

# Software Engineering Principles and Practices FINAL - Sam Kelly

*by Samuel Jackson Kelly*

---

**Submission date:** 28-Apr-2023 08:22PM (UTC+0100)

**Submission ID:** 203940316

**File name:** 41605\_Samuel\_Jackson\_Kelly\_Software\_Engineering\_Principles\_and\_Practices\_FINAL\_-\_Sam\_Kelly\_490085\_1184646553.docx (1.48M)

**Word count:** 4263

**Character count:** 19408

## **Software Engineering Design & Version Control**

Samuel Kelly – Student ID 2017070

Software Principles and Practices

28<sup>th</sup> April 2023

## Table of Contents

|                                                |    |
|------------------------------------------------|----|
| Table of Figures .....                         | 2  |
| Introduction.....                              | 2  |
| Design Evaluation.....                         | 4  |
| Wire Frames and Design.....                    | 4  |
| User stories.....                              | 5  |
| Class Responsibility Collaboration Cards ..... | 9  |
| Software Testing Evaluation.....               | 13 |
| Version Control.....                           | 15 |
| Conclusion .....                               | 17 |
| Appendix .....                                 | 18 |

## Table of Figures

|                                                                                                                             |    |
|-----------------------------------------------------------------------------------------------------------------------------|----|
| Figure 1 Wireframe diagram for the connect 4 web based game .....                                                           | 5  |
| Figure 2 User stories map for a "Player" persona .....                                                                      | 6  |
| Figure 3 User story for updated connect 4 mechanics .....                                                                   | 7  |
| Figure 4 Tutorial instructions user story .....                                                                             | 7  |
| Figure 5 User story for account creation not int he initial release .....                                                   | 8  |
| Figure 6 CRC Cards for the connect 4 project.....                                                                           | 10 |
| Figure 7 CRC Card for the board itself.....                                                                                 | 11 |
| Figure 8 CRC Card for the player .....                                                                                      | 11 |
| Figure 9 Setting name and email in a new GIT repository .....                                                               | 15 |
| Figure 10 Adding new txt files to the GIT repository .....                                                                  | 18 |
| Figure 11 Staging a new file and checking the status of the files being tracked by GIT .....                                | 18 |
| Figure 12 Committing the files and checking to see if the working tree is empty and the<br>commit has been successful ..... | 18 |
| Figure 13 Adding a new file and sending it to the staging area.....                                                         | 19 |
| Figure 14 Adding edits to the new file.....                                                                                 | 19 |
| Figure 15 Committing the new changes to the edited file .....                                                               | 19 |

## Introduction

This report will focus on the design and evaluation of a web based connect 4 game. The game adds some extra rules to the traditional two player game of connect 4. The first of these rules is the ability to capture your opponents counter by trapping it in between two of your own counters. This can either be done vertically, horizontally, or diagonally. The second of the new additions is a mechanic that takes effect when a player has captured three of their opponents counters. Once captured the three counters will be destroyed and all counters above this will fall down into the holes to fill the gaps up.

This model of connect 4 was the basis for the design and evaluation in this report. A number of design choices and features will be discussed as well as some relevant technologies used in industry in order to make software development a smoother and more transparent process. Some user interface design for the web based game will also be discussed within this report.

The use of user stories and class responsibility collaboration cards are standard practice in the workplace with more and more companies adopting the agile principles into their everyday workflow. Agile focuses on collaboration and transparency putting the customers needs first to make sure the product developed is satisfactory. Agile principles make collaboration easier with industry standards to how things should be categorised and listed. This report will demonstrate a few of these principles and show how easy collaboration can be with a few easy to implement steps.

## Design Evaluation

When creating a web based application there are a number of things to be considered. The first of these is the layout. User satisfaction is of the utmost importance, if a user is not having a good time when using a product then they are more likely to leave the website or stop using the product. To best capture a user an interesting and simple user interface was designed and will be covered in the next section.

