataGridViewContentAlignment.MiddleCenter;
1425            dataGridViewCellStyle5.BackColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(100)))), ((int)(((byte)(164)))), ((int)(((byte)(209)))
    ↪ ));
1426            dataGridViewCellStyle5.Font = new System.Drawing.Font("Segoe UI"
    ↪ , 9.75F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.
    ↪ Point);
1427            dataGridViewCellStyle5.ForeColor = System.Drawing.Color.White;
1428            dataGridViewCellStyle5.SelectionBackColor = System.Drawing.Color
    ↪ .FromArgb(((int)(((byte)(100)))), ((int)(((byte)(164)))), ((int)(((
    ↪ byte)(209)))));
```

```
1429            dataGridViewCellStyle5.SelectionForeColor = System.Drawing.
    ↪ SystemColors.HighlightText;
1430            dataGridViewCellStyle5.WrapMode = System.Windows.Forms.
    ↪ DataGridViewTriState.True;
1431            this.dgvEmpAvailableJobs.ColumnHeadersDefaultCellStyle =
    ↪ dataGridViewCellStyle5;
1432            this.dgvEmpAvailableJobs.ColumnHeadersHeightSizeMode = System.
    ↪ Windows.Forms.DataGridViewColumnHeadersHeightSizeMode.AutoSize;
1433            this.dgvEmpAvailableJobs.Columns.AddRange(new System.Windows.
    ↪ Forms.DataGridViewColumn[] {
1434            this.dataGridViewTextBoxColumn1,
1435            this.dataGridViewTextBoxColumn2,
1436            this.Column1});
1437            dataGridViewCellStyle6.Alignment = System.Windows.Forms.
    ↪ DataGridViewContentAlignment.MiddleLeft;
1438            dataGridViewCellStyle6.BackColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))
    ↪ ));
1439            dataGridViewCellStyle6.Font = new System.Drawing.Font("Segoe UI"
    ↪ , 9F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.
    ↪ Point);
1440            dataGridViewCellStyle6.ForeColor = System.Drawing.Color.White;
1441            dataGridViewCellStyle6.SelectionBackColor = System.Drawing.Color
    ↪ .FromArgb(((int)(((byte)(167)))), ((int)(((byte)(108)))), ((int)(((
    ↪ byte)(179)))));
1442            dataGridViewCellStyle6.SelectionForeColor = System.Drawing.
    ↪ SystemColors.HighlightText;
1443            dataGridViewCellStyle6.WrapMode = System.Windows.Forms.
    ↪ DataGridViewTriState.True;
1444            this.dgvEmpAvailableJobs.DefaultCellStyle =
    ↪ dataGridViewCellStyle6;
1445            this.dgvEmpAvailableJobs.Dock = System.Windows.Forms.DockStyle.
    ↪ Fill;
1446            this.dgvEmpAvailableJobs.GridColor = System.Drawing.Color.White;
1447            this.dgvEmpAvailableJobs.Location = new System.Drawing.Point(0,
    ↪ 0);
1448            this.dgvEmpAvailableJobs.Name = "dgvEmpAvailableJobs";
1449            this.dgvEmpAvailableJobs.ReadOnly = true;
1450            this.dgvEmpAvailableJobs.RowHeadersVisible = false;
1451            this.dgvEmpAvailableJobs.RowTemplate.Height = 25;
1452            this.dgvEmpAvailableJobs.SelectionMode = System.Windows.Forms.
    ↪ DataGridViewSelectionMode.FullRowSelect;
1453            this.dgvEmpAvailableJobs.Size = new System.Drawing.Size(370,
    ↪ 300);
1454            this.dgvEmpAvailableJobs.TabIndex = 1;
1455            //
1456            // dataGridViewTextBoxColumn1
1457            //
1458            this.dataGridViewTextBoxColumn1.HeaderText = "Job Title";
1459            this.dataGridViewTextBoxColumn1.Name = "
    ↪ dataGridViewTextBoxColumn1";
1460            this.dataGridViewTextBoxColumn1.ReadOnly = true;
1461            this.dataGridViewTextBoxColumn1.SortMode = System.Windows.Forms.
    ↪ DataGridViewColumnSortMode.NotSortable;
1462            this.dataGridViewTextBoxColumn1.Width = 150;
1463            //
1464            // dataGridViewTextBoxColumn2
1465            //
1466            this.dataGridViewTextBoxColumn2.HeaderText = "Salary";
1467            this.dataGridViewTextBoxColumn2.Name = "
    ↪ dataGridViewTextBoxColumn2";
```

```
1468            this.dataGridViewTextBoxColumn2.ReadOnly = true;
1469            this.dataGridViewTextBoxColumn2.SortMode = System.Windows.Forms.
    ↪ DataGridViewColumnSortMode.NotSortable;
1470            //
1471            // Column1
1472            //
1473            this.Column1.HeaderText = "Band";
1474            this.Column1.Name = "Column1";
1475            this.Column1.ReadOnly = true;
1476            //
1477            // tabPage8
1478            //
1479            this.tabPage8.BackColor = System.Drawing.Color.FromArgb(((int)
    ↪ (((byte)(31)))), ((int)(((byte)(31)))), ((int)(((byte)(31)))));
1480            this.tabPage8.Controls.Add(this.panel29);
1481            this.tabPage8.Controls.Add(this.panel26);
1482            this.tabPage8.Controls.Add(this.panel27);
1483            this.tabPage8.Controls.Add(this.panel28);
1484            this.tabPage8.Location = new System.Drawing.Point(4, 24);
1485            this.tabPage8.Name = "tabPage8";
1486            this.tabPage8.Padding = new System.Windows.Forms.Padding(3);
1487            this.tabPage8.Size = new System.Drawing.Size(768, 306);
1488            this.tabPage8.TabIndex = 8;
1489            this.tabPage8.Text = "Partners";
1490            //
1491            // panel29
1492            //
1493            this.panel29.Location = new System.Drawing.Point(536, 215);
1494            this.panel29.Name = "panel29";
1495            this.panel29.Size = new System.Drawing.Size(229, 88);
1496            this.panel29.TabIndex = 7;
1497            //
1498            // panel26
1499            //
1500            this.panel26.Controls.Add(this.btnPartDump);
1501            this.panel26.Controls.Add(this.btnAskOnDate);
1502            this.panel26.Controls.Add(this.btnParGenMore);
1503            this.panel26.Location = new System.Drawing.Point(379, 3);
1504            this.panel26.Name = "panel26";
1505            this.panel26.Size = new System.Drawing.Size(386, 100);
1506            this.panel26.TabIndex = 6;
1507            //
1508            // btnAskOnDate
1509            //
1510            this.btnAskOnDate.BackColor = System.Drawing.Color.FromArgb(((
    ↪ int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
1511            this.btnAskOnDate.FlatStyle = System.Windows.Forms.FlatStyle.
    ↪ Popup;
1512            this.btnAskOnDate.ForeColor = System.Drawing.Color.White;
1513            this.btnAskOnDate.Location = new System.Drawing.Point(78, 0);
1514            this.btnAskOnDate.Name = "btnAskOnDate";
1515            this.btnAskOnDate.Size = new System.Drawing.Size(75, 100);
1516            this.btnAskOnDate.TabIndex = 1;
1517            this.btnAskOnDate.Text = "Ask on date";
1518            this.btnAskOnDate.UseVisualStyleBackColor = false;
1519            this.btnAskOnDate.Click += new System.EventHandler(this.
    ↪ btnAskOnDate_Click);
1520            //
1521            // btnParGenMore
1522            //
1523            this.btnParGenMore.BackColor = System.Drawing.Color.FromArgb(((
```

```
    ↪ int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
1524            this.btnParGenMore.FlatStyle = System.Windows.Forms.FlatStyle.
    ↪ Popup;
1525            this.btnParGenMore.ForeColor = System.Drawing.Color.White;
1526            this.btnParGenMore.Location = new System.Drawing.Point(0, 0);
1527            this.btnParGenMore.Name = "btnParGenMore";
1528            this.btnParGenMore.Size = new System.Drawing.Size(72, 100);
1529            this.btnParGenMore.TabIndex = 0;
1530            this.btnParGenMore.Text = "Generate more potential partners";
1531            this.btnParGenMore.UseVisualStyleBackColor = false;
1532            this.btnParGenMore.Click += new System.EventHandler(this.
    ↪ btnParGenMore_Click);
1533            //
1534            // panel27
1535            //
1536            this.panel27.Controls.Add(this.lblPartAge);
1537            this.panel27.Controls.Add(this.lblPartDateOfBirth);
1538            this.panel27.Controls.Add(this.lblPartLastName);
1539            this.panel27.Controls.Add(this.lblPartFirstName);
1540            this.panel27.Location = new System.Drawing.Point(379, 109);
1541            this.panel27.Name = "panel27";
1542            this.panel27.Size = new System.Drawing.Size(386, 100);
1543            this.panel27.TabIndex = 5;
1544            //
1545            // lblPartAge
1546            //
1547            this.lblPartAge.AutoSize = true;
1548            this.lblPartAge.ForeColor = System.Drawing.Color.White;
1549            this.lblPartAge.Location = new System.Drawing.Point(18, 58);
1550            this.lblPartAge.Name = "lblPartAge";
1551            this.lblPartAge.Size = new System.Drawing.Size(31, 15);
1552            this.lblPartAge.TabIndex = 3;
1553            this.lblPartAge.Text = "Age:";
1554            //
1555            // lblPartDateOfBirth
1556            //
1557            this.lblPartDateOfBirth.AutoSize = true;
1558            this.lblPartDateOfBirth.ForeColor = System.Drawing.Color.White;
1559            this.lblPartDateOfBirth.Location = new System.Drawing.Point(18,
    ↪ 43);
1560            this.lblPartDateOfBirth.Name = "lblPartDateOfBirth";
1561            this.lblPartDateOfBirth.Size = new System.Drawing.Size(78, 15);
1562            this.lblPartDateOfBirth.TabIndex = 2;
1563            this.lblPartDateOfBirth.Text = "Date Of Birth:";
1564            //
1565            // lblPartLastName
1566            //
1567            this.lblPartLastName.AutoSize = true;
1568            this.lblPartLastName.ForeColor = System.Drawing.Color.White;
1569            this.lblPartLastName.Location = new System.Drawing.Point(18, 28)
    ↪ ;
1570            this.lblPartLastName.Name = "lblPartLastName";
1571            this.lblPartLastName.Size = new System.Drawing.Size(69, 15);
1572            this.lblPartLastName.TabIndex = 1;
1573            this.lblPartLastName.Text = "Last Name: ";
1574            //
1575            // lblPartFirstName
1576            //
1577            this.lblPartFirstName.AutoSize = true;
1578            this.lblPartFirstName.ForeColor = System.Drawing.Color.White;
1579            this.lblPartFirstName.Location = new System.Drawing.Point(18,
```

```
    ↪ 13);
1580            this.lblPartFirstName.Name = "lblPartFirstName";
1581            this.lblPartFirstName.Size = new System.Drawing.Size(70, 15);
1582            this.lblPartFirstName.TabIndex = 0;
1583            this.lblPartFirstName.Text = "First Name: ";
1584            //
1585            // panel28
1586            //
1587            this.panel28.Controls.Add(this.dgvPartPotential);
1588            this.panel28.Location = new System.Drawing.Point(3, 3);
1589            this.panel28.Name = "panel28";
1590            this.panel28.Size = new System.Drawing.Size(370, 300);
1591            this.panel28.TabIndex = 3;
1592            //
1593            // dgvPartPotential
1594            //
1595            this.dgvPartPotential.AllowUserToAddRows = false;
1596            this.dgvPartPotential.AllowUserToDeleteRows = false;
1597            this.dgvPartPotential.AllowUserToResizeColumns = false;
1598            this.dgvPartPotential.AllowUserToResizeRows = false;
1599            dataGridViewCellStyle7.BackColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))
    ↪ ));
1600            this.dgvPartPotential.AlternatingRowsDefaultCellStyle =
    ↪ dataGridViewCellStyle7;
1601            this.dgvPartPotential.BackgroundColor = System.Drawing.Color.
    ↪ FromArgb(((int)(((byte)(31)))), ((int)(((byte)(31)))), ((int)(((byte)
    ↪ (31)))));
1602            this.dgvPartPotential.CellBorderStyle = System.Windows.Forms.
    ↪ DataGridViewCellBorderStyle.SingleHorizontal;
1603            dataGridViewCellStyle8.Alignment = System.Windows.Forms.
    ↪ DataGridViewContentAlignment.MiddleCenter;
1604            dataGridViewCellStyle8.BackColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(100)))), ((int)(((byte)(164)))), ((int)(((byte)(209)))
    ↪ ));
1605            dataGridViewCellStyle8.Font = new System.Drawing.Font("Segoe UI"
    ↪ , 9.75F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.
    ↪ Point);
1606            dataGridViewCellStyle8.ForeColor = System.Drawing.Color.White;
1607            dataGridViewCellStyle8.SelectionBackColor = System.Drawing.Color
    ↪ .FromArgb(((int)(((byte)(100)))), ((int)(((byte)(164)))), ((int)(((
    ↪ byte)(209)))));
1608            dataGridViewCellStyle8.SelectionForeColor = System.Drawing.
    ↪ SystemColors.HighlightText;
1609            dataGridViewCellStyle8.WrapMode = System.Windows.Forms.
    ↪ DataGridViewTriState.True;
1610            this.dgvPartPotential.ColumnHeadersDefaultCellStyle =
    ↪ dataGridViewCellStyle8;
1611            this.dgvPartPotential.ColumnHeadersHeightSizeMode = System.
    ↪ Windows.Forms.DataGridViewColumnHeadersHeightSizeMode.AutoSize;
1612            this.dgvPartPotential.Columns.AddRange(new System.Windows.Forms.
    ↪ DataGridViewColumn[] {
1613            this.dataGridViewTextBoxColumn3,
1614            this.dataGridViewTextBoxColumn4,
1615            this.dataGridViewTextBoxColumn5,
1616            this.DateOfBirth,
1617            this.index});
1618            dataGridViewCellStyle9.Alignment = System.Windows.Forms.
    ↪ DataGridViewContentAlignment.MiddleLeft;
1619            dataGridViewCellStyle9.BackColor = System.Drawing.Color.FromArgb
    ↪ (((int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))
```

