

BSc Website Development - Major Project Dissertation

by Ashley Thomas

FILE	17885_ASHLEY_THOMAS_BSC_WEBSITE_DEVELOPMENT_-_MAJOR_PROJECT DISSERTATION_78196_69555026.DOCX (1.08M)		
TIME SUBMITTED	21-APR-2016 12:21PM	WORD COUNT	12841
SUBMISSION ID	56192731	CHARACTER COUNT	67557



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

Investigating the use of Software Testing within Mobile Applications Created Using WPF

by
Ashley Thomas

Supervisor: Mike Dacey
Second Marker: Kemi
Ademoye

**Project submitted as part
of the requirements for the
award of BSc Website
Development**

Table of Contents

Abstract	2
Aims and Objectives	2
Summary of the Methods Used	2
Summary of the Major Findings	2
Summary of the Conclusion and Recommendation	2
Introduction	3
Literature Review	3
Software Testing	3
Bugs within an Application	4
Validation and Verification	4
Software Testing Within Mobile Apps	5
Software Testing Within A WPF Application	9
Development Methodology	11
Waterfall Method	11
Agile Method	12
Requirements Analysis	14
Research Methodology	15
Design and Implementation	17
Evaluation	22
Conclusions and Recommendations	23
Reflection	24
References	26
Appendices	30

Abstract

Aims and Objectives

The aims of the project is to research about the different software testing techniques that can be used for a mobile application in order to build a working application within WPF, that has some of the best software testing techniques used to make sure that the application runs without any errors or bugs.

The objectives of this this project is to research through the different software testing techniques that can be applied to mobile applications, and also to narrow down the best techniques in order to use them within the application.

Summary of the Methods Used

The research method that will be used for this project will be the research onion. This method of research is chosen as it can be utilised to the full potential, and all of the layers can be used in order to narrow down and find out the best techniques of software testing for mobile applications.

The development methods that will be used within this project will be both the waterfall and agile methodologies. The waterfall methodology will be used to plan out the stages of the entire project. The stages will be used to go through what the requirements are, what the design will look like, how the application will be implemented, and so on. The agile methodology will be used in order to plan out sub-stages within the waterfall method, to allow time for plenty more software tests to be done within the application.

Summary of the Major Findings

In the end of this project there will be a section for the evaluation, which will give information about what the major findings are going to be. Whether or not the software testing techniques used are the best, and also whether everything researched was used within the application.

There will also be a section which will be about any software testing techniques that could have been used, as well as techniques that were not useful, and that was taken out of the application due to there being no need to use that particular form of testing.

Another section of the evaluation will contain any changes that could take place in the future, and also what would be done differently if the project was to be done again.

Summary of the Conclusion and Recommendation

There will be an overall summary of the conclusions and recommendations of the project. The conclusion will contain information summing up the overall project, what the different types of methods were used, as well as whether the application has any errors or bugs, once all of the unit and user interface tests have been put in place.

The recommendations are going to contain any information about the application itself, for example whether or not the software testing techniques that were chosen, are the best for mobile applications. It is also going to be about whether

there could be any more techniques that were mentioned within the literature review that could have been used within the application itself.

Introduction

In the recent years, mobile applications have become a huge place in the market due to almost everyone now owning devices such as smartphones and tablets. This project will be outlining the many different techniques and strategies used for software testing to make sure that the application works best for mobile phones. In this report it will going through the different ways to test your application as well as outlining what a WPF application is, how you can test the mobile application within visual studios WPF, and the best way to test it. All of this is needed to make sure that your mobile application will run at its best with no errors.

This report will also be outlining the different research and development methodologies in order to research and develop the different software tests in order to test mobile applications that are created using WPF.

The overall aim of this project is to find out what the best software testing techniques are to use to make sure that any mobile application that is created using WPF, is working correctly without any errors or bugs.

The objectives of this project is to research through different software testing techniques to find out which ones are best to use in order to create a working mobile application that has been created using WPF and that has no errors or bugs. Another objective is to use the development methodologies in order to create a working mobile application, with no errors or bugs.

The design and implementation part of the project is where the application is going to be designed, then built with the software testing techniques chosen and put in place within the application.

Literature Review

Software Testing

When creating something such as a mobile application one of the most important aspects is testing that piece of software. You test software to make sure that anything you create will work at its best and will not have any errors [1]. If you were to create a program for a company then you will need to do a lot of software testing because if you sent out the program without testing to make sure it works the way the customer would like, then not only could there be numerous errors that show up, but also the program might not even do what the customer wants. So for example if a company wanted a program that stores customers information in a database, then you will need to test to see if the database connection even works on the program.

Most people do tend to either not do software testing when it comes to mobile applications, or if they do, they will not do as many forms of testing. This is because software testing can be very time consuming and can cost a company a lot more money, when it comes to mobile applications.

Bugs within an Application

In order to understand what software testing is all about and the reason why we test any piece of software before releasing it, we are first going to need to know what bugs are within the application and what forms of software testing can be used to make sure that the bugs are gone from the application.

A bug within an application can come in many different forms [2]. Most of the forms the bugs come in do tend to occur when the application does not do something that the specification says it should or if the application does do something that the specification says it should not do. Another way you would know if your application has a bug is if it runs slow or if the tester can see that the user would view the application as it not being right, so for example if the application had buttons that were too small for the user to even press, or if the display was difficult to read in some lighting, either way these are all forms of bugs.