## Wire Frames and Design

Thinking about the design of a website based game is possibly the most important factor when creating a website. When users first click onto the website the first thing they want to see is the main theme of the website. Although it won't be discussed in detail here a good colour scheme and font are very important to the overall ease of digest of the website. The main focus of the design here will be the wire frame and layout of the website.

By drawing a wireframe diagram for this website the planning and organisation will be far clearer and more transparent. In environments where more than one developer is working on a project making sure that all programmers are on the same page and have the same ultimate goal creates a fluid and stress free environment. Wireframes allow the user experience of the website to be considered from the get go. If a website the website is to be coded and have all functionality added before any developers know the overall feel and design of the web pages it could create some usability issues. Stemming from this a test driven development approach may be used in a case such as this website. The use of the wireframe also allows iteration and testing to be made easier and allows updates to the design be easier seen by all members of a team.

Looking at figure 1 below a simple wireframe diagram is shown. In this diagram the main game layout of the website is shown, as well as a number of early features that would benefit the users. A few options can be seen namely the "Rules and Help" button and the two options for local or online play. Although the app will be a web based application many users may be sitting next to the person they want to play, to make this as easy as possible the option to play from the same device was added. Online play is a feature many users look for in modern day web applications. In order to meet the demand for players allowing an option to

play against a random opponent from the internet will benefit the overall use of the app and keep users on for a longer period of time.

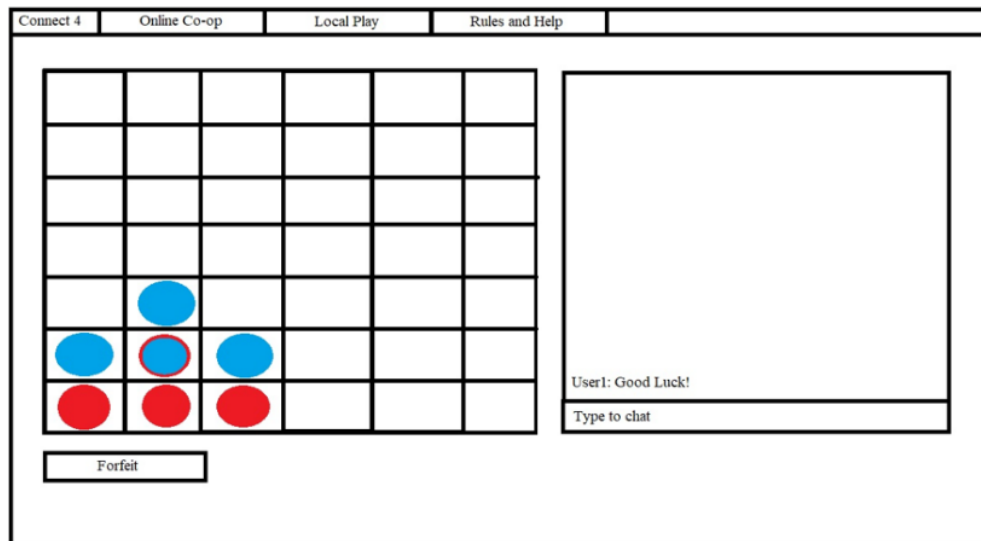


Figure 1 Wireframe diagram for the connect 4 web based game

## User stories

A crucial aspect of modern day software development is the adoption of user stories. Using user stories to develop a program allows the programmers to put what matters first the most. Making sure a product has the user experience being considered first has lead to a widespread improvement in usability of programs. In the past developers were left to their devices when programming and often ended up creating a product that wasn't in line with what a customer may have wanted. By adopting agile principles in the workplace the customers needs are met and the development process is streamlined.

Looking at figures 2 through 5 below the user stories for this website can be seen. Figure 2 shows the overall user story map for what a player of the game may wish to see. An important point to note about the user story map is that the login and account creation options are moved to the first update of the website. Although a lot of web apps and websites ask a user to create an account before they do anything, in this case the account creation is unnecessary and may even turn users away from interacting with the website. A connect 4 type game has no requirement to collect user data or to target ads.