```
    ↪ ));
1620            dataGridViewCellStyle9.Font = new System.Drawing.Font("Segoe UI"
    ↪ , 9F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.
    ↪ Point);
1621            dataGridViewCellStyle9.ForeColor = System.Drawing.Color.White;
1622            dataGridViewCellStyle9.SelectionBackColor = System.Drawing.Color
    ↪ .FromArgb(((int)(((byte)(167)))), ((int)(((byte)(108)))), ((int)(((
    ↪ byte)(179)))));
1623            dataGridViewCellStyle9.SelectionForeColor = System.Drawing.
    ↪ SystemColors.HighlightText;
1624            dataGridViewCellStyle9.WrapMode = System.Windows.Forms.
    ↪ DataGridViewTriState.True;
1625            this.dgvPartPotential.DefaultCellStyle = dataGridViewCellStyle9;
1626            this.dgvPartPotential.Dock = System.Windows.Forms.DockStyle.Fill
    ↪ ;
1627            this.dgvPartPotential.GridColor = System.Drawing.Color.White;
1628            this.dgvPartPotential.Location = new System.Drawing.Point(0, 0);
1629            this.dgvPartPotential.Name = "dgvPartPotential";
1630            this.dgvPartPotential.ReadOnly = true;
1631            this.dgvPartPotential.RowHeadersVisible = false;
1632            this.dgvPartPotential.RowTemplate.Height = 25;
1633            this.dgvPartPotential.SelectionMode = System.Windows.Forms.
    ↪ DataGridViewSelectionMode.FullRowSelect;
1634            this.dgvPartPotential.Size = new System.Drawing.Size(370, 300);
1635            this.dgvPartPotential.TabIndex = 1;
1636            //
1637            // dataGridViewTextBoxColumn3
1638            //
1639            this.dataGridViewTextBoxColumn3.HeaderText = "First Name";
1640            this.dataGridViewTextBoxColumn3.Name = "
    ↪ dataGridViewTextBoxColumn3";
1641            this.dataGridViewTextBoxColumn3.ReadOnly = true;
1642            this.dataGridViewTextBoxColumn3.SortMode = System.Windows.Forms.
    ↪ DataGridViewColumnSortMode.NotSortable;
1643            //
1644            // dataGridViewTextBoxColumn4
1645            //
1646            this.dataGridViewTextBoxColumn4.HeaderText = "Last Name";
1647            this.dataGridViewTextBoxColumn4.Name = "
    ↪ dataGridViewTextBoxColumn4";
1648            this.dataGridViewTextBoxColumn4.ReadOnly = true;
1649            this.dataGridViewTextBoxColumn4.SortMode = System.Windows.Forms.
    ↪ DataGridViewColumnSortMode.NotSortable;
1650            //
1651            // dataGridViewTextBoxColumn5
1652            //
1653            this.dataGridViewTextBoxColumn5.HeaderText = "Gender";
1654            this.dataGridViewTextBoxColumn5.Name = "
    ↪ dataGridViewTextBoxColumn5";
1655            this.dataGridViewTextBoxColumn5.ReadOnly = true;
1656            this.dataGridViewTextBoxColumn5.SortMode = System.Windows.Forms.
    ↪ DataGridViewColumnSortMode.NotSortable;
1657            this.dataGridViewTextBoxColumn5.Width = 90;
1658            //
1659            // DateOfBirth
1660            //
1661            this.DateOfBirth.HeaderText = "Date of Birth";
1662            this.DateOfBirth.Name = "DateOfBirth";
1663            this.DateOfBirth.ReadOnly = true;
1664            //
1665            // index
```

```
1666            //
1667            this.index.HeaderText = "index";
1668            this.index.Name = "index";
1669            this.index.ReadOnly = true;
1670            this.index.Visible = false;
1671            //
1672            // ttMainToolTip
1673            //
1674            this.ttMainToolTip.BackColor = System.Drawing.Color.FromArgb(((
    ↪ int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
1675            this.ttMainToolTip.ForeColor = System.Drawing.Color.White;
1676            this.ttMainToolTip.ToolTipIcon = System.Windows.Forms.
    ↪ ToolTipIcon.Info;
1677            this.ttMainToolTip.ToolTipTitle = "Score Information";
1678            //
1679            // btnPartDump
1680            //
1681            this.btnPartDump.BackColor = System.Drawing.Color.FromArgb(((int
    ↪ )(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
1682            this.btnPartDump.FlatStyle = System.Windows.Forms.FlatStyle.
    ↪ Popup;
1683            this.btnPartDump.ForeColor = System.Drawing.Color.White;
1684            this.btnPartDump.Location = new System.Drawing.Point(168, 0);
1685            this.btnPartDump.Name = "btnPartDump";
1686            this.btnPartDump.Size = new System.Drawing.Size(75, 100);
1687            this.btnPartDump.TabIndex = 2;
1688            this.btnPartDump.Text = "Dump your partner";
1689            this.btnPartDump.UseVisualStyleBackColor = false;
1690            this.btnPartDump.Click += new System.EventHandler(this.
    ↪ btnPartDump_Click);
1691            //
1692            // frmMainGameScreen
1693            //
1694            this.AutoScaleDimensions = new System.Drawing.SizeF(7F, 15F);
1695            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
1696            this.BackColor = System.Drawing.Color.FromArgb(((int)(((byte)
    ↪ (31)))), ((int)(((byte)(31)))), ((int)(((byte)(31)))));
1697            this.ClientSize = new System.Drawing.Size(800, 450);
1698            this.Controls.Add(this.panel2);
1699            this.Controls.Add(this.panel1);
1700            this.Icon = ((System.Drawing.Icon)(resources.GetObject("$this.
    ↪ Icon")));
1701            this.Name = "frmMainGameScreen";
1702            this.Text = "Virtual-Life main game screen";
1703            this.Load += new System.EventHandler(this.frmMainGameScreen_Load
    ↪ );
1704            this.panel1.ResumeLayout(false);
1705            this.panel1.PerformLayout();
1706            ((System.ComponentModel.ISupportInitialize)(this.pictureBox1)).
    ↪ EndInit();
1707            this.panel2.ResumeLayout(false);
1708            this.tabControl1.ResumeLayout(false);
1709            this.tabPage1.ResumeLayout(false);
1710            this.panel7.ResumeLayout(false);
1711            this.panel6.ResumeLayout(false);
1712            this.panel6.PerformLayout();
1713            this.panel5.ResumeLayout(false);
1714            this.panel5.PerformLayout();
1715            this.panel3.ResumeLayout(false);
1716            this.panel3.PerformLayout();
1717            this.tabPage2.ResumeLayout(false);
```

```
1718            this.panel13.ResumeLayout(false);
1719            this.panel13.PerformLayout();
1720            this.panel4.ResumeLayout(false);
1721            ((System.ComponentModel.ISupportInitialize)(this.dgvFamily)).
    ↪ EndInit();
1722            this.tabPage3.ResumeLayout(false);
1723            this.panel17.ResumeLayout(false);
1724            this.panel17.PerformLayout();
1725            this.panel14.ResumeLayout(false);
1726            this.panel15.ResumeLayout(false);
1727            this.panel15.PerformLayout();
1728            this.tabPage5.ResumeLayout(false);
1729            this.panel18.ResumeLayout(false);
1730            this.panel18.PerformLayout();
1731            this.panel19.ResumeLayout(false);
1732            this.tabPage6.ResumeLayout(false);
1733            this.panel23.ResumeLayout(false);
1734            this.panel22.ResumeLayout(false);
1735            this.panel22.PerformLayout();
1736            this.panel20.ResumeLayout(false);
1737            this.panel20.PerformLayout();
1738            this.tabPage7.ResumeLayout(false);
1739            this.panel10.ResumeLayout(false);
1740            this.panel10.PerformLayout();
1741            this.panel8.ResumeLayout(false);
1742            this.panel9.ResumeLayout(false);
1743            this.panel9.PerformLayout();
1744            this.panel11.ResumeLayout(false);
1745            ((System.ComponentModel.ISupportInitialize)(this.
    ↪ dgvEmpAvailableJobs)).EndInit();
1746            this.tabPage8.ResumeLayout(false);
1747            this.panel26.ResumeLayout(false);
1748            this.panel27.ResumeLayout(false);
1749            this.panel27.PerformLayout();
1750            this.panel28.ResumeLayout(false);
1751            ((System.ComponentModel.ISupportInitialize)(this.
    ↪ dgvPartPotential)).EndInit();
1752            this.ResumeLayout(false);
1753
1754        }
1755
1756        #endregion
1757        private System.Windows.Forms.Button btnSaveToFile;
1758        private System.Windows.Forms.Panel panel1;
1759        private System.Windows.Forms.Button btnSaveAndHome;
1760        private System.Windows.Forms.PictureBox pictureBox1;
1761        private System.Windows.Forms.Panel panel2;
1762        private System.Windows.Forms.TabControl tabControl1;
1763        private System.Windows.Forms.TabPage tabPage1;
1764        private System.Windows.Forms.TabPage tabPage2;
1765        private System.Windows.Forms.Panel panel7;
1766        private System.Windows.Forms.Panel panel6;
1767        private System.Windows.Forms.Panel panel5;
1768        private System.Windows.Forms.Panel panel3;
1769        private System.Windows.Forms.TextBox txtEventBox;
1770        private System.Windows.Forms.Label label1;
1771        private System.Windows.Forms.Panel panel4;
1772        private System.Windows.Forms.Label lblMPFirstName;
1773        private System.Windows.Forms.Panel panel12;
1774        private System.Windows.Forms.Panel panel13;
1775        private System.Windows.Forms.Label lblMPGender;
```