The main cause of software application bugs are the specification stage [3]. In some cases this is due to the specification not being thorough enough and things need to be added to prevent these bugs. Other cases might be that the specification is constantly changing and things are getting left out. Planning out the different stages is vitally important because it stops these bugs from ruining great applications.

Another big cause of bugs within a software application is the design stage. You could compare the design stage to an architect creating blueprints for the layout to a new building. The same could be said for software applications, if the design of the application is not done correctly by it being rushed or constantly being changed while in the building stage of the application then these will be huge causes for bugs within the system.

Validation and Verification

Validation and verification are both very important when it comes to software testing because they both help in making sure that the application meets the user requirements. Validation is a way to reduce the amount of errors that are being entered into the application [4]. The way validation works is that there will be a number of tests which will bring an error message up to the user when they are using the application. So for example if the application had a textbox where the user would need to enter in their email address, the email would need the "@" symbol in order for it to be an actual email address. The validation would be bringing up an error message next to the email textbox saying that the user needs to put in certain symbols for it to be a recognisable email. This creates a lot less errors within the application and also will let the developer see if they are meeting the user requirements. It also lets the user see if they have forgotten something within that textbox, which can be very useful for them.

Verification is when the developer makes sure that the application works correctly and everything has been implemented within the user requirements. Verification is done at the start of developing the application and runs on throughout the entire project. It includes numerous meetings with both the user that the application is being developed for and team meetings if there is more than one person working on the application. All of these meetings would be to ensure that everything such as the code, specifications and user requirements are being evaluated and thought through, every step of the way [5]. This is a very good method of testing because it ensures that every part of the application is being thought through very carefully and meets all of the user requirements. An example of verification would be if you were creating a table you will need to make sure all of the parts to the table are checked to see if all of the legs are the same size. If one of the legs are too small, then the table is incomplete and it will not work properly, and it will become very unstable. This is a good analogy for when you are about to build an application because much like the table that is unstable when one part is not corrected, the same can be said about the application because it will not work correctly when one part is not working, it then becomes very unstable and might not even work.

Software Testing Within Mobile Apps

One very important aspect of software testing when it comes to developing a mobile application is to make sure that the screen resolution size can fit all mobile devices. If this has not been corrected then the mobile application will look very strange and parts of the application will not be in the correct places, so for example a button might be too small to see or it might even appear at the bottom of the screen, when it should only appear in the top of the screen. Every mobile device has different screen sizes, some are much bigger than others, so when you test your software you will need to test the screen resolution so it fits to all different mobile devices.

There are three very effective techniques when it comes to testing software [6], these techniques are: structural testing, functional testing and code reading.

Structural testing

This is when you test the internal structure of the application, opposed to the functionality of the application, so for example you will test what certain loops do within the code and how they work.

Functional testing

This is when you test the functionality of the application, so you will need to construct test data in order to check that certain functions of the application are correct [7]. For example you could have a function that adds an item to a list and then you will need to construct some test data to make sure that the item does add to the list.

Code Reading

This is when you will identify the certain classes in the application, determine what that class is trying to do and then create certain functions in order to make the application work at its best. For example if you were creating a class and called it "Calculator", then you will need many functions such as, "Add", "Subtract", "Divide" and so on.

Performance Testing

Performance testing is used to determine how fast the application is running, as well as the scalability and how responsive the application is [8]. This method is when the developer will write a test before they finish the starting point of their code. It allows the developer to think through their requirements and design stages before they start writing the code. So for example if a developer did not use this form of testing, then they could finish with an application but it might not even meet the non-functional requirements of the user. TDD is a great way of planning out ahead of time.

GUI Testing

There are a few testing tools that are able to automatically go through the graphical user interface within an application and are then able to find any faults or errors within the code. There are two different types of tools [9] that the developer can use. These tools are called "AndroidRipper" and "Testdroid". Along with finding any errors within the code, AndroidRipper will also reverse engineer the code and then create abstract models of the graphical user interface.

Stress Testing

This form of software testing allows the developer to look through what the user requirements are and see if they can go above the expectations. This form of testing is performance testing but at a higher level [10] and it also uses a high level of simulated load. Both stress testing and performance testing are usually done in conjunction with one another.

Load Testing

This form of testing is where the developer will check to see how fast their application will load. They will usually test their piece of software at peak times, to see if their application will maintain a current speed or if it will break. Load testing [11] is used to evaluate the responsiveness of a system with the performance requirements.

Security Testing

This is a very important form of software testing due to many complications that can happen when dealing with important information from different users. If your code is not up to standard when it comes to security then many hackers will try to break your application and this could be very dangerous if you were to build an application that stores important information about the users. If you stored important information such as bank cards within your application, then the security of your software application must be constantly up to date. In order for you to do security testing you will need to reveal any flaws within your application, so for example you will need to try and break the code in order to fix the security. Security testing [12] will determine whether or not your application maintains the functionality and protects the data of the user as it should.

Unit Testing – closely related to TDD

This is when the programmer will grab the smallest testable part of their code and test to see if it works as it should, then they will test each individual unit until they have tested their entire code [13]. You will need to test every function within your code. Most software will automatically do the unit testing for you, but you are able to do it manually.