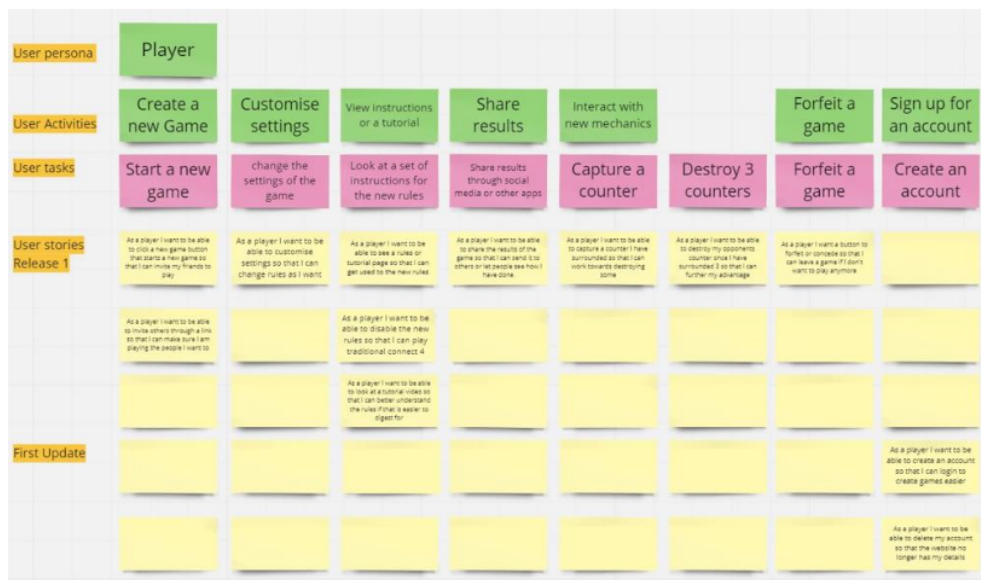


Figure 2 User stories map for a "Player" persona

Taking a closer look at the breakdown of users stories for this website, a few of the new features in the connect 4 game could be difficult to understand given in just text form. To best get around this a tutorial tab was added to the wireframe diagram as seen in figure 1. This tab would include a video explaining the updated rules of the game as well as text explaining the rules, this can be seen in figure 4. The users stories used here highlight important details that increase the ease of use for anyone using the website. They also provide insight into how players might interact with the new rules to connect 4. It is entirely possible a user would go on the website and think the game is normal connect 4, to help avoid this scenario the rules and tutorial section would prove incredibly useful.

The user stories are however, not without their drawbacks being fairly limited in how the user will interact with the website they don't give particular insight into how the user would play the game. The design of the website is considered thoroughly but when thinking about the actual player interaction with the board and updated mechanics there aren't many mentions of this.