```
1776        private System.Windows.Forms.Label lblMPLastName;
1777        private System.Windows.Forms.Label lblMPDateOfBirth;
1778        private System.Windows.Forms.Label lblMPAge;
1779        private System.Windows.Forms.Label lblMPSexuality;
1780        private System.Windows.Forms.Label label2;
1781        private System.Windows.Forms.Button btnAgeUp;
1782        private System.Windows.Forms.Label lblInGameDate;
1783        private System.Windows.Forms.DataGridView dgvFamily;
1784        private System.Windows.Forms.DataGridViewTextBoxColumn
    ↪ relationshipToMain;
1785        private System.Windows.Forms.DataGridViewTextBoxColumn firstName;
1786        private System.Windows.Forms.DataGridViewTextBoxColumn lastName;
1787        private System.Windows.Forms.Label lblFamReasonForDeath;
1788        private System.Windows.Forms.Label lblFamDateOfDeath;
1789        private System.Windows.Forms.Label lblFamInRelationship;
1790        private System.Windows.Forms.Label lblFamDateOfBirth;
1791        private System.Windows.Forms.Label lblFamAge;
1792        private System.Windows.Forms.Label lblFamSexuality;
1793        private System.Windows.Forms.Label lblFamGender;
1794        private System.Windows.Forms.Label lblFamLastName;
1795        private System.Windows.Forms.Label lblFamFirstName;
1796        private System.Windows.Forms.Label lblFamLivingStatus;
1797        private System.Windows.Forms.Label lblFamRealtionshipToMain;
1798        private System.Windows.Forms.ProgressBar prbEducationScore;
1799        private System.Windows.Forms.Label label18;
1800        private System.Windows.Forms.ProgressBar prbLifeScore;
1801        private System.Windows.Forms.ProgressBar prbMedicalScore;
1802        private System.Windows.Forms.ProgressBar prbHappinessScore;
1803        private System.Windows.Forms.ProgressBar prbJobScore;
1804        private System.Windows.Forms.Label label17;
1805        private System.Windows.Forms.Label label16;
1806        private System.Windows.Forms.Label label15;
1807        private System.Windows.Forms.Label label14;
1808        private System.Windows.Forms.TabPage tabPage3;
1809        private System.Windows.Forms.Panel panel17;
1810        private System.Windows.Forms.TextBox txtEduEducationEvents;
1811        private System.Windows.Forms.Label label20;
1812        private System.Windows.Forms.Panel panel14;
1813        private System.Windows.Forms.Panel panel15;
1814        private System.Windows.Forms.Label lblEduEducationSituation;
1815        private System.Windows.Forms.Label lblEduCurrentEducationPoints;
1816        private System.Windows.Forms.Panel panel16;
1817        private System.Windows.Forms.Label lblEduCurrentYear;
1818        private System.Windows.Forms.Button btnEduWithdraw;
1819        private System.Windows.Forms.ToolTip ttMainToolTip;
1820        private System.Windows.Forms.TabPage tabPage5;
1821        private System.Windows.Forms.Panel panel18;
1822        private System.Windows.Forms.TextBox txtCriCrimeEvents;
1823        private System.Windows.Forms.Label label19;
1824        private System.Windows.Forms.Panel panel19;
1825        private System.Windows.Forms.Button btnCriCommitCrime;
1826        private System.Windows.Forms.Panel panel21;
1827        private System.Windows.Forms.TabPage tabPage6;
1828        private System.Windows.Forms.Panel panel25;
1829        private System.Windows.Forms.Button btnEditSaveChanges;
1830        private System.Windows.Forms.Panel panel23;
1831        private System.Windows.Forms.Panel panel24;
1832        private System.Windows.Forms.Panel panel22;
1833        private System.Windows.Forms.Panel panel20;
1834        private System.Windows.Forms.Label label9;
1835        private System.Windows.Forms.ComboBox cboEditSexuality;
```

```
1836        private System.Windows.Forms.ComboBox cboEditEyeColour;
1837        private System.Windows.Forms.Label label6;
1838        private System.Windows.Forms.TextBox txtEditFirstName;
1839        private System.Windows.Forms.ComboBox cboEditHairColour;
1840        private System.Windows.Forms.Label label8;
1841        private System.Windows.Forms.Label label5;
1842        private System.Windows.Forms.TextBox txtEditLastName;
1843        private System.Windows.Forms.Label label4;
1844        private System.Windows.Forms.Label label7;
1845        private System.Windows.Forms.Label label3;
1846        private System.Windows.Forms.ComboBox cboEditGender;
1847        private System.Windows.Forms.Button btnEndLife;
1848        private System.Windows.Forms.Label label11;
1849        private System.Windows.Forms.Label label10;
1850        private System.Windows.Forms.TabPage tabPage7;
1851        private System.Windows.Forms.Panel panel8;
1852        private System.Windows.Forms.Panel panel9;
1853        private System.Windows.Forms.Panel panel11;
1854        private System.Windows.Forms.TabPage tabPage8;
1855        private System.Windows.Forms.Panel panel29;
1856        private System.Windows.Forms.Panel panel26;
1857        private System.Windows.Forms.Panel panel27;
1858        private System.Windows.Forms.Panel panel28;
1859        private System.Windows.Forms.DataGridView dgvEmpAvailableJobs;
1860        private System.Windows.Forms.DataGridViewTextBoxColumn
    ↪ dataGridViewTextBoxColumn1;
1861        private System.Windows.Forms.DataGridViewTextBoxColumn
    ↪ dataGridViewTextBoxColumn2;
1862        private System.Windows.Forms.DataGridViewTextBoxColumn Column1;
1863        private System.Windows.Forms.Button btnJobQuitJob;
1864        private System.Windows.Forms.Label lblJobCurrentJobBand;
1865        private System.Windows.Forms.Label lblJobCurrentJobSalary;
1866        private System.Windows.Forms.Label lblJobCurrentJobTitle;
1867        private System.Windows.Forms.Label label12;
1868        private System.Windows.Forms.Button btnJobReshuffleJobBoard;
1869        private System.Windows.Forms.Panel panel10;
1870        private System.Windows.Forms.Label label13;
1871        private System.Windows.Forms.Label label21;
1872        private System.Windows.Forms.Button btnJobApplyForJob;
1873        private System.Windows.Forms.Label lblJobCurrentJobDuration;
1874        private System.Windows.Forms.Button btnParGenMore;
1875        private System.Windows.Forms.DataGridView dgvPartPotential;
1876        private System.Windows.Forms.Button btnAskOnDate;
1877        private System.Windows.Forms.DataGridViewTextBoxColumn
    ↪ dataGridViewTextBoxColumn3;
1878        private System.Windows.Forms.DataGridViewTextBoxColumn
    ↪ dataGridViewTextBoxColumn4;
1879        private System.Windows.Forms.DataGridViewTextBoxColumn
    ↪ dataGridViewTextBoxColumn5;
1880        private System.Windows.Forms.DataGridViewTextBoxColumn DateOfBirth;
1881        private System.Windows.Forms.DataGridViewTextBoxColumn index;
1882        private System.Windows.Forms.Label lblPartFirstName;
1883        private System.Windows.Forms.Label lblPartAge;
1884        private System.Windows.Forms.Label lblPartDateOfBirth;
1885        private System.Windows.Forms.Label lblPartLastName;
1886        private System.Windows.Forms.Button btnPartDump;
1887    }
1888 }
```

Listing A.10: FILE/frmMainGameScreen.Designer.cs

A.1.6 Show Saved Games

[FILE]frmShowSavedGames.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.IO;
using Newtonsoft.Json.Linq;
using Newtonsoft.Json;


namespace A_level_Computer_Science_Project
{
    public partial class frmShowSavedGames : Form
    {
        public MainCharacter mainCharacter = new MainCharacter(); //creating new public object of type Person called mainCharacter
        public Event[] eventArray = new Event[1000];
        public string filepath;
        public ControlClass controlClass = new ControlClass();
        public frmShowSavedGames()
        {
            InitializeComponent();
        }

        private void btnLoad_Click(object sender, EventArgs e)
        {
            openFile();
            //MessageBox.Show(filepath);
        }
        public void openFile()
        {

            //code below originally came from Microsoft docs - then it was mutilated by me!
            // fileContent = string.Empty;
            //controlClass.Filepath = string.Empty;

            //now try using a folderbrowserdialog
            FolderBrowserDialog folderDlg = new FolderBrowserDialog();
            folderDlg.ShowNewFolderButton = true;
            // Show the FolderBrowserDialog.
            DialogResult result = folderDlg.ShowDialog();
            if (result == DialogResult.OK)
            {
                //txtSaveLocation.Text = folderDlg.SelectedPath;
                Environment.SpecialFolder root = folderDlg.RootFolder;
                string loadFilepath = folderDlg.SelectedPath.ToString();
                //MessageBox.Show(loadFilepath);
                filepath = loadFilepath;

                //Read data in from the file and deserialize it into the objects
                MainCharacter mainCharacter = JsonConvert.DeserializeObject<MainCharacter>(File.ReadAllText(loadFilepath + "\\mainCharacter.json"));
```

```
53              //Note: Just do main character on its own here then do the
    ↪ rest when the data is actually passed across (due to accessibility
    ↪ issues)
54
55
56              populateForm(mainCharacter);
57
58          }
59
60
61      }
62      public void populateForm(MainCharacter mainCharacter)
63      {
64          //Main character
65          txtFirstName.Text = mainCharacter.FirstName;
66          txtLastName.Text = mainCharacter.LastName;
67          txtGender.Text = mainCharacter.Gender;
68          txtSexuality.Text = mainCharacter.Sexuality;
69          txtHairColour.Text = mainCharacter.HairColour;
70          txtEyeColour.Text = mainCharacter.EyeColour;
71      }
72
73      private void panel3_Paint(object sender, PaintEventArgs e)
74      {
75
76      }
77
78      private void btnLoadGame_Click(object sender, EventArgs e)
79      {
80
81          //Read data in from the file and deserialize it into the objects
82          MainCharacter mainCharacter = JsonConvert.DeserializeObject<
    ↪ MainCharacter>(File.ReadAllText(filepath + "\\mainCharacter.json"));
83          Event[] eventArray = JsonConvert.DeserializeObject<Event[]>(File
    ↪ .ReadAllText(filepath + "\\eventArray.json"));
84          ControlClass controlClass = JsonConvert.DeserializeObject<
    ↪ ControlClass>(File.ReadAllText(filepath + "\\controlClass.json"));
85          GenericFamilyMember[] familyArray = JsonConvert.
    ↪ DeserializeObject<GenericFamilyMember[]>(File.ReadAllText(filepath + "
    ↪ \\familyArray.json"));
86          Score mainCharacterScores = JsonConvert.DeserializeObject<Score
    ↪ >(File.ReadAllText(filepath + "\\mainCharacterScores.json"));
87          Partner partner = JsonConvert.DeserializeObject<Partner>(File.
    ↪ ReadAllText(filepath + "\\partner.json"));
88
89          controlClass.Filepath = filepath;
90
91          //MessageBox.Show(mainCharacterScores.LifeScore.ToString());
92
93          frmMainGameScreen fMainGameScreen = new frmMainGameScreen(
94              mainCharacter,
95              eventArray,
96              controlClass,
97              familyArray,
98              mainCharacterScores,
99              partner
100             ); //create new frmMainGameScreen with parameter
    ↪ mainCharacter
101         fMainGameScreen.Show(); //show new main game screen.
102         this.Hide(); //hide this form
103         //fLoadingBar.Hide();
```

```
104         }
105
106         private void btnHome_Click(object sender, EventArgs e)
107         {
108             frmMainMenu fMainMenu = new frmMainMenu();
109             fMainMenu.Show();
110             this.Hide();
111         }
112     }
113 }
```