Test Driven Development

Test driven development is very closely related to unit testing and it is when the programmer will combine something called test-first development with something else called refactoring [14]. Test-first development revolves around the idea of actually writing the test before you finish writing just about enough code to run that test. So for example you could have something within the code that allows you to read and write data to a database. Just before you finish writing the code for the read data, you would write the test to see if it actually reads the data from the database. As soon as this is finished and the test works, then you would move on to do the exact same thing but instead this time you create the test around writing data to a database.

Refactoring is when you will restructure the already written code, without actually changing the way the applications front-end works [15]. So for example you could have an application that runs perfectly but it could have hundreds of pages of code to run through. This is where refactoring takes place because you will change the code behind the application but the application has not changed to the user point of view. In order to tidy up the code you could use something called polymorphism and something else called inheritance which can turn hundreds of pages of code into only a few pages because it stops repetition of the code by implementing different classes and putting all of the base code into one class which other classes will derive that code from. An example of these would be, with polymorphism you could have a person class which has a reading method. Then you could have a student class that extends the person class and you could then use any method within that person class. So you could then use the reading method. An example of the inheritance would be if you had a person class and you have a student class which extends the person class then student inherits everything that is not set to private from the person class. If an attribute or a method is set private then the student class would not be able to use that attribute or method. Another way of looking at this would be a child inherits certain things from their parent, so again if you had two classes, one called child and one called parent and there was a method called hair. Then the child class could inherit the hair method from the parent class.

Regression Testing

You would use this form of testing after you have edited your code because this allows you to uncover any software bugs you might have [16]. If you change your old code in any way there could always be a bug within your application, but with using this method of software testing it allows you to seek out any of the bugs that might be hidden within your code.

Test Plan

A test plan is a documentation process that allows the programmer to test their different objectives and it is also used to see if they are going in the right direction when it comes to the user requirements. Test plans [17] will go through the different resources needed in order to complete one part of the plan. It also plans out the scheduled dates for when one part of the test plan starts and when it ends, much like how a project plan works. This is a very good way to test whether the application created will be able to work correctly and it makes sure that the user requirements are being followed every step of the way.

White Box Testing

This form of software testing [18] will examine the structure of the program then it will come up with some test data from the code that the programmer has used. It will verify any security holes that are missing within the code. This can be very important when building a very large and important program where personal information is stored because the security of the program will need to be the best and unbreakable. There are many other names for this type of testing such as glass box testing, clear box testing. It uses these names because just like a glass that is see through but at the same time you can see the structure of the glass, the testing will look through all of the functions and get straight to the structure of the code within the program, then it will get some test data in order to test the programs use. The structural testing and unit testing that was previously mentioned are also both a form of white box testing. This is a really good strategy to use to test whether your program meets the requirements of the user [19] because firstly you will need to look at the internal structure of the code, then you will need to run some test cases and this will then check to see if all of the requirements that the user has given will work correctly. Based on the test cases the user has, they will need to have expected outputs with the actual outputs that the program has in order to check what is working and what does not work.

Advantages:

The advantages to using white box testing is that it will make the developer to think carefully about the implementation of the code [20]. It will also reveal some hidden errors within your code that otherwise would go unnoticed. It also reveals the issues related to dead code, or code that does not cater to the programming practices that every programmer should use. Any unused lines of code that can introduce new bugs into your program can be taken out with this form of testing because the tester will ask about the implementation of each section and this will find if you have any unused lines of code.

Disadvantages:

The disadvantages with white box testing are that it can be expensive because it will take a lot of time and money to do white box testing. It also has a slight possibility that it can accidentally miss out a few lines of code within the program [21]. You will also need to have extensive knowledge when it comes to the programming language that you are using in order to perform white box testing. If you were to analyse each line within the code then is basically impossible and very time consuming which means that there might be defects within the code.

Black Box Testing

Black box testing, also known as behavioural testing is basically the opposite of white box testing because instead of testing the structure of the code, you will be testing the functions within the code. This method of testing is named black box testing due to the software application to the tester is like a black box[22], they are not able to see what the inner workings within the application are, they will only know how to use the application. Usually with this form of testing [23] you will have two separate people, one for designing the program and the other one for testing the program. The tester only needs to know the inputs and what the expected outputs will be and not how the program works. The tester will not need to look at the code because they will act as the user instead of the

designer. The functional testing that was previously mentioned is a part of the black box testing.

Advantages:

The tester of the program [24] does not need to know any technical programming languages because they are only there to act as someone that is using the system. As soon as the functional specifications have been completed by the tester, the test cases can then be designed to keep the requirements of the user on track. The person carrying out the black box tests will be able to test large chunks of code at one time, this will save them a lot of time for them to carry on with creating and testing their application. It is also a major advantage that there are two people [25], one person developing the program and the other person acting as the user because you will be getting two different peoples perspectives on the application.

Disadvantages:

It can be quite difficult to identify all possible inputs from the user within the short amount of time frame you have. So writing the test cases is going to be a lot more difficult and slower to create [26]. There can only be a small amount of test cases performed, so the program might still have some bugs hidden somewhere within the code. There is a lack of good testing all due to the person who is testing not having good knowledge of the application. It is a lot more difficult to test everything that the developer would like due to the tester not knowing the code behind the application, so they will not know anything about any of the errors or bugs behind the code.