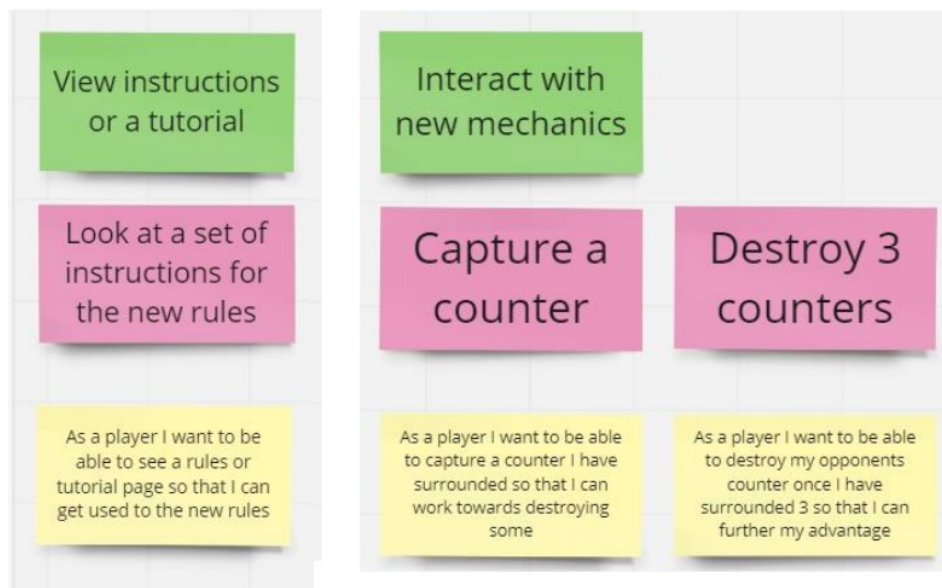


Figure 3 User story for updated connect 4 mechanics

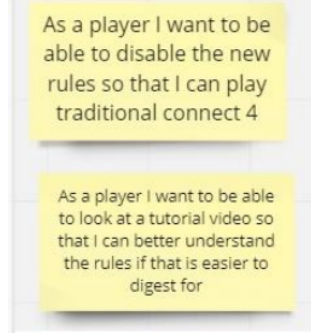


Figure 4 Tutorial instructions user story

Aside from the details of the website briefly mentioned was the addition of a create an account button. Despite not being particularly vital in the first stages of a project it maybe be a feature to consider after the product has been shipped. Continuous updates are a must have in modern programming and no user story map or product is complete without some foresight for the future. Figure 5 shows the intention of adding a create account button to the website once the first update is pushed. This allows users to get used to the website and the game and then at a later date if they wish they can make an account to save and update their progress and possible other features not yet planned out.

The user stories provided here offer a good insight into the initial release of the website and the immediate future. In terms of content and future updates the nature of this

website doesn't require a lot of upkeep and maintenance, there is of course room to provide updates to the ruleset of the connect 4 and even ways for new rules to be added.

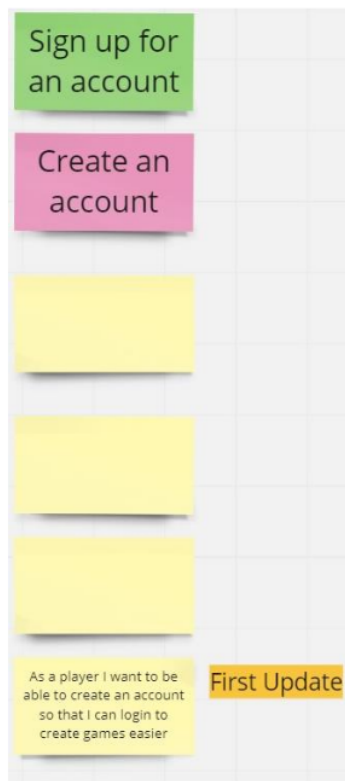


*Figure 5 User story for account creation not in the initial release*

Figure 5 shows the only real first update needed to the website after its initial release and push to market. However, this does not mean that more user stories can be added in the future. One of the main benefits of user stories is that in agile workplaces should a developer or anyone think of a new user story they can simply write it down and put it up on the ideas board. The usefulness of user stories can not be understated in the field of program development.

With all the benefits to user stories there are of course some drawbacks. User stories often lack in depth information on what each feature should do, there is a lot of room for misinterpretations of what each feature should do and often needs some discussion to make sure all members of a team are on the same page. On the same note as this user stories are often only about a single feature for example a login feature and a logout feature would be

two separate user stories. This creates a lot of similar but very slightly different user stories that can add up over the course of a whole project. User stories are also only created by the development team there is not a lot of room for feedback from outside sources when discussing user stories so there is room for some of the features to be unrelated to what the customer wanted in the first place. These stories also do not account for the other requirements in the development process, security is an essential part of any application to make sure that company and user data stays protected. User stories do not account for the security needs related to each feature.