Listing A.11: FILE/frmShowSavedGames.cs

[FILE]frmShowSavedGames.Designer.cs

```
1
2 namespace A_level_Computer_Science_Project
3 {
4     partial class frmShowSavedGames
5     {
6         /// <summary>
7         /// Required designer variable.
8         /// </summary>
9         private System.ComponentModel.IContainer components = null;
10
11         /// <summary>
12         /// Clean up any resources being used.
13         /// </summary>
14         /// <param name="disposing">true if managed resources should be
    ↪ disposed; otherwise, false.</param>
15         protected override void Dispose(bool disposing)
16         {
17             if (disposing && (components != null))
18             {
19                 components.Dispose();
20             }
21             base.Dispose(disposing);
22         }
23
24         #region Windows Form Designer generated code
25
26         /// <summary>
27         /// Required method for Designer support - do not modify
28         /// the contents of this method with the code editor.
29         /// </summary>
30         private void InitializeComponent()
31         {
32             this.components = new System.ComponentModel.Container();
33             System.ComponentModel.ComponentResourceManager resources = new
    ↪ System.ComponentModel.ComponentResourceManager(typeof(
    ↪ frmShowSavedGames));
34             this.panel1 = new System.Windows.Forms.Panel();
35             this.btnHome = new System.Windows.Forms.Button();
36             this.pictureBox1 = new System.Windows.Forms.PictureBox();
37             this.panel2 = new System.Windows.Forms.Panel();
38             this.btnLoad = new System.Windows.Forms.Button();
39             this.btnLoadGame = new System.Windows.Forms.Button();
40             this.txtFirstName = new System.Windows.Forms.TextBox();
41             this.label1 = new System.Windows.Forms.Label();
42             this.txtLastName = new System.Windows.Forms.TextBox();
43             this.label2 = new System.Windows.Forms.Label();
44             this.label3 = new System.Windows.Forms.Label();
```

```
45              this.txtSexuality = new System.Windows.Forms.TextBox();
46              this.label4 = new System.Windows.Forms.Label();
47              this.label5 = new System.Windows.Forms.Label();
48              this.label6 = new System.Windows.Forms.Label();
49              this.panel3 = new System.Windows.Forms.Panel();
50              this.txtEyeColour = new System.Windows.Forms.TextBox();
51              this.txtHairColour = new System.Windows.Forms.TextBox();
52              this.txtGender = new System.Windows.Forms.TextBox();
53              this.label7 = new System.Windows.Forms.Label();
54              this.toolTip1 = new System.Windows.Forms.ToolTip(this.components
    ↪ );
55              this.panel1.SuspendLayout();
56              ((System.ComponentModel.ISupportInitialize)(this.pictureBox1)).
    ↪ BeginInit();
57              this.panel3.SuspendLayout();
58              this.SuspendLayout();
59              //
60              // panel1
61              //
62              this.panel1.BackColor = System.Drawing.Color.FromArgb(((int)(((
    ↪ byte)(31)))), ((int)(((byte)(31)))), ((int)(((byte)(31)))));
63              this.panel1.Controls.Add(this.btnHome);
64              this.panel1.Controls.Add(this.pictureBox1);
65              this.panel1.Controls.Add(this.panel2);
66              this.panel1.Location = new System.Drawing.Point(12, 12);
67              this.panel1.Name = "panel1";
68              this.panel1.Size = new System.Drawing.Size(776, 71);
69              this.panel1.TabIndex = 1;
70              //
71              // btnHome
72              //
73              this.btnHome.BackColor = System.Drawing.Color.FromArgb(((int)(((
    ↪ byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
74              this.btnHome.BackgroundImage = global::
    ↪ A_level_Computer_Science_Project.Properties.Resources.home;
75              this.btnHome.BackgroundImageLayout = System.Windows.Forms.
    ↪ ImageLayout.Zoom;
76              this.btnHome.FlatStyle = System.Windows.Forms.FlatStyle.Popup;
77              this.btnHome.Location = new System.Drawing.Point(699, 0);
78              this.btnHome.Name = "btnHome";
79              this.btnHome.Size = new System.Drawing.Size(77, 71);
80              this.btnHome.TabIndex = 29;
81              this.btnHome.UseVisualStyleBackColor = false;
82              this.btnHome.Click += new System.EventHandler(this.btnHome_Click
    ↪ );
83              //
84              // pictureBox1
85              //
86              this.pictureBox1.BackgroundImage = global::
    ↪ A_level_Computer_Science_Project.Properties.Resources.Logo_cropped;
87              this.pictureBox1.BackgroundImageLayout = System.Windows.Forms.
    ↪ ImageLayout.Zoom;
88              this.pictureBox1.Location = new System.Drawing.Point(161, 0);
89              this.pictureBox1.Name = "pictureBox1";
90              this.pictureBox1.Size = new System.Drawing.Size(449, 71);
91              this.pictureBox1.TabIndex = 3;
92              this.pictureBox1.TabStop = false;
93              //
94              // panel2
95              //
96              this.panel2.Location = new System.Drawing.Point(128, 115);
```

```
97              this.panel2.Name = "panel2";
98              this.panel2.Size = new System.Drawing.Size(200, 100);
99              this.panel2.TabIndex = 2;
100             //
101             // btnLoad
102             //
103             this.btnLoad.BackColor = System.Drawing.Color.FromArgb(((int)(((
    ↪ byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
104             this.btnLoad.BackgroundImage = global::
    ↪ A_level_Computer_Science_Project.Properties.Resources.folder;
105             this.btnLoad.BackgroundImageLayout = System.Windows.Forms.
    ↪ ImageLayout.Zoom;
106             this.btnLoad.FlatStyle = System.Windows.Forms.FlatStyle.Popup;
107             this.btnLoad.Location = new System.Drawing.Point(0, 0);
108             this.btnLoad.Name = "btnLoad";
109             this.btnLoad.Size = new System.Drawing.Size(71, 48);
110             this.btnLoad.TabIndex = 0;
111             this.toolTip1.SetToolTip(this.btnLoad, "Open a file");
112             this.btnLoad.UseVisualStyleBackColor = false;
113             this.btnLoad.Click += new System.EventHandler(this.btnLoad_Click
    ↪ );
114             //
115             // btnLoadGame
116             //
117             this.btnLoadGame.BackColor = System.Drawing.Color.FromArgb(((int
    ↪ )(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
118             this.btnLoadGame.BackgroundImage = global::
    ↪ A_level_Computer_Science_Project.Properties.Resources.import;
119             this.btnLoadGame.BackgroundImageLayout = System.Windows.Forms.
    ↪ ImageLayout.Zoom;
120             this.btnLoadGame.FlatStyle = System.Windows.Forms.FlatStyle.Flat
    ↪ ;
121             this.btnLoadGame.Location = new System.Drawing.Point(67, 0);
122             this.btnLoadGame.Name = "btnLoadGame";
123             this.btnLoadGame.Size = new System.Drawing.Size(78, 48);
124             this.btnLoadGame.TabIndex = 1;
125             this.toolTip1.SetToolTip(this.btnLoadGame, "Import to game");
126             this.btnLoadGame.UseVisualStyleBackColor = false;
127             this.btnLoadGame.Click += new System.EventHandler(this.
    ↪ btnLoadGame_Click);
128             //
129             // txtFirstName
130             //
131             this.txtFirstName.BackColor = System.Drawing.Color.FromArgb(((
    ↪ int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
132             this.txtFirstName.BorderStyle = System.Windows.Forms.BorderStyle
    ↪ .None;
133             this.txtFirstName.ForeColor = System.Drawing.Color.White;
134             this.txtFirstName.Location = new System.Drawing.Point(81, 124);
135             this.txtFirstName.Name = "txtFirstName";
136             this.txtFirstName.ReadOnly = true;
137             this.txtFirstName.Size = new System.Drawing.Size(121, 16);
138             this.txtFirstName.TabIndex = 14;
139             //
140             // label1
141             //
142             this.label1.AutoSize = true;
143             this.label1.ForeColor = System.Drawing.Color.White;
144             this.label1.Location = new System.Drawing.Point(8, 124);
145             this.label1.Name = "label1";
146             this.label1.Size = new System.Drawing.Size(64, 15);
```

```
147             this.label1.TabIndex = 17;
148             this.label1.Text = "First Name";
149             //
150             // txtLastName
151             //
152             this.txtLastName.BackColor = System.Drawing.Color.FromArgb(((int
    ↪ )(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
153             this.txtLastName.BorderStyle = System.Windows.Forms.BorderStyle.
    ↪ None;
154             this.txtLastName.ForeColor = System.Drawing.Color.White;
155             this.txtLastName.Location = new System.Drawing.Point(81, 146);
156             this.txtLastName.Name = "txtLastName";
157             this.txtLastName.ReadOnly = true;
158             this.txtLastName.Size = new System.Drawing.Size(121, 16);
159             this.txtLastName.TabIndex = 15;
160             //
161             // label2
162             //
163             this.label2.AutoSize = true;
164             this.label2.ForeColor = System.Drawing.Color.White;
165             this.label2.Location = new System.Drawing.Point(8, 146);
166             this.label2.Name = "label2";
167             this.label2.Size = new System.Drawing.Size(63, 15);
168             this.label2.TabIndex = 20;
169             this.label2.Text = "Last Name";
170             //
171             // label3
172             //
173             this.label3.AutoSize = true;
174             this.label3.ForeColor = System.Drawing.Color.White;
175             this.label3.Location = new System.Drawing.Point(27, 168);
176             this.label3.Name = "label3";
177             this.label3.Size = new System.Drawing.Size(45, 15);
178             this.label3.TabIndex = 21;
179             this.label3.Text = "Gender";
180             //
181             // txtSexuality
182             //
183             this.txtSexuality.BackColor = System.Drawing.Color.FromArgb(((
    ↪ int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
184             this.txtSexuality.BorderStyle = System.Windows.Forms.BorderStyle
    ↪ .None;
185             this.txtSexuality.ForeColor = System.Drawing.Color.White;
186             this.txtSexuality.Location = new System.Drawing.Point(81, 190);
187             this.txtSexuality.Name = "txtSexuality";
188             this.txtSexuality.ReadOnly = true;
189             this.txtSexuality.Size = new System.Drawing.Size(121, 16);
190             this.txtSexuality.TabIndex = 22;
191             //
192             // label4
193             //
194             this.label4.AutoSize = true;
195             this.label4.ForeColor = System.Drawing.Color.White;
196             this.label4.Location = new System.Drawing.Point(21, 190);
197             this.label4.Name = "label4";
198             this.label4.Size = new System.Drawing.Size(54, 15);
199             this.label4.TabIndex = 23;
200             this.label4.Text = "Sexuality";
201             //
202             // label5
203             //
```

```
204             this.label5.AutoSize = true;
205             this.label5.ForeColor = System.Drawing.Color.White;
206             this.label5.Location = new System.Drawing.Point(10, 214);
207             this.label5.Name = "label5";
208             this.label5.Size = new System.Drawing.Size(66, 15);
209             this.label5.TabIndex = 24;
210             this.label5.Text = "Hair colour";
211             //
212             // label6
213             //
214             this.label6.AutoSize = true;
215             this.label6.ForeColor = System.Drawing.Color.White;
216             this.label6.Location = new System.Drawing.Point(14, 235);
217             this.label6.Name = "label6";
218             this.label6.Size = new System.Drawing.Size(62, 15);
219             this.label6.TabIndex = 25;
220             this.label6.Text = "Eye colour";
221             //
222             // panel3
223             //
224             this.panel3.BackColor = System.Drawing.Color.FromArgb(((int)(((
    ↪ byte)(31)))), ((int)(((byte)(31)))), ((int)(((byte)(31)))));
225             this.panel3.Controls.Add(this.label7);
226             this.panel3.Controls.Add(this.txtEyeColour);
227             this.panel3.Controls.Add(this.txtHairColour);
228             this.panel3.Controls.Add(this.txtGender);
229             this.panel3.Controls.Add(this.label6);
230             this.panel3.Controls.Add(this.label5);
231             this.panel3.Controls.Add(this.label4);
232             this.panel3.Controls.Add(this.txtSexuality);
233             this.panel3.Controls.Add(this.label3);
234             this.panel3.Controls.Add(this.label2);
235             this.panel3.Controls.Add(this.txtLastName);
236             this.panel3.Controls.Add(this.label1);
237             this.panel3.Controls.Add(this.txtFirstName);
238             this.panel3.Controls.Add(this.btnLoadGame);
239             this.panel3.Controls.Add(this.btnLoad);
240             this.panel3.Location = new System.Drawing.Point(12, 89);
241             this.panel3.Name = "panel3";
242             this.panel3.Size = new System.Drawing.Size(776, 349);
243             this.panel3.TabIndex = 2;
244             this.panel3.Paint += new System.Windows.Forms.PaintEventHandler(
    ↪ this.panel3_Paint);
245             //
246             // txtEyeColour
247             //
248             this.txtEyeColour.BackColor = System.Drawing.Color.FromArgb(((
    ↪ int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
249             this.txtEyeColour.BorderStyle = System.Windows.Forms.BorderStyle
    ↪ .None;
250             this.txtEyeColour.ForeColor = System.Drawing.Color.White;
251             this.txtEyeColour.Location = new System.Drawing.Point(82, 234);
252             this.txtEyeColour.Name = "txtEyeColour";
253             this.txtEyeColour.ReadOnly = true;
254             this.txtEyeColour.Size = new System.Drawing.Size(121, 16);
255             this.txtEyeColour.TabIndex = 28;
256             //
257             // txtHairColour
258             //
259             this.txtHairColour.BackColor = System.Drawing.Color.FromArgb(((
    ↪ int)(((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
```

```
260             this.txtHairColour.BorderStyle = System.Windows.Forms.
    ↪ BorderStyle.None;
261             this.txtHairColour.ForeColor = System.Drawing.Color.White;
262             this.txtHairColour.Location = new System.Drawing.Point(82, 212);
263             this.txtHairColour.Name = "txtHairColour";
264             this.txtHairColour.ReadOnly = true;
265             this.txtHairColour.Size = new System.Drawing.Size(121, 16);
266             this.txtHairColour.TabIndex = 27;
267             //
268             // txtGender
269             //
270             this.txtGender.BackColor = System.Drawing.Color.FromArgb(((int)
    ↪ (((byte)(102)))), ((int)(((byte)(102)))), ((int)(((byte)(102)))));
271             this.txtGender.BorderStyle = System.Windows.Forms.BorderStyle.
    ↪ None;
272             this.txtGender.ForeColor = System.Drawing.Color.White;
273             this.txtGender.Location = new System.Drawing.Point(81, 168);
274             this.txtGender.Name = "txtGender";
275             this.txtGender.ReadOnly = true;
276             this.txtGender.Size = new System.Drawing.Size(121, 16);
277             this.txtGender.TabIndex = 26;
278             //
279             // label7
280             //
281             this.label7.AutoSize = true;
282             this.label7.Font = new System.Drawing.Font("Segoe UI", 15.75F,
    ↪ System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point);
283             this.label7.ForeColor = System.Drawing.Color.White;
284             this.label7.Location = new System.Drawing.Point(27, 82);
285             this.label7.Name = "label7";
286             this.label7.Size = new System.Drawing.Size(162, 30);
287             this.label7.TabIndex = 29;
288             this.label7.Text = "Main Character";
289             //
290             // frmShowSavedGames
291             //
292             this.AutoScaleDimensions = new System.Drawing.SizeF(7F, 15F);
293             this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
294             this.BackColor = System.Drawing.Color.FromArgb(((int)(((byte)
    ↪ (31)))), ((int)(((byte)(31)))), ((int)(((byte)(31)))));
295             this.ClientSize = new System.Drawing.Size(800, 450);
296             this.Controls.Add(this.panel3);
297             this.Controls.Add(this.panel1);
298             this.Icon = ((System.Drawing.Icon)(resources.GetObject("$this.
    ↪ Icon")));
299             this.Name = "frmShowSavedGames";
300             this.Text = "Load Saved Game";
301             this.panel1.ResumeLayout(false);
302             ((System.ComponentModel.ISupportInitialize)(this.pictureBox1)).
    ↪ EndInit();
303             this.panel3.ResumeLayout(false);
304             this.panel3.PerformLayout();
305             this.ResumeLayout(false);
306 
307         }
308 
309         #endregion
310         private System.Windows.Forms.Panel panel1;
311         private System.Windows.Forms.PictureBox pictureBox1;
312         private System.Windows.Forms.Panel panel2;
313         private System.Windows.Forms.Button btnLoad;
```

```
        private System.Windows.Forms.Button btnLoadGame;
        private System.Windows.Forms.TextBox txtFirstName;
        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.TextBox txtLastName;
        private System.Windows.Forms.Label label2;
        private System.Windows.Forms.Label label3;
        private System.Windows.Forms.TextBox txtSexuality;
        private System.Windows.Forms.Label label4;
        private System.Windows.Forms.Label label5;
        private System.Windows.Forms.Label label6;
        private System.Windows.Forms.Panel panel3;
        private System.Windows.Forms.TextBox txtEyeColour;
        private System.Windows.Forms.TextBox txtHairColour;
        private System.Windows.Forms.TextBox txtGender;
        private System.Windows.Forms.Button btnHome;
        private System.Windows.Forms.Label label7;
        private System.Windows.Forms.ToolTip toolTip1;
    }
}
```