Software Testing Within A WPF Application

When creating a mobile application you are always going to need to test it to make sure that it works correctly and it also meets the requirements of the user. One of the main uses to test a mobile application is to check to see if the application works on any device. This is just in case different users have different screen sizes, so the application will need to stretch to their size of the screen without it looking distorted. As you can see testing the software is one of the most important things a developer can do in order for their application to work and run at its best and to make sure that it meets the requirements and design of the user.

A windows presentation foundation application, or WPF as it is usually called, is basically used to create a windows-based application, of which you will be able to create the interface for the application [27]. There is one software testing technique called user interface testing that would great in order to fully test the interface of this application, making sure that no errors or bugs can reside within the interface itself.

Before the testing can begin the mobile application will need to have started so there can be some code to test. The best method to test any application is to make sure that it works correctly and that there are no bugs or errors, and that is to make sure that every time a certain part of the code has been done, the developer will need to test that part of the code. This is basically unit testing which resides within white box testing. This is the best method to use because it allows the developer to check over every little detail of their code to see if anything needs to be added.

When developing the application within the project, the plan is that each function is going to be tested to check to see if it is working correctly, so unit testing will be used here because it is used to test each function of the code. There are namespaces within the Microsoft visual studio software that supply classes that allow unit testing. The namespace itself has attributes which give the testing information such as what is being tested, to the test engine. This method makes sure that every part of the code is tightly sealed and no bugs or errors can get within the code and it also allows the developer to look through every part of their code to check to see if anything is missing or if there is a need to add more functions.

There are a few different ways that unit testing works. Depending on the workflow, you can either test the functions or the classes. The functions will have a different form of testing than the classes. For example if you were to test a function then you are going to need to create test files that will test the specified function. These files will specify what function is going to be tested and it will then return an array of objects. In order to test classes you are going to need to create test files just like the functions but you are going to need to specify what class is going to be tested instead, it will then return a single test case object.

In order to create unit tests within visual studios WPF application, there are going to be a few things needed. The first thing will be to create a new project and search for something called a class library. This is where all of the classes of the application will be stored. It gives the developer easy access to view and change the classes when they are stored within a class library. The next step is to create another project but this time search for something called unit test project. This is where the developer will create white box tests that will test the functions within the application. For example if you had an adder class that will add two numbers together, then within the white box tests class under the unit test project, the developer would create the adder object and then use a tool within visual studio called "assert". This tool is used as a way to say that something within the code is going to be tested. There are four different stages to complete a unit test. The first stage is called the setup which is where you would create the instance of the class object. This was previously mentioned above with the adder class. The next part is called the exercise stage which is where the developer will put the code that is going to be tested, so for example this part would be what the number one and number two variables are equal to. The next part is called the verify stage which checks the result of the test using the assert tool to flag any errors within the code. The last part is called the teardown stage which is where you close down any resources that were open when running the code, so for example it will close something like a database or a file that was open.

Another form of testing that would be great to use within the WPF application would be the UI unit tests. This is where the developer will create black box tests that will test the interface of the application. These tests can be carried out by the user instead of the developer because these tests record what the user will enter in on the application and then fill in the code associated with the actions of the user. So for example the first thing the user would do is hit the record button, then they would run the application and if the application had two text boxes with a button that says add, the user would then type in two numbers in the two different text boxes and then click the add button. After that they would then hit the record button again to stop the recording. Once that has been done they will then have an option to generate the code for the actions the user has done.

There is also going to be the need to create some test cases, this is just to check to see what the user requirements are, what the expected outcome of those cases are and what the actual outcomes of those cases are.

Development Methodology

In order to develop this application there is going to be the need to pick and choose from multiple methodologies. There are many different ones that can be chosen, below will show a few of the methods and give an explanation of what they are. There will also be a requirements analysis which will show how they can be applied and why those certain methodology techniques have been chosen for this application.

Waterfall Method

The waterfall model is based around the idea of working one stage at a time [28]. Once one stage has been completed, you will then move on to the next stage. There is a big problem with this because if you change something, you cannot go back a stage, you will have to restart the whole model [29].

The waterfall model has many different advantages and disadvantages that are shown below.

Advantages

The waterfall model has an extensive record keeping system that allows further improvement on the project within the future [30]. It also shows the client what they are going to expect out of the project. They will have an idea of the timescale, cost, how big the size of the project is and also what the program is going to do when it is completed. They will know all of this before the project begins because the waterfall model does not allow any changes or errors within the project.

Disadvantages

Once one of the steps has been completed, you will not be able to go back to that step to change something. Instead you would have to restart the whole project over [31]. So if you had an error somewhere within the model, you would have to restart everything over again. This is a very big disadvantage

because if you were working on a really big project that you were already a few weeks into and something went wrong you would have to start it all over again, so those few weeks would have been a waste of time and money.

Another disadvantage is that the testing stage of the application is only done at the end [32]. So if there were bugs or errors within the code, then you could be there for days or even weeks to try and find them all and fix them. If there was a bug within the program at an early stage, then that bug might have affected the way the other code was written.

The waterfall model also does not take into account what the client needs within the future of the project [33]. So for example if the client has asked that they would like a simple calculator program built for them, then the waterfall model is good to use, but if they all of a sudden had said that they would like it to be a scientific calculator with a lot more calculations they could use, then any progress that has been made will need to be restarted. You would need to make sure that the client has everything they want within the program before the project can begin.