### Class Responsibility Collaboration Cards

User stories are not the only thing to consider Class Responsibility Collaboration cards also known as CRC Cards provide a way for the developers to see which classes need to be coded and where they should be related. CRC Cards are often one of the most useful parts of development in terms of the overall picture it gives the developers. Depending on the size of the project the amount of CRC Cards and their complexity will vary. Being a relatively simple project there are only a few things to consider when thinking about the class relationships. The Class responsibility approach to development has a number of benefits. CRC Cards allow for developers to make easy to understand cards for programs. If one developer sees a CRC Card they will immediately understand the responsibilities of the class and which other classes it may depend on to get it's information. They can be easily modified and updated on the fly unlike user stories where you may have to add a new one every time, CRC Cards can be updated and edited as needed.

Taking a look at a few of the CRC Cards related to this project the responsibilities of the class as well as the collaborators are clearly stated in each card. Figure 6 shows the breakdown of all the CRC Cards used in the planning process of this web based application. In this section a more in depth look will be taken on a few of the key CRC Cards made in the development process. When creating CRC cards for a project it is important to take time and think of every possible need a project may have. One of the benefits of this approach is that working in a team allows all the members to come up with different classes and responsibilities that other members may not have noticed.

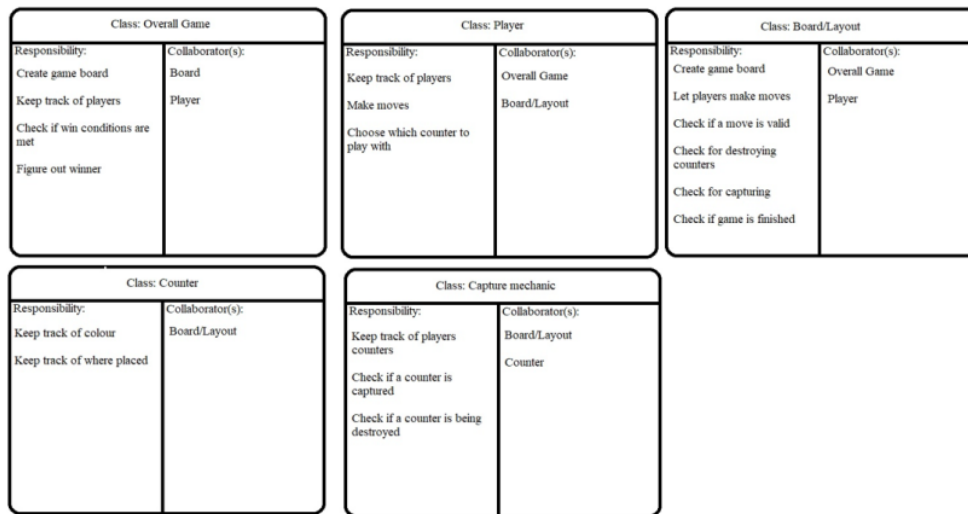


Figure 6 CRC Cards for the connect 4 project

In figure 7 the CRC Card for the board of the game itself is show. Although it may seem trivial there will be other classes that are dependant on the board for how they function a couple examples of this would be the overall function of the game and the players in the game. The game itself is reliant on the board because without the board there are no slots for the counters to go in, this is a similar point for the player without the board itself there are no places for the players to place their counters. A few of the responsibilities of the board are to create the layout at the start of the game, allow the players to place their counters and make sure the move they want to do is valid. It is also the responsibility of the board to make sure all counters being captured and destroyed are happening at the right times.

This card is vital the overall performance and smooth running of the game. In some instances the cards may not be as useful as others, in this case all cards created are needed for the game to operate. If a programmer was to miss out on any of the cards used here then the game itself would not function. One thing these cards fail to cover is the performance of the website. In this case only the operation of the game itself is considered if this project was to be pushed to release the function of the website would also need to be considered

The second CRC card that is important to the operation of the game is the player. If the players moves are not tracked properly then the counters that are captured and destroyed may not be calculated properly. Similarly to the board card the player card has the overall

game and the board listed as collaborators. These cards have plenty of information for any competent developer to make use of when writing a program, they contain the responsibilities of the class which need to be coded in as well as the collaborators who rely on the class. In addition to this the CRC card approach to development allows these cards to be changed as the needs of the project develop.