Listing A.12: FILE/frmShowSavedGames.Designer.cs

A.2 Classes

A.2.1 Control Class

[FILE]ControlClass.cs

```
using System;
using System.Collections.Generic;
using System.Text;

namespace A_level_Computer_Science_Project
{
    public class ControlClass
    {
        //private attributes
        private DateTime inGameDate;
        private int numberOfEvents;
        private int nextEvent;
        private string filepath;
        private int numberOfFamily;
        private int nextFamily;
        private string nameOfCurrentSchool;
        private int currentSchoolYear;
        private int eduScorePlus;
        private int happinessScorePlus;


        //get and sets
        //GENERIC GAME STUFF----
        public DateTime InGameDate { get; set; }
        public string Filepath { get; set; }
        //EVENTS-------
        public int NumberOfEvents { get; set; }
        public int NextEvent { get; set; }
        //FAMILY--------
        public int NumberOfFamily { get; set; }
        public int NextFamily { get; set; }
        //EDUCATION------
        public string NameOfCurrentSchool { get; set; }
```

```
34          public int CurrentSchoolYear { get; set; }
35          public int EduScorePlus { get; set; }
36
37          //HAPPINESS SCORE-----
38          public int HappinessScorePlus { get; set; }
39
40      }
41  }
```

Listing A.13: FILE/controlClass.cs

A.2.2 Detailed Character

[FILE]DetailedCharacter.cs

```
1   using System;
2   using System.Collections.Generic;
3   using System.Text;
4
5   namespace A_level_Computer_Science_Project
6   {
7       public class DetailedCharacter : Person
8       {
9           Random rand = new Random();
10
11          private bool inEducation;
12          private string educationSituation;
13          private bool inJob;
14          private string hairColour;
15          private string eyeColour;
16
17          //get and set
18          public bool InEducation { get; set; }
19          public string EducationSituation { get; set; }
20          public bool InJob { get; set; }
21          public string HairColour { get; set; }
22          public string EyeColour { get; set; }
23
24
25
26      }
27  }
```

Listing A.14: FILE/DetailedCharacter.cs

A.2.3 Event

[FILE]Event.cs

```
1   using System;
2   using System.Collections.Generic;
3   using System.Text;
4
5   namespace A_level_Computer_Science_Project
6   {
7       public class Event
8       {
9           private DateTime dateHappened;
10          private string description;
11          private string category;
12
13          //get and set
14          public DateTime DateHappened { get; set; }
```

```
        public string Description { get; set; }
        public string Category { get; set; }
    }
}
```

Listing A.15: FILE/Event.cs

A.2.4 Generic Family Member

[FILE]GenericFamilyMember.cs

```
using System;
using System.Collections.Generic;
using System.Text;


namespace A_level_Computer_Science_Project
{
    public class GenericFamilyMember : Person
    {


        private string relationshipToMain;

        //get and set
        public string RelationshipToMain { get; set; }


        //constructor
        public GenericFamilyMember populateGenericFamilyMember(int
    ↪ charNumber, MainCharacter mainCharacter)
        {
            //construct here
            //go through a swtich statement to set the character based on
    ↪ what their relationship to the main is.
            //setup random number generation
            Random rnd = new Random();
            GenericFamilyMember familyMember = new GenericFamilyMember();
            switch (charNumber)
            {
                case 0:
                    //Dad to main character
                    RelationshipToMain = "Father";
                    FirstName = genMFN();
                    LastName = mainCharacter.LastName;
                    Gender = "Male";
                    Sexuality = "Straight";
                    DateOfBirth = genParentAge(mainCharacter.DateOfBirth);
                    LivingStatus = true;
                    InRelationship = true;
                    Age = calcAge(DateOfBirth);
                    break;
                case 1:
                    //mum to main character
                    RelationshipToMain = "Mother";
                    FirstName = genFFN();
                    switch (rnd.Next(1, 3))
                    {
                        case 1:
                            //same last name as father and main character
                            LastName = mainCharacter.LastName;
                            break;
```

```
                        case 2:
                        default:
                            //different last name to father and main
    ↪ character - need to randomGen this
                            LastName = genLN();
                            break;
                    }
                    Gender = "Female";
                    Sexuality = "Straight";
                    DateOfBirth = genParentAge(mainCharacter.DateOfBirth);
                    LivingStatus = true;
                    InRelationship = true;
                    Age = calcAge(DateOfBirth);
                    break;
                default:
                    FirstName = null;
                    break;
            }
            return familyMember;
        }

        static DateTime genParentAge(DateTime mcDOB)
        {
            //need to generate a date between 16 and 30 years ago from main
    ↪ character birthday.
            DateTime dateOfBirth;
            Random rand = new Random();
            int randomNumber = rand.Next(5845, 10957);
            dateOfBirth = mcDOB.AddDays(-randomNumber);


            return dateOfBirth;
        }



    }
}
```

Listing A.16: FILE/GenericFamilyMember.cs

A.2.5 Job

[FILE]Job.cs

```
using System;
using System.Collections.Generic;
using System.Text;

namespace A_level_Computer_Science_Project
{
    public class Job
    {
        private string jobTitle;
        private int salary;
        private int band;

        public string JobTitle { get; set; }
        public int Salary { get; set; }
        public int Band { get; set; }

        public Job generateJob (int band)
```

```
18          {
19              string[] band1Job = { "Cleaner", "Wearhouse opperative", "
    ↪ Grocery picker", "Sahara delivery driver", "Burger flipper", "Hospital
    ↪  porter", "Community care worker", "Care assistant", "Handyperson", "
    ↪ Nursery assistant" };
20              string[] band2Job = { "College security guard", "Pet groomer", "
    ↪ Engineer", "Secretary", "Line cook", "Plumber", "Cleaning company
    ↪ owner", "HR advisor", "Vehicle Technician", "Tanker driver" };
21              string[] band3Jobs = { "Supervisor", "Senior Vehicle Technician"
    ↪ , "Head of Resources", "Product manager", "Web developer", "Comms
    ↪ manager", "Software engineer", "Project manager", "Site manager", "
    ↪ Sales Director" };
22              string[] band4Jobs = { "Store manager", "Finance Manager", "Site
    ↪  supervisor", "Nursing home manager", "IT Solutions Architect", "
    ↪ Senior API Developer", "Electrical Installation Lecturer", "Matron", "
    ↪ Defects manager", "Family Lawyer" };
23              string[] band5Jobs = { "CEO", "Retail park manager", "Hotel
    ↪ manager", "Specialty doctor", "User interface designer", "SQL
    ↪ Developer", "DevOps engineer", "Senior Software consultant", "
    ↪ Executive head teacher", "Hospital director" };
24              Job newJob = new Job();
25              Random rnd = new Random();
26              switch (band)
27              {
28                  case 1:
29                      Salary = 17000;
30                      Band = 1;
31                      JobTitle = band1Job[rnd.Next(1, 9)];
32                      break;
33                  case 2:
34                      Salary = 34000;
35                      Band = 2;
36                      JobTitle = band2Job[rnd.Next(1, 9)];
37                      break;
38                  case 3:
39                      Salary = 51000;
40                      Band = 3;
41                      jobTitle = band3Jobs[rnd.Next(1, 9)];
42                      break;
43
44                  case 4:
45                      Salary = 68000;
46                      Band = 4;
47                      jobTitle = band4Jobs[rnd.Next(1, 9)];
48                      break;
49                  case 5:
50                      Salary = 150000;
51                      Band = 5;
52                      jobTitle = band5Jobs[rnd.Next(1, 9)];
53                      break;
54              }
55              return newJob;
56          }
57      }
58  }
```

Listing A.17: FILE/Job.cs

A.2.6 Main Character

[FILE]MainCharacter.cs

```
using System;
using System.Collections.Generic;
using System.Text;

namespace A_level_Computer_Science_Project
{
    public class MainCharacter : DetailedCharacter
    {
        //attributes

        private int expenses;
        private int bankBalance;
        private int jobBand;
        private string jobTitle;
        private int jobSalary;
        private int jobCurrentDuration;
        private int salary;

        //get and set
        public int Expenses { get; set; }
        public int BankBalance { get; set; }
        public int JobBand { get; set; }
        public string JobTitle { get; set; }
        public int JobSalary { get; set; }
        public int JobCurrentDuration { get; set; }
    }
}
```

Listing A.18: FILE/MainCharacter.cs

A.2.7 Partner

[FILE]Partner.cs

```
using System;
using System.Collections.Generic;
using System.Text;

namespace A_level_Computer_Science_Project
{
    public class Partner : DetailedCharacter
    {
        Random rnd = new Random();
        private string whereMet;
        private DateTime whenMet;

        //get and set
        public string WhereMet { get; set; }
        public DateTime WhenMet { get; set; }

        public Partner generateDetailedChar(string gender, string sexuality,
    ↪  DateTime today)
        {
            Partner potential = new Partner();

            if (gender == "Female")
            {
                if (sexuality == "Straight")
                {
                    //generate a man
                    FirstName = genMFN();
                    LastName = genLN();
```

```
                    Sexuality = "Straight";
                    Gender = "Male";
                    LivingStatus = true;
                    DateOfBirth = today.AddDays(-rnd.Next(5840, 14600));
                    Age = calcAge(DateOfBirth);
                }else if (sexuality == "Homosexual")
                {
                    //generate woman
                    FirstName = genFFN();
                    LastName = genLN();
                    Sexuality = "Homosexual";
                    Gender = "Female";
                    LivingStatus = true;
                    DateOfBirth = today.AddDays(-rnd.Next(5840, 14600));
                    Age = calcAge(DateOfBirth);
                }
            }
            if (gender == "Male")
            {
                if (sexuality == "Homosexual")
                {
                    //generate a man
                    FirstName = genMFN();
                    LastName = genLN();
                    Sexuality = "Homosexual";
                    Gender = "Male";
                    LivingStatus = true;
                    DateOfBirth = today.AddDays(-rnd.Next(5840, 14600));
                    Age = calcAge(DateOfBirth);
                }
                else if (sexuality == "Straight")
                {
                    //generate woman
                    FirstName = genFFN();
                    LastName = genLN();
                    Sexuality = "Straight";
                    Gender = "Female";
                    LivingStatus = true;
                    DateOfBirth = today.AddDays(-rnd.Next(5840, 14600));
                    Age = calcAge(DateOfBirth);
                }
            }


            return potential;
        }

    }
}
```