This model can be used in creating the application within this project by going through all the different stages of the waterfall method. The stages are requirements, design, implementation, testing and finally maintenance. The requirements will be all about what the application is going to be, as well as what types of software testing will be used within creating the application. The design stage is going to be about what the application is going to look like as well as the main functions of what the application is going to do. The next stage is the implementation where the application will be implemented as well as the different software tests that will run alongside the application. The final stage which is the maintenance, will be about keeping the application and especially the software tests as up to date as possible.

The big downfall that will not work with this project is that the waterfall methods testing stage is at the end and because this project is all about software testing, there will need to be some changes. Another downfall is that nothing can be changed and no errors are allowed. That is where the agile methodology comes into this project.

Agile Method

The agile model was created to basically solve all of the problems that come with the waterfall model [34]. Basically the agile method breaks down the stages of the waterfall method into different smaller modules. Each module will have a certain goal to be completed, whether it is a few hours or days. At the end of each module that has been completed, the project is evaluated and the entire program is tested extensively for any errors or bugs that might be hidden within the program. The client feedback is then incorporated into the cycle before the next module can start.

The agile model has many different advantages and disadvantages that are shown below.

Advantages

The agile method allows you to change the project and the program after the initial planning has been done [35]. This would mean that if the client was to change something, then there would be no problem because this method works at an incremental approach, which means the different stages are broken down into much smaller modules that can be altered at any time.

Seeing as the agile method allows you to make changes at any time, it will be a lot easier to keep up to date with the newer developments and also allows you to keep adding more features.

At the end of every module that has been completed, you will go through the projects priorities and re-evaluate to see if any changes or features need to be added [36]. This will also allow the client to add their feedback on what they want the program to do, or any new features they would like to add.

There will also be extensive software testing within the agile method because at the end of every module that has been completed, the entire program is tested for any hidden errors or bugs, they can all be caught within the development process, unlike the waterfall method that does all the testing at the end. With the agile method, because the testing is done so thoroughly throughout the entire development process, the program itself can be released at the end of any cycle. This would mean that the program is going to reach the launch date without any errors or bugs, it will be a perfectly completed program.

Disadvantages

If there is no project manager, then the project can be just a complete series of code modules, which means that instead of thinking every step through and talking with the client, you just continue coding the program until it is complete [37]. The project could then entirely end up as either over budget for what the client wants, or the launch date could end up being postponed.

Seeing as the agile method does not have a definitive plan all the way through, you could end up with a completely different program to what was first thought about [38]. For example if the client wanted a simple program to be made, maybe for someone who is not good with technology, but then the client keeps adding to the program, then you could end up with something a lot more complicated than what the original plan was.

This model can be used in creating this application by using the different stage modules that it works with. The application will have stages just like the waterfall model, where every stage is thought out before anything has started. The only difference is that this application will also incorporate the agile model where each of the stages will have their own modules of which will allow more testing within each module. This is due to the project being based around the idea of testing software for mobile devices very thoroughly. Basically each module will have software tests involved to allow the application to run and work the best it can with little to no bugs and errors.

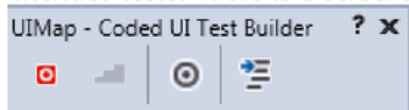
Requirements Analysis

The two main methodologies to use within this project would be some of the techniques used in the waterfall and the agile models. Both of the methods have their own advantages and disadvantages which have previously been mentioned, but there are a couple of techniques that have been chosen from both methods in order to create a really good working application that has very little to no errors and bugs.

The technique used from the waterfall method is to go through every stage it has. So that would mean that there would be four stages of which this application would be built around. The four stages would be the requirements, design, implementation and maintenance. There is one stage of the waterfall method that will not be needed within this project which is the testing stage. This has been taken out because the agile method is also being included and that method allows for testing throughout every stage.

The technique used from the agile method is to have many different modules within each of the stages to allow the application to have the software testing it needs in order to create an application that has no errors and bugs. This is going to work by testing every step of the application, once one part has been complete. For example if there was a bank application that stores the current amount of money you have, then once that part of the application has been built, you would then run a test to see if that money gets stored within that persons account. The next part could be adding money to that persons account, which when that part of the code has been completed, the next part would be to run a test to see if that money has been added to their account.

The software needed in order to complete the application is going to be visual studio enterprise edition. This software is needed because it has a certain tool which is used to do the user interface tests. This tool is called coded user interface tests. There is a screenshot below of what the tool looks like.



As you can see the first button is used to record the actions that the user has done on the interface. Once the user has finished with the test, they will then click the record button again to stop the recording. The next button is used to show the user their recorded steps. The button that looks like a target, is used to select a certain part of the interface, such as the button. Once the button has been clicked, there will be a UI map that appears. This map is used to see the information on that button, such as the name and what happens when that button is clicked.

When it comes to the hardware needed in order to complete this application, the only piece of hardware that is needed would be either a laptop or a desktop pc. There will not be any need to have really good specifications in the computer, just so long as it is good enough to run the software installed, which is visual studio.