| Class: Board/Layout           |                  |
|-------------------------------|------------------|
| Responsibility:               | Collaborator(s): |
| Create game board             | Overall Game     |
| Let players make moves        | Player           |
| Check if a move is valid      |                  |
| Check for destroying counters |                  |
| Check for capturing           |                  |
| Check if game is finished     |                  |

Figure 7 CRC Card for the board itself

| Class: Player                     |                  |
|-----------------------------------|------------------|
| Responsibility:                   | Collaborator(s): |
| Keep track of players             | Overall Game     |
| Make moves                        | Board/Layout     |
| Choose which counter to play with |                  |

Figure 8 CRC Card for the player

Despite all of the upside to using CRC cards when developing a project there is quite a few limitations. Not only are they time consuming to make and consider every aspect of a class they are also only useful in an object oriented programming environment, meaning if there are different methods wanting to be used during the development phase then the CRC cards

are no longer useful for that approach. These cards similarly to user stories also lack formal documentation, they do not provide new team members with any information relating to the codebase that the project is being done in which can cause some issues if the new team member is not very experienced in the current code base.

## Software Testing Evaluation



Testing software is an essential part of the process, without **prospering** testing along the path of development there would be no way to tell if a program or application was performing as expect. By making sure regular test are done during the development cycle both the developers and the customer will be up to date on how the application is going. Testing allows developers to see if any functions are not performing as expected and if any changes are being made. When testing software there are two approaches that best suit the needs of the project at different intervals.

White box testing is a testing method that allows the person running the test to have access to the internal workings of the software while they are testing. Someone who might be in this position could be a developer who has just coded a new feature for a website and is looking to make sure it works, they may put the website up on a local live server and click around to make sure everything is working as intended. This type of testing is incredibly useful and often very easy to perform in the development stages. The developers know what is happening with their code and what they want it to do in each case so it allows them to run the code and test as they go. Using this method allows developers to avoid a number of problems they could run into if they are to program all the code at once and then test it at the end. This is similar to the waterfall methodology which is often not very productive in most cases and part of the reason the agile principles have become so popular in modern times.

This method of testing is often used in unit testing or integration testing as mentioned. The developer will test a specific part of the code as they go along to make sure that it is all working as intended. This can however, be very time consuming as it takes time away from coding to test the features. In most cases though the time taken out of coding to run the tests is well worth it for the ease of mind knowing each feature is working at the end of the development process.

In contrast to white box testing black box testing is often ran by people who are brought in from outside of the company to specifically test the software without knowing the inner workings. Black box testing is carried out to make sure that users who aren't developers can use the software without needing to know all the inner workings. This method of testing is used to make sure the software and product are working as intended. As mentioned earlier the white box testing is very time consuming as it takes a lot of time to test all the features

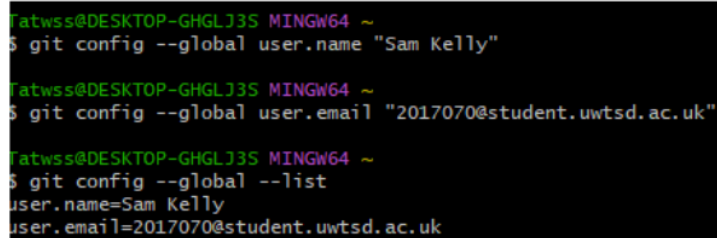
individually. This makes black box testing a relatively faster process as the users do not need to know the inner workings and are simply told to test how the program is working. They can sit down for an hour or so and test all the features of the website and report any problems that they come across in that time.

Both versions of testing should be used in conjunction with one another to make sure that the best version of the software is being pushed to live. If the product isn't tested by people who don't know what is happening then the developers will have no idea if it all works together or if someone who isn't tech literate will be able to use it. Software development is an intricate and fickle process. By utilising all the methods of testing developers can be sure that the best version of their product is being sent to the users and that they will have to make minimal changes after release.