Listing A.19: FILE/Partner.cs

A.2.8 Person

[FILE]Person.cs

```
using System;
using System.Collections.Generic;
using System.Text;

namespace A_level_Computer_Science_Project
{
```

```
7     public class Person
8     {
9         //Set all private attributes for the class
10        private string firstName;
11        private string lastName;
12        private string gender;
13        private string sexuality;
14        private int age;
15        private bool livingStatus;
16        private DateTime dateOfBirth;
17        private DateTime dateOfDeath;
18        private string reasonForDeath;
19        private bool inRelationship;
20        //get and set methods (nb. use uppercase first character for
    ↪ accessable get/set)
21        public string FirstName { get; set; }
22        public string LastName { get; set; }
23        public string Gender { get; set; }
24        public string Sexuality { get; set; }
25        public int Age { get; set; }
26        public bool LivingStatus { get; set; }
27        public DateTime DateOfBirth { get; set; }
28        public DateTime DateOfDeath { get; set; }
29        public string ReasonForDeath { get; set; }
30        public bool InRelationship { get; set; }
31
32
33        //functions used in constructors
34        public static string genLN()
35        {
36            //Generate female first name
37            Random rnd = new Random();
38            string[] name = { "Smith", "Brown", "Wilson", "Thomson", "
    ↪ Robertson", "Campbell", "Stewart", "Macdonald", "Murray", "Reid", "
    ↪ Taylor", "Clark", "Mitchell", "Ross", "Watson", "Miller", "Gray", "
    ↪ Simpson", "Duncan", "Bell", "Grant", "Mackenzie", "Allan", "Wood", "
    ↪ Muir", "Watt", "King", "Bruce", "Boyle", "Douglas" };
39            string returnName = name[rnd.Next(1, 29)];
40            return returnName;
41        }
42        public static string genMFN()
43        {
44            //generate male first name
45            Random rnd = new Random();
46            string[] name = { "Liam", "Noah", "Oliver", "Elijah", "William",
    ↪  "James", "Benjamin", "Ben", "Lucas", "Henry", "Alex", "Ethan", "
    ↪ Daniel", "Sebstian", "Jack", "Matt", "John", "Joe", "David", "Josh", "
    ↪ Julien", "Leo", "Isaac", "Thomas", "Max", "Andy", "Phill", "Harvey", "
    ↪ Ryan" };
47            string returnName = name[rnd.Next(1, 29)];
48            return returnName;
49        }
50
51        public static string genFFN()
52        {
53            //Generate female first name
54            Random rnd = new Random();
55            string[] name = { "Olivia", "Sophia", "Maria", "Mia", "Evelyn",
    ↪ "Jess", "Ella", "Zoe", "Jemma", "Gemma", "Emily", "Nuala", "Maggie", "
    ↪ Ciara", "Scarlett", "Layla", "Chloe", "Ellie", "Hazel", "Lucy", "Niamh
    ↪ ", "Kat", "Victoria", "Lily", "Hannah", "Chloe", "Lara", "Bella", "
```

```
    ↪ Ruby" };
            string returnName = name[rnd.Next(1, 29)];
            return returnName;
        }

        public int calcAge(DateTime input)
        {
            int difference = (DateTime.Today.Date - input.Date).Days;
            float toround = (difference / 365);
            int age = ((int)toround);

            return age;
        }
    }

}
```

Listing A.20: FILE/Person.cs

A.2.9 Score

[FILE]Score.cs

```
using System;
using System.Collections.Generic;
using System.Text;

namespace A_level_Computer_Science_Project
{
    public class Score
    {
        private int educationScore;
        private int jobScore;
        private int happinessScore;
        private int medicalScore;
        private int lifeScore;

        public int EducationScore { get; set; }
        public int JobScore { get; set; }
        public int HappinessScore { get; set; }
        public int MedicalScore { get; set; }
        public int LifeScore { get; set; }
    }
}
```

Listing A.21: FILE/Score.cs

A.3 Supporting Files

A.3.1 Program

[FILE]Program.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace A_level_Computer_Science_Project
{
    static class Program
```

```
10      {
11          /// <summary>
12          ///  The main entry point for the application.
13          /// </summary>
14          [STAThread]
15          static void Main()
16          {
17              Application.SetHighDpiMode(HighDpiMode.SystemAware);
18              //Application.EnableVisualStyles();
19              Application.SetCompatibleTextRenderingDefault(false);
20              Application.Run(new frmMainMenu());
21
22          }
23      }
24
25  }
```

Listing A.22: FILE/Program.cs

Appendix B

Example Game Saved Data

This is the saved game data produced from the game in video FL2.
NB: These files have been passed through a JSON File Formatter to improve the readability of the data. The game saves the data as a single line without any formatting etc.

B.1 Files

B.1.1 Control Class

[FILE]controlClass.json

```
{
    "InGameDate":"2043-02-18T00:00:00+00:00",
    "Filepath":"C:\\Users\\thoma\\Documents\\Virtual-Life Game Data",
    "NumberOfEvents":1000,
    "NextEvent":42,
    "NumberOfFamily":30,
    "NextFamily":2,
    "NameOfCurrentSchool":"Eastview Institute",
    "CurrentSchoolYear":13,
    "EduScorePlus":100,
    "HappinessScorePlus":25
}
```

Listing B.1: FILE/controlclass.json

B.1.2 Event Array

The file below has been altered further from the formatted file (described above) and original file which the game produces. The game generates 1000 elements to the `eventArray`, all of these, no matter if they are null get written out to the file, resulting in a file with 5002 lines (which takes up about 95 pages in this document). From a space efficiency point of view, the file has been trimmed, only showing 1 null event. There are 957 more in the original file.

[FILE]eventArray.json

```
[
    {
        "DateHappened":"2022-02-18T00:00:00+00:00",
        "Description":"You were born to parents Leo and Lara",
        "Category":"Life Event"
    },
    {
        "DateHappened":"2023-02-18T00:00:00+00:00",
        "Description":"You turned 1",
        "Category":"Main character birthday"
```

```
    },
    {
        "DateHappened":"2024-02-18T00:00:00+00:00",
        "Description":"You turned 2",
        "Category":"Main character birthday"
    },
    {
        "DateHappened":"2025-02-18T00:00:00+00:00",
        "Description":"You turned 3",
        "Category":"Main character birthday"
    },
    {
        "DateHappened":"2026-02-18T00:00:00+00:00",
        "Description":"You turned 4",
        "Category":"Main character birthday"
    },
    {
        "DateHappened":"2026-02-18T00:00:00+00:00",
        "Description":"Leo (your father) died due to a heart attack",
        "Category":"Death"
    },
    {
        "DateHappened":"2026-09-01T00:00:00",
        "Description":"Start infant school in Reception. Your parents chose
↪ Westview Institute for you.",
        "Category":"Education"
    },
    {
        "DateHappened":"2027-02-18T00:00:00+00:00",
        "Description":"You turned 5",
        "Category":"Main character birthday"
    },
    {
        "DateHappened":"2027-09-01T00:00:00",
        "Description":"Move into Year 1 at infant school.",
        "Category":"Education"
    },
    {
        "DateHappened":"2028-02-18T00:00:00+00:00",
        "Description":"You turned 6",
        "Category":"Main character birthday"
    },
    {
        "DateHappened":"2028-09-01T00:00:00",
        "Description":"Move into year 2 at infant school. This is your last
↪ year there. Your parents are frantically trying to find a place at
↪ Junior school for you.",
        "Category":"Education"
    },
    {
        "DateHappened":"2029-02-18T00:00:00+00:00",
        "Description":"You turned 7",
        "Category":"Main character birthday"
    },
    {
        "DateHappened":"2029-09-01T00:00:00",
        "Description":"Start year 3 at Junior school. You make lots of
↪ friends very quickly. Your parents managed to find you a place at Lone
↪  Oak all-through.",
        "Category":"Education"
    },
```

```
    {
        "DateHappened":"2030-02-18T00:00:00+00:00",
        "Description":"You turned 8",
        "Category":"Main character birthday"
    },
    {
        "DateHappened":"2030-09-01T00:00:00",
        "Description":"Move into year 4 at Junior School.",
        "Category":"Education"
    },
    {
        "DateHappened":"2031-02-18T00:00:00+00:00",
        "Description":"You turned 9",
        "Category":"Main character birthday"
    },
    {
        "DateHappened":"2031-09-01T00:00:00",
        "Description":"Move into year 5 at Junior School.",
        "Category":"Education"
    },
    {
        "DateHappened":"2032-02-18T00:00:00+00:00",
        "Description":"You turned 10",
        "Category":"Main character birthday"
    },
    {
        "DateHappened":"2032-09-01T00:00:00",
        "Description":"Move into year 6 at Junior School. You have SATS
    ↪ exams at the end of this year.",
        "Category":"Education"
    },
    {
        "DateHappened":"2033-02-18T00:00:00+00:00",
        "Description":"You turned 11",
        "Category":"Main character birthday"
    },
    {
        "DateHappened":"2033-09-01T00:00:00",
        "Description":"Start year 7 at Secondary School. You don't make many
    ↪  friends to begin with. You are lucky enough to be attending Westview
    ↪ Institute, without that persuasive letter your dad sent, this wouldn't
    ↪  have happned.",
        "Category":"Education"
    },
    {
        "DateHappened":"2034-02-18T00:00:00+00:00",
        "Description":"You turned 12",
        "Category":"Main character birthday"
    },
    {
        "DateHappened":"2034-09-01T00:00:00",
        "Description":"Start yer 8 at Secondary School. You now have lots of
    ↪  friends..",
        "Category":"Education"
    },
    {
        "DateHappened":"2035-02-18T00:00:00+00:00",
        "Description":"You turned 13",
        "Category":"Main character birthday"
    },
    {
```

```
        "DateHappened":"2035-09-01T00:00:00",
        "Description":"Start year 9 at Secondary school.",
        "Category":"Education"
    },
    {
        "DateHappened":"2035-02-18T00:00:00+00:00",
        "Description":"Steal a car",
        "Category":"Crime"
    },
    {
        "DateHappened":"2035-02-18T00:00:00+00:00",
        "Description":"Steal a car",
        "Category":"Crime"
    },
    {
        "DateHappened":"2036-02-18T00:00:00+00:00",
        "Description":"You turned 14",
        "Category":"Main character birthday"
    },
    {
        "DateHappened":"2036-09-01T00:00:00",
        "Description":"Start year 10 at Secondary school.",
        "Category":"Education"
    },
    {
        "DateHappened":"2037-02-18T00:00:00+00:00",
        "Description":"You turned 15",
        "Category":"Main character birthday"
    },
    {
        "DateHappened":"2037-09-01T00:00:00",
        "Description":"Start year 11 at Secondary school. You have GCSE
    ↪ exams at the end of the year.",
        "Category":"Education"
    },
    {
        "DateHappened":"2038-02-18T00:00:00+00:00",
        "Description":"You turned 16",
        "Category":"Main character birthday"
    },
    {
        "DateHappened":"2038-09-01T00:00:00",
        "Description":"Start year 12 at College. Your parents wanted you to
    ↪ take English, Maths and Chemistry but you are taking Drama, music and
    ↪ Physics. You're now attending Eastview Institute.",
        "Category":"Education"
    },
    {
        "DateHappened":"2039-02-18T00:00:00+00:00",
        "Description":"You turned 17",
        "Category":"Main character birthday"
    },
    {
        "DateHappened":"2039-09-01T00:00:00",
        "Description":"Start year 13 at College. You have your A-levels at
    ↪ the end of this year.",
        "Category":"Education"
    },
    {
        "DateHappened":"2040-02-18T00:00:00+00:00",
        "Description":"You turned 18",
```

```
        "Category":"Main character birthday"
    },
    {
        "DateHappened":"2041-02-18T00:00:00+00:00",
        "Description":"You turned 19",
        "Category":"Main character birthday"
    },
    {
        "DateHappened":"2041-02-18T00:00:00+00:00",
        "Description":"Lara (your mother) died due to being crushed by a
    ↪ stampede of hungry campers",
        "Category":"Death"
    },
    {
        "DateHappened":"2041-02-18T00:00:00+00:00",
        "Description":"You applied for and got the job: Grocery picker",
        "Category":"Job"
    },
    {
        "DateHappened":"2042-02-18T00:00:00+00:00",
        "Description":"You turned 20",
        "Category":"Main character birthday"
    },
    {
        "DateHappened":"2043-02-18T00:00:00+00:00",
        "Description":"You turned 21",
        "Category":"Main character birthday"
    },
    {
        "DateHappened":"2043-02-18T00:00:00+00:00",
        "Description":"You died due to being strangled by a fez tassel",
        "Category":"Death"
    },
    {
        "DateHappened":"0001-01-01T00:00:00",
        "Description":null,
        "Category":null
    }
]
```

Listing B.2: FILE/eventArray.json

B.1.3 Family Array

In a similar way to the `eventArray` file, the `familyArray` file has also been altered by removing 26 null elements from the array.

[FILE]familyArray.json