There are going to be some tools that are needed to run the software testing techniques within the application. The reason to use the enterprise package of visual studio is because that is the only package that comes with both the coded user interface testing tool, and the unit testing tool. These tools are the coded user interface testing, which comes with the enterprise version of visual studio. Coded user interface testing will be used to create the interface tests. The other tool is unit tests, which also comes with the enterprise version of visual studio. This tool will then be used to test all of the functions within the application.

Research Methodology

There has previously been mentioned about the different types of methods used to collect the data for this project but not the underlying theories and concepts behind these methods.

The research onion is one of these concepts that are used in developing research strategies [39]. When viewed as a diagram, the research onion is set up as a bunch of layers which describe more details of the research process. There are five different layers which are, research philosophy, research approaches, research strategies, time horizons and data collection methods. Within these layers there are different methods of researching how that layer can be completed by the style of project you are doing, whether it is more focused on experimenting on the application or more focused towards the users wants and needs, where there will be a lot of interviews and surveys.

Research philosophy – Each of the choices that will be made within this layer will require very careful planning and thinking because this layer provides not only the structure of the research but also any limitations within following certain decisions that have been made [40]. It also helps in finding how the researcher can collect data that they will analyse in order to have their findings. Positivism is a philosophy that can be used which will generate a question that can be tested and then how they can be used within the real world. It is also used to create research where another researcher will be able to create the same result using their findings. Positivism is going to be used here within this project because there is going to be the need to answer many different questions about the application, such as how a software testing method such as unit testing can be used here to build an application that will work without any errors or bugs. Then the code that has been created to test that piece of code can be used by other developers so they can generate the same result, but only if they are working on an application that is very similar.

Research approaches – This layer contains two different terms, inductive and deductive. Inductive is when you are researching something to create your own theory. This means that you will need to research everything within that topic in order to create your own theory [41]. If there is not that much information within the field you are studying, then inductive would be the best approach to use. Deductive on the other hand is when you will start off your project using a statement or even a question that the aim will then be to answer it. The deductive approach works the opposite of inductive because you will use existing

theories in order to answer the statement or question you have. For example if you had the question "What are unit tests?", then you would need to research through the different journals and theories in order to find out what the answer to this question would be. This project will be based around the inductive approach because there is going to be the need to search through current research about software testing methods in order to create a mobile application that does not have any errors or bugs.

Research strategies – This refers to the style of research that you will use to collect and analyse the data. There can be more than one choice within this layer that can be used to collect and analyse the data [42]. The experiment research strategy would be used here for this project because it is all about experimenting on the research that has been found. Within this project the research was about different software testing strategies and one in particular called unit testing. The experiments would then be all about how unit testing among some of the other testing methods can be implemented within mobile applications and what the best testing method approach is to use for mobile applications.

Time horizons – There are two different types of time horizons that can be used, one is called cross-sectional and the other is called longitudinal. The cross-sectional design can use both quantitative and qualitative data and it uses different aspects of either groups or individuals. Longitudinal design also uses both quantitative and qualitative data but it is used to study events and different behaviours over a much larger timescale [43]. The longitudinal design will be used within this project due to the need of studying the different events and behaviours that the application is going to do in order to derive the software tests.

Data collection methods – This layer is used for the researcher to find out what data collection method is best to use [44]. This includes decisions on whether they will need a small group, whether they are going to use questionnaires and if they are going to do interviews then what questions are going to be on the interviews. All the decisions and tools the researcher is going to use will need to fit in with all of the other layers of the research onion.

The data collection methods that are going to be used within this report are going to be positivism which will be used to look at how different software testing methods such as unit testing can be used to build a mobile application that will work without any errors or bugs. The next method will be inductive which will require current research about software testing within mobile applications. Following this method will be experimenting with the different types of software tests that can be used within a mobile application, such as unit testing and user interface testing. The final method is the longitudinal design which will be used to study the different behaviours that the application will do, where the software tests will then derive from.

The reason this form of method was chosen to research software testing within mobile applications was because the research onion has many layers of research that allows this project to have the best current, up-to-date versions of testing mobile applications.

The software testing process of this project will be correct and allow for the application to have no errors and bugs because this project is full of experiments that will have many different reasons for each of the tests. This will make sure that all of the tests have a reason for doing them and it will not have a test that there is no need of. For example if there was a bank application where the user can add money to their account, and if there was a test to check to see if the user has divided two numbers together, then as you can clearly see this is a test that does not need to be included because if you are adding to an account, then there is no need to have a test that will use a division sum.

It is always good to do some testing experiments even if it is just to test something that might not even need it, simply because not only can it be easily deleted without any consequences, but it might also be useful if there are any changes to the application within the future that might need that test to be run to check if there are any errors and bugs within the new code.

Design and Implementation

Design

While designing the application there is going to be the need to implement software tests throughout the entire application. The software testing techniques being used are going to be both unit testing and user interface testing.

In order to test a mobile application there are a few techniques that will be used in order to make sure that the application will run without any errors or bugs that would ruin the application. The first form of testing is called unit testing, this is a form of white box testing and it is used to test the functions that are within the different classes. This application is going to be a bank account application, so one of the unit testing methods are going to be when the user updates their bank account if they have purchased something, then it will show the expected result after the purchase and it will also show what the actual result with an amount added to their account, which will then have a string that says the account was not debited correctly. This is a test that will make sure that it will update the account balance correctly and there can be no money in their account such as "0.001".