## Version Control

In order to make sure that all changes made on a project are recorded and kept as evidence, many workplaces use version control software that allows changes to files and new files to be committed to a server and the edits of the files are public. One of the most popular version control software's available is GIT. GIT is an easy to use piece of version control software that allows the user to set which program they want as their editor as well as commit changes and new files easily and quickly with a number of simple commands.

The first thing to do when installing GIT is to make sure the name and email are setup properly for the GIT version being used. Setting the name and email allows. In a local setup the name and email may not matter as much but in the workplace one of the easiest ways of identifying who has made a change is from the name attached to the commit. Figure 9 shows how to set the name and email properly in GIT.



```
Fatwss@DESKTOP-GHGLJ3S MINGW64 ~  
$ git config --global user.name "Sam Kelly"  
  
Fatwss@DESKTOP-GHGLJ3S MINGW64 ~  
$ git config --global user.email "2017070@student.uwtsd.ac.uk"  
  
Fatwss@DESKTOP-GHGLJ3S MINGW64 ~  
$ git config --global --list  
user.name=Sam Kelly  
user.email=2017070@student.uwtsd.ac.uk
```

*Figure 9 Setting name and email in a new GIT repository*

Version control software is a key aspect when working in the field of software development, it allows changes to the source code to be tracked and it makes the developer think twice about what they are committing. By using version control software it allows the version of the code that is live to be untampered and still allow development to be happening behind closed doors. If every developer had to work on the live version of an application then features would break and be down for maintenance all the time. Although using version control software does not completely eliminate the accidental commit of a file that isn't ready. It does keep a record of who has made what changes to the project and allows rollbacks to take place if something goes wrong when pushing to the release version of the code.

Version control software should be used in nearly all workplaces with a handful of employees, although for an individual the software may be too much effort compared to the benefit they would get out of it. It is always good practice to make sure you are up to date with what is being used in the workplace in modern times. Staying up to date with software and techniques makes someone a far more employable person as there is less to teach them during the onboarding phase. Using version control software also requires an intense amount of storage space if it is a large scale project the FAANG (Facebook, Apple, Amazon, Netflix, Google) companies need petabytes of storage to keep their projects up to date and committed to their version control software.

Once the name and email are setup anyone can get started using GIT, the appendix below shows how to follow a step by step process of adding new files to the repository committing them and then creating a new file and showcasing how to edit a file and committing the edit.

## Conclusion

A number of software design practices that have become industry standards over the years have improved the quality of the development cycle an immense amount. Wireframes, CRC Cards, and User Stories have created a user first development environment to improve the overall user experience. Using agile principles and practices in the workplace has allowed the customer and the developers to stay in touch and transparent with each other. Working in sprints in the workplace has allowed each stage of a project to be tailored to how the customer wants. Although CRC cards can be very time consuming to get started in the first place if the project relies on object oriented program then they can save a large amount of time when create and implementing new classes.

Black box and white box testing are vital to the success of any program. Black box testing becomes far more useful when a project is nearing completion and requires the overall function of a program to be tested and should not be skipped. Whitebox testing can be done alongside the development of the project and both remain useful throughout development of any given project.

Version control software allows everyone involved in a project to be up to speed with the latest changes and developments. Using GIT makes the process incredibly easy and is a very valuable asset to every team involved in development, the usefulness of programs such as GIT should not be undermined and should be of the utmost priority when making changes to code in order to have proper fail safes in place should the code not work after a number of changes.