```
[
    {
        "RelationshipToMain":"Father",
        "FirstName":"Leo",
        "LastName":"Bruce",
        "Gender":"Male",
        "Sexuality":"Straight",
        "Age":20,
        "LivingStatus":false,
        "DateOfBirth":"2005-11-28T00:00:00+00:00",
        "DateOfDeath":"2026-09-26T00:00:00+01:00",
        "ReasonForDeath":"a heart attack",
```

```
        "InRelationship":true
    },
    {
        "RelationshipToMain":"Mother",
        "FirstName":"Lara",
        "LastName":"Macdonald",
        "Gender":"Female",
        "Sexuality":"Straight",
        "Age":42,
        "LivingStatus":false,
        "DateOfBirth":"1999-02-22T00:00:00+00:00",
        "DateOfDeath":"2041-08-30T00:00:00+01:00",
        "ReasonForDeath":"being crushed by a stampede of hungry campers",
        "InRelationship":true
    },
    {
        "RelationshipToMain":null,
        "FirstName":null,
        "LastName":null,
        "Gender":null,
        "Sexuality":null,
        "Age":0,
        "LivingStatus":false,
        "DateOfBirth":"0001-01-01T00:00:00",
        "DateOfDeath":"0001-01-01T00:00:00",
        "ReasonForDeath":null,
        "InRelationship":false
    }
]
```

Listing B.3: FILE/familyArray.json

B.1.4 Main Character

[FILE]mainCharacter.json

```
{
    "Expenses":0,
    "BankBalance":0,
    "JobBand":1,
    "JobTitle":"Grocery picker",
    "JobSalary":17023,
    "JobCurrentDuration":1,
    "InEducation":true,
    "EducationSituation":null,
    "InJob":true,
    "HairColour":"Brown",
    "EyeColour":"Gray",
    "FirstName":"William",
    "LastName":"Bruce",
    "Gender":"Male",
    "Sexuality":"Straight",
    "Age":21,
    "LivingStatus":false,
    "DateOfBirth":"2022-02-18T00:00:00+00:00",
    "DateOfDeath":"2043-04-10T00:00:00+01:00",
    "ReasonForDeath":"being strangled by a fez tassel",
    "InRelationship":false
}
```

Listing B.4: FILE/mainCharacter.json

B.1.5 Main Character Scores

[FILE]mainCharacterScores.json

```
{
    "EducationScore":780,
    "JobScore":4,
    "HappinessScore":539,
    "MedicalScore":1000,
    "LifeScore":0
}
```

Listing B.5: FILE/mainCharacterScores.json

B.1.6 Partner

[FILE]partner.json