Another unit test would be to have a test method that when the user has a debit amount is less than zero and then it should throw an expected exception argument that debit amount out of range. This unit test would not have any asserts within the method because the assert is handled within the expected exception.

The other software testing technique that will be used within this application is a form of black box testing which is called user interface testing. These are used to test the interface that the user is going to work with. It has a few buttons that can be used to do the testing easily. The first thing to do is to press the record button, which will record all of the actions that are performed within the application. The next thing is to run the application and go through each of the tests that need to be done, one at a time. The first test could be to add money

to the current account. Once the money has been added and the actions have been performed, click on the record button once again to stop the recording. After that there will be another button called generate code, this is used to generate the code for the actions that have just been recorded. The next thing that is going to come up will be the name for the test that has just been done, which it would be called something that revolves around adding money to the account. Once that has been completed the code will be generated and that will then be completed.

Another user interface test that can be done is a test where you add money multiple times to the account and then the test would be revolved around seeing if the ending result to the account did add up correctly. For example if you were to add numbers multiple times to the account, then the numbers should all be added instead of replaced. If the numbers are replacing themselves instead of adding together then the test would fail and something would be wrong within the code and that would then need to be changed.

There are some important factors that are used within this application. They are as previously mentioned both unit testing and user interface testing. The user interface testing is a very important factor to the implementation of mobile applications. This is because all mobile applications have interfaces that the user will need to interact with in order for the application to work correctly. If there was no user interface testing within a mobile application, then this could cause many problems to the code that works behind the interface due to not properly testing the interface. You could use unit testing instead of user interface testing, but the unit tests are only used to test the functions within the code. It will not be able to test the actual interface which could cause many errors and bugs later on within the other stages of development. Using both unit testing to test the functions in the code and also using the user interface testing, which tests the interface that the user is going to work with, will be the best testing techniques to use to make the mobile application run without any errors or bugs.

There are also a few factors that were mentioned earlier that are not going to be used within creating the application. A few of these factors that were mentioned was load testing and refactoring. These factors were not chosen to be used within the application due to them not being needed. For example the load testing was not needed due to there being not much reasoning behind testing to see how fast the application will run. With there not being any errors or bugs once the application is completed, then it will run as fast as it ever could, so there is no reason to run load tests. Refactoring was also not needed due to there being no reason to restructure the code once it has already been written. Unit testing will test the code, so there will be no need to restructure anything as it will already be structured and tested correctly.

The design processes that are used to create this application are going to be based around the idea of a mobile banking application. The application will be used for the user to update the money within the bank account. The user will also be able to check to see if the debit amount they have paid for will come off of their current balance within their account. If there is no money within their account and they have used their card for a debit transaction, then the program will throw an argument where their current balance is out of range and is going into the minus range, which is not good for the application. The objective within

this project is to find and use the best software testing methods within a mobile application. The best methods that were chosen are both unit testing and user interface testing.

The processes contribute in the successful completing of this application due to both the unit testing and the user interface testing being used within a mobile application to make sure that there are no errors or bugs. The processes that are used will be successful because all of the tests will show if there are any errors within the code or within the actual tests themselves. If there are any errors or bugs within the code or tests, then they will be fixed until the application works perfectly without any errors or bugs.

This was all done due to there not being much software testing around mobile applications. There is a lot of research revolving around software testing as a whole, but not as much research around software testing within mobile applications. This is why the project is all about the best solutions of software testing around mobile applications. This is also why unit testing and user interface testing were both used within the mobile application.

The white box testing, which is the unit tests, and the black box testing, which is the user interface tests, are both going to work best by creating two separate projects within the project solution. One of them will be called white box testing and the other one will be called black box testing. All of the classes that will be used within this application are all going to need to go within a main class library. This library will hold all of the classes which will make it easier to access each class that will be needed in order to complete all of the tests.

Implementation

There are going to be requirements that are needed for the mobile application that is going to be created. Seeing as this project is based around software testing mobile applications, the requirements are going to be about testing the mobile application around the idea of how the user is going to interact with the interface. These requirements were derived from the research based around testing mobile applications.

There is a use case diagram in the appendices section that shows how the application will be used with the user and how the software tests that were created are interlinked between both the tests and the application itself. It shows how the tests work in the background of the application to make sure that there are no bugs or errors within the application.

The interface that has been created for the application is the bank account of a user. The interface shows how much the user currently has in their account, in order to add any more money into their account the admin will need to use the textbox provided in order to add the money into their account. Below is a prototype diagram of what the application is going to look like.