## Appendix

```
Tatwss@DESKTOP-GHGLJ3S MINGW64 ~/firstfile (master)
$ echo "Hello world this is a new file" > introfile.txt

Tatwss@DESKTOP-GHGLJ3S MINGW64 ~/firstfile (master)
$ echo "Goodbye new file it was nice to see you" > farewell.tct

Tatwss@DESKTOP-GHGLJ3S MINGW64 ~/firstfile (master)
$ echo "Goodbye new file it was nice to see you" > farewell.txt

Tatwss@DESKTOP-GHGLJ3S MINGW64 ~/firstfile (master)
$ cat introfile.txt
Hello world this is a new file

Tatwss@DESKTOP-GHGLJ3S MINGW64 ~/firstfile (master)
$ cat farewell.txt
Goodbye new file it was nice to see you
```

Figure 10 Adding new txt files to the GIT repository

```
Tatwss@DESKTOP-GHGLJ3S MINGW64 ~/firstfile (master)
$ git add introfile.txt
warning: in the working copy of 'introfile.txt', LF will be replaced by CRLF the next time Git touches it

Tatwss@DESKTOP-GHGLJ3S MINGW64 ~/firstfile (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   introfile.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    farewell.tct
    farewell.txt
```

Figure 11 Staging a new file and checking the status of the files being tracked by GIT

```
Tatwss@DESKTOP-GHGLJ3S MINGW64 ~/firstfile (master)
$ git commit -m "First commit of new files"
[master (root-commit) e69e85c] First commit of new files
4 files changed, 4 insertions(+)
create mode 100644 extra.txt
create mode 100644 farewell.tct
create mode 100644 farewell.txt
create mode 100644 introfile.txt

Tatwss@DESKTOP-GHGLJ3S MINGW64 ~/firstfile (master)
$ git status
On branch master
nothing to commit, working tree clean
```

Figure 12 Committing the files and checking to see if the working tree is empty and the commit has been successful

```

Tatwss@DESKTOP-GHGLJ3S MINGW64 ~/firstfile (master)
$ echo "This is an additional file to test features" > extra.txt

Tatwss@DESKTOP-GHGLJ3S MINGW64 ~/firstfile (master)
$ cat extra.txt
This is an additional file to test features

Tatwss@DESKTOP-GHGLJ3S MINGW64 ~/firstfile (master)
$ git add .
warning: in the working copy of 'extra.txt', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'farewell.txt', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'farewell.txt', LF will be replaced by CRLF the next time Git touches it

```

Figure 13 Adding a new file and sending it to the staging area

```

Tatwss@DESKTOP-GHGLJ3S MINGW64 ~/firstfile (master)
$ git commit -m "First commit for edit"
[master 76d4203] First commit for edit
1 file changed, 1 insertion(+)
create mode 100644 edit.txt

Tatwss@DESKTOP-GHGLJ3S MINGW64 ~/firstfile (master)
$ vi edit.txt

Tatwss@DESKTOP-GHGLJ3S MINGW64 ~/firstfile (master)
$ cat edit.txt
New file for editing, this file has now been edited

Here are some more edits

```

Figure 14 Adding edits to the new file

```

Tatwss@DESKTOP-GHGLJ3S MINGW64 ~/firstfile (master)
$ git add edit.txt
warning: in the working copy of 'edit.txt', LF will be replaced by CRLF the next time Git touches it

Tatwss@DESKTOP-GHGLJ3S MINGW64 ~/firstfile (master)
$ git commit -m "Edit is being committed to GIT"
[master 2bb2429] Edit is being committed to GIT
1 file changed, 66 insertions(+), 1 deletion(-)

```

Figure 15 Committing the new changes to the edited file

# Software Engineering Principles and Practices FINAL - Sam Kelly

## GRADEMARK REPORT

### FINAL GRADE

67 / 100

### GENERAL COMMENTS

#### Instructor

Eval of Design (20): 10  
User Stories (20): 14  
CRC (20): 15  
SW Testing (20): 13  
VC (20): 15  
Feedback:  
=====

The evaluation of the design process was a little brief and could have focused more on the benefits and drawbacks of the process.

I wanted a little more discussion of the benefits of SW testing, and you have not considered "testable" code.

I was expecting more debate of the importance of version control. You have demonstrated GIT working.

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

---

PAGE 9

---

PAGE 10

---

PAGE 11

---

PAGE 12

---

PAGE 13

---



**Comment 1**

???

PAGE 14

---

PAGE 15

---

PAGE 16

---

PAGE 17

---

PAGE 18

---

PAGE 19

---