```
{
    "WhereMet":null,
    "WhenMet":"0001-01-01T00:00:00",
    "InEducation":false,
    "EducationSituation":null,
    "InJob":false,
    "HairColour":null,
    "EyeColour":null,
    "FirstName":null,
    "LastName":null,
    "Gender":null,
    "Sexuality":null,
    "Age":0,
    "LivingStatus":false,
    "DateOfBirth":"0001-01-01T00:00:00",
    "DateOfDeath":"0001-01-01T00:00:00",
    "ReasonForDeath":null,
    "InRelationship":false
}
```

Listing B.6: FILE/partner.json

Appendix C

References

C.1 Resources Used For Implementation

Listed below are resources which were used to aid programming during the implementation phase of the project.

- Microsoft C# documentation
- Stack Overflow
- Code Project
- Geeks For Geeks
- Net Informations
- Code Grepper
- W3 Schools
- Dot Net Perls
- C Sharp Corner

C.2 Newtonsoft Package

I used the `Newtonsoft.JSON` package during implementation to aid me with handling JSON files. Listed below are the resources I used to aid me with this.

- Newtonsoft JSON home page
- Serializing JSON documentation
- Deserialize Anonymous Type documentation

Appendix D

Screenshots Of Forms

D.1 Main Menu

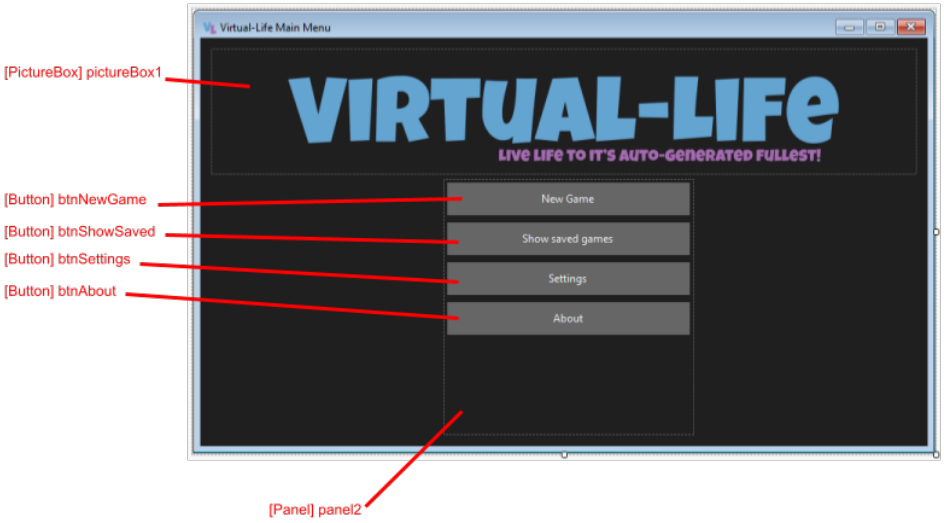


Figure D.1: Annotated main menu form

D.2 New Game

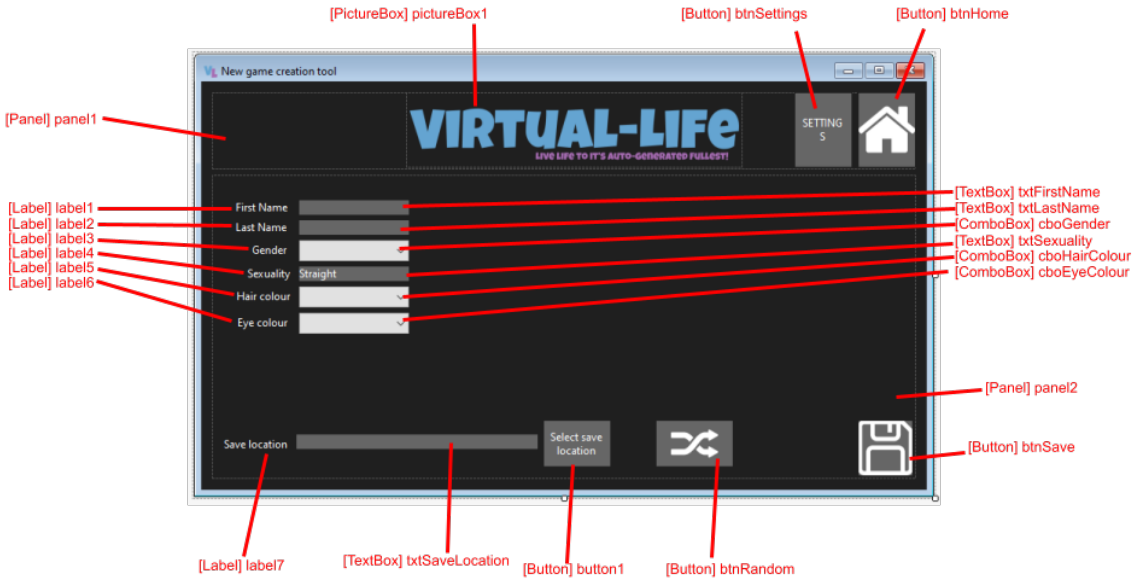


Figure D.2: Annotated new game form

D.3 Show Saved Games

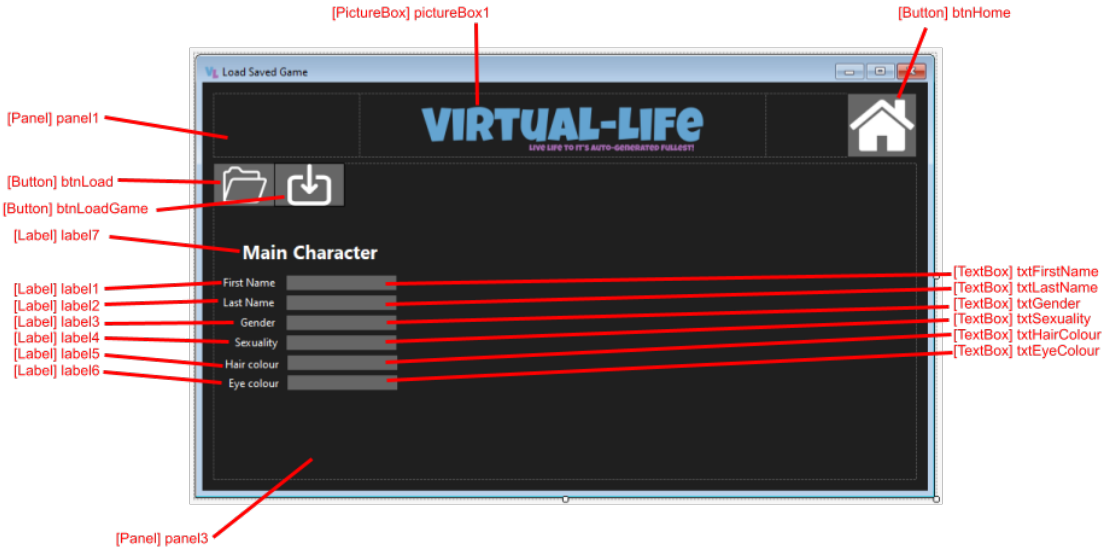


Figure D.3: Annotated show saved games form

D.4 About

Figure D.4: Annotated about form

D.5 Choice Box

Figure D.5: Annotated choice box form

D.6 Main Game Screen

D.6.1 Main Page

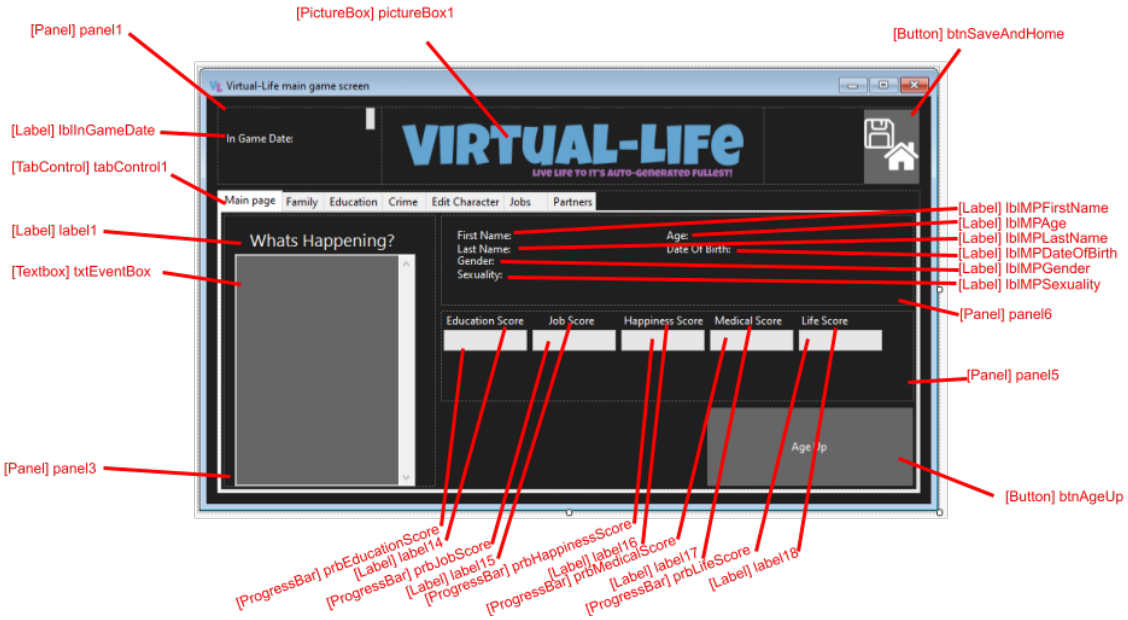


Figure D.6: Annotated main game tab of main game screen form

D.6.2 Crime

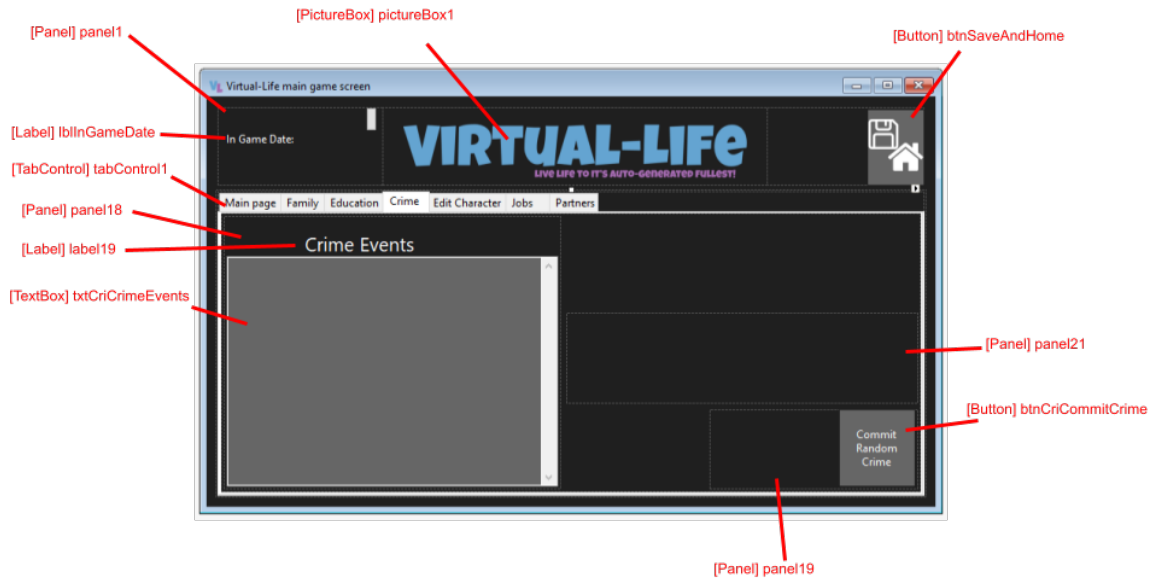


Figure D.7: Annotated crime tab of main game screen form

D.6.3 Edit Character

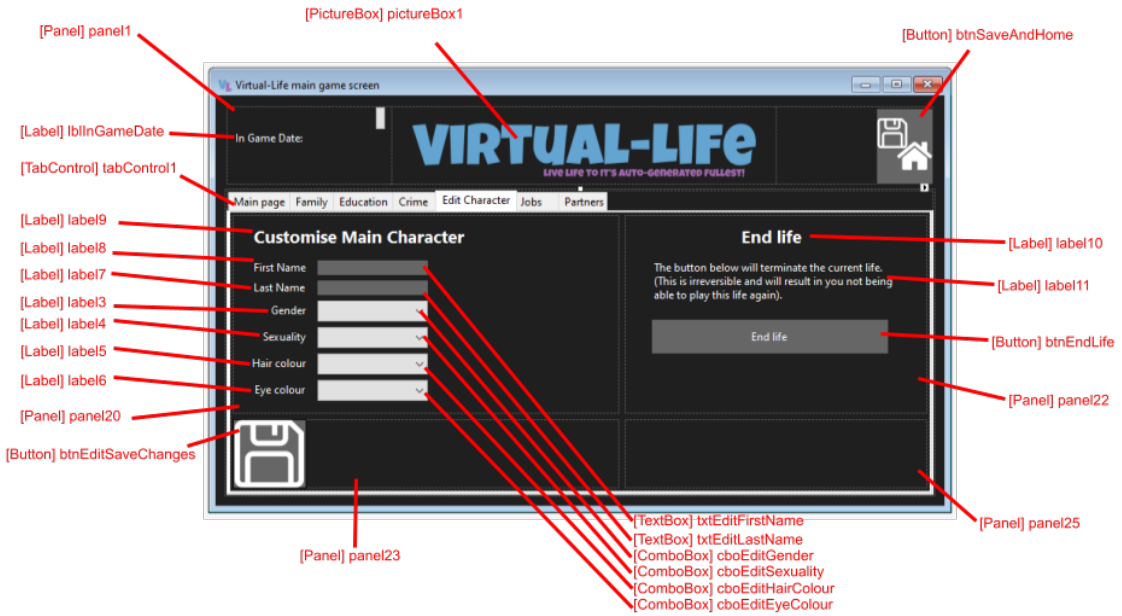


Figure D.8: Annotated edit character tab of main game screen form

D.6.4 Education

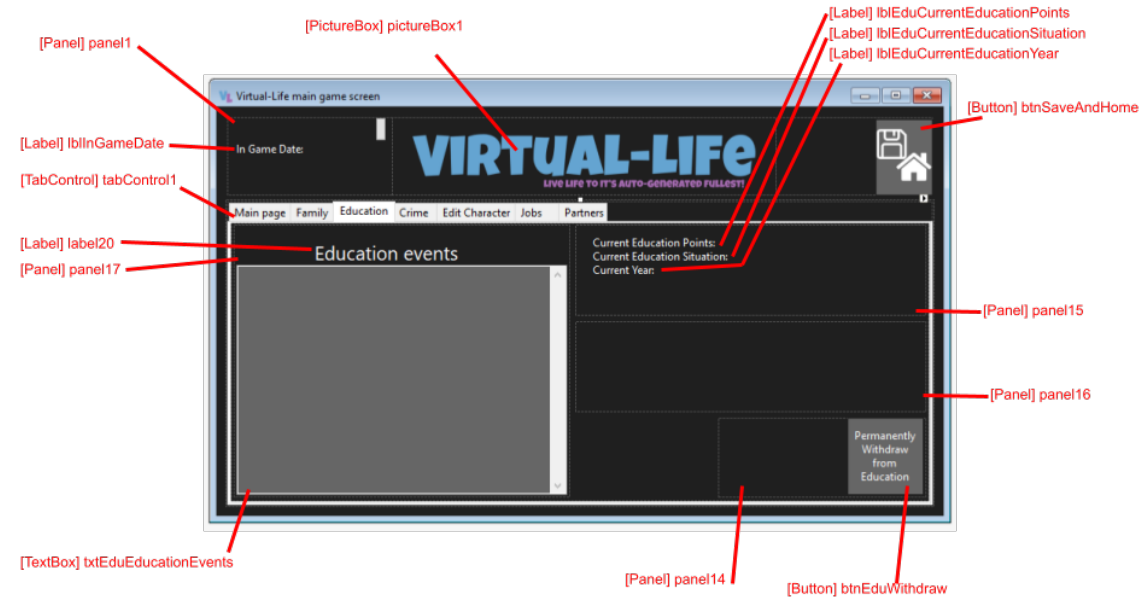


Figure D.9: Annotated education tab of main game screen form

D.6.5 Family

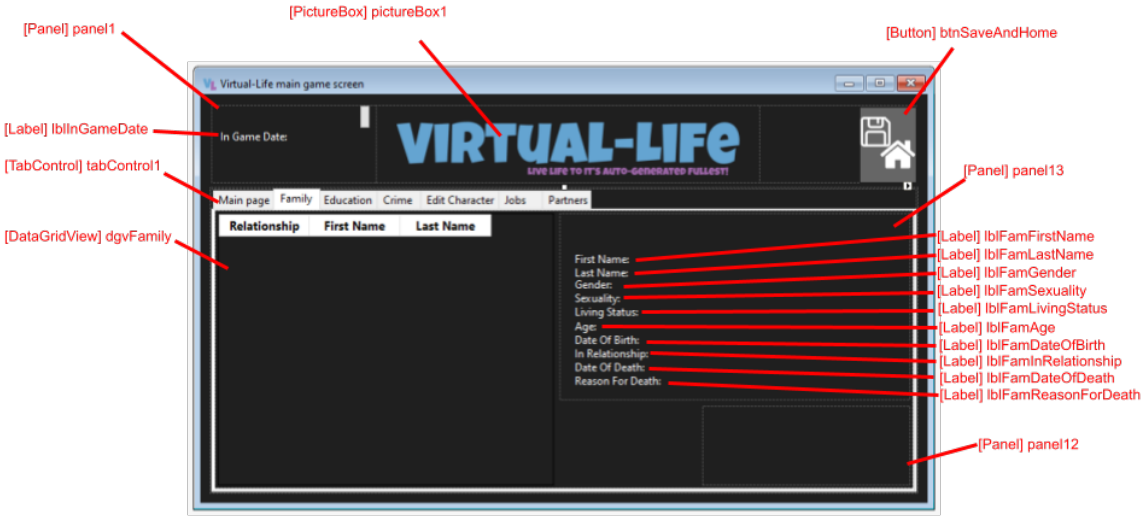


Figure D.10: Annotated family tab of main game screen form

D.6.6 Jobs

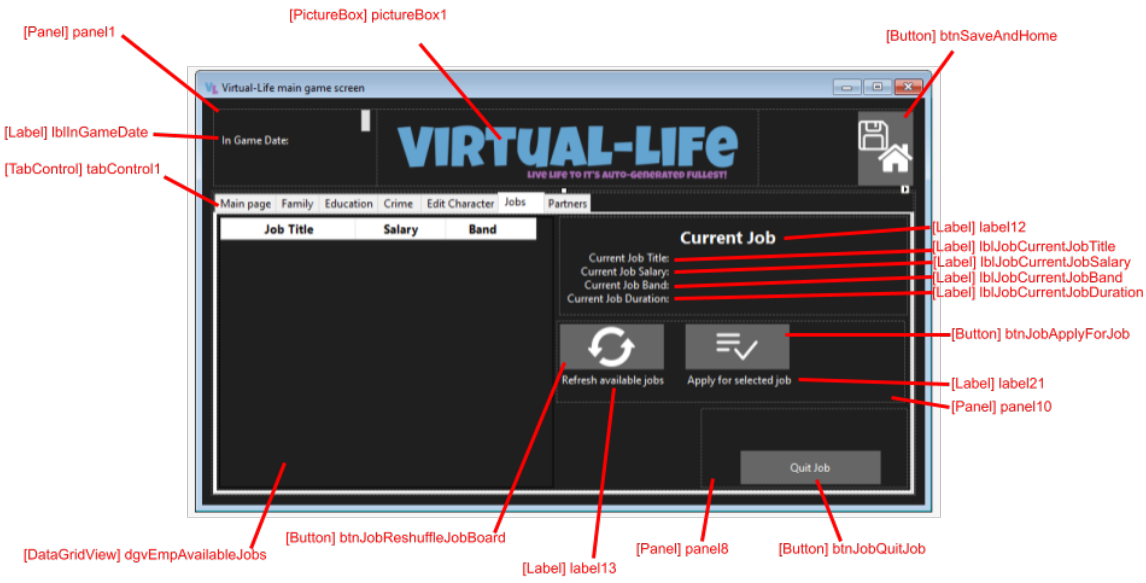


Figure D.11: Annotated jobs tab of main game screen form

D.6.7 Partners

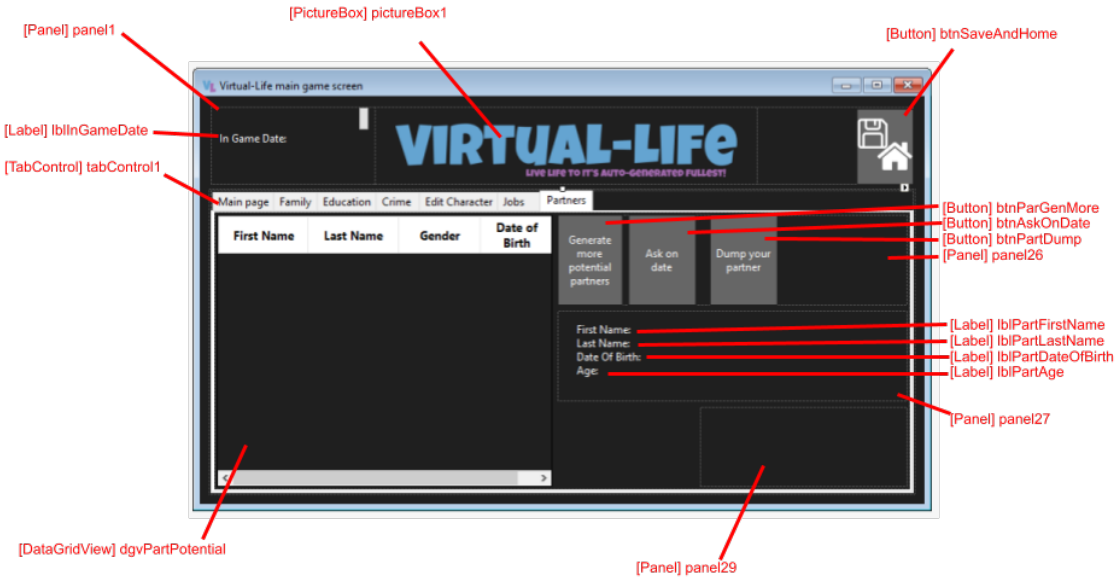


Figure D.12: Annotated partners tab of main game screen form

Appendix E

Full Index

Contents

List of Figures

Listings