There were a few problems that came up when completing the application. One of the problems came when adding a unit test which updates the balance when the user account has been debited. Below is a screenshot showing this problem that had occurred.

```

using System;
using Microsoft.VisualStudio.TestTools.UnitTesting;
using MainLibrary;

namespace WhiteBoxTests
{
    [TestClass]
    public class BankAccountTests
    {
        [TestMethod]
        public void WithValidAmountInAccount_UpdatesTheBalance()
        {
            double balance = 15.99;
            double debitAmount = 5.55;
            double expectedAmount = 8.44;
            BankAccount account = new BankAccount("Mr. Earl Richards", balance);

            account.Debit(debitAmount);

            // assert
            double actualAmount = account.Balance;
            Assert.AreEqual(expectedAmount, actualAmount, 0.001, "Account was not debited correctly");
        }
    }
}

```

This problem was then later solved by correcting the expected variable. The expected variable was not correct due to the numbers not adding up to the balance. The debit amount variable and the expected variable should add up to the balance and as the screenshot shows, this was not the case. The screenshot below shows that this error was corrected.

```

using System;
using Microsoft.VisualStudio.TestTools.UnitTesting;
using MainLibrary;

namespace WhiteBoxTests
{
    [TestClass]
    public class BankAccountTests
    {
        [TestMethod]
        public void WithValidAmountInAccount_UpdatesTheBalance()
        {
            double balance = 15.99;
            double debitAmount = 5.55;
            double expectedAmount = 10.44;
            BankAccount account = new BankAccount("Mr. Earl Richards", balance);

            account.Debit(debitAmount);

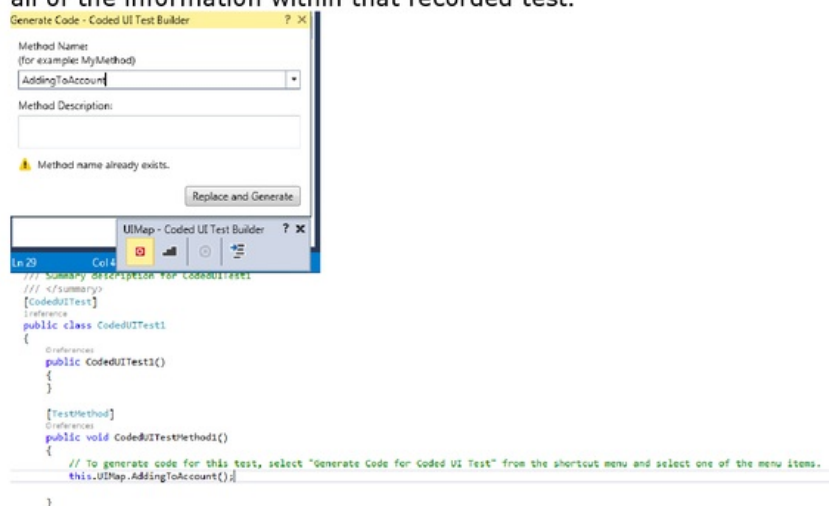
            // assert
            double actualAmount = account.Balance;
            Assert.AreEqual(expectedAmount, actualAmount, 0.001, "Account was not debited correctly");
        }
    }
}

```

The unit testing that was done has some methods. One of the unit testing methods used was to test to see if the account updates the balance when the user has a valid amount of money in their account. The next unit test method was used to test to check if the user has less than zero within their account, then it should throw an out of range exception. The next unit test method used was to test to check that if the debit amount is more than the current balance, then it should throw an out of range exception. The final unit test method used was to test to see if the user account is in credit, which if it is then the test will throw an out of range exception. The out of range exceptions are used instead of the normal assert method because within this test, the assert is handled by an expected exception, which is the argument out of range exception.

The user interface testing that was done within the application has a few methods that were automatically generated by the tools that are used within the user interface tests.

One of the tests that was recorded was to see if the button adds the number to the account. Below is a screenshot of what happens after the recording has been done and you have clicked the generate code button. Underneath that screenshot is another one that shows the method that was created, which holds all of the information within that recorded test.



Below is a screenshot of the code that was generated within the adding to account method, which is found within the UI map designer file.

```
public partial class UIMap
{
    /// <summary>
    /// AddingToAccount - Use 'AddingToAccountParams' to pass parameters into this method.
    /// </summary>
    /// <references>
    public void AddingToAccount()
    {
        #region Variable Declarations
        upfedit uAddTextBoxEdit = this.UIMainWindow.Window.UIAddTextBoxEdit;
        upfbutton uAddButton = this.UIMainWindow.Window.UIAddButton;
        #endregion

        // Click "addTextBox" text box
        Mouse.Click(uAddTextBoxEdit, new Point(28, 10));

        // Type '2' in 'addTextBox' text box
        uAddTextBoxEdit.Text = this.AddingToAccountParams.UIAddTextBoxEditText;

        // Click "Add" button
        Mouse.Click(uAddButton, new Point(61, 13));
    }
}
```

There was an error within creating this part of the test, unfortunately when running this user interface test, there was an error that came up, which it says that the playback failed to find the control that has the given search properties. Upon further research about this error, there was not a way that could be found that will fix this error.

Evaluation

With the data that was collected from the literature, it seems that both unit testing and user interface testing are both the best form of testing in order to test mobile applications within WPF. There are some conclusions that can be made from this data that was collected. It seems that user interface testing is the best form of software testing to use when building a mobile application within WPF. Unit testing is also a really good technique to use but from the data collected within the implementation of this project, it seems that user interface testing is the best and easiest technique to use to test mobile applications within WPF. This is due to not only can the developer test the interface of the mobile application, but they can also give the application to the user which they can then test the different aspects of the interface. The code can then be automatically generated to test the actions that the user has performed within the application. Unit testing is also very useful and important within testing mobile applications because the developer is able to test to see if every function within the application is working correctly. This is a very good method to wipe away any errors or bugs that may occur within the application.

The data within the implementation part of this project shows that the data does back up the research gathered within the literature review. The literature review talks about both unit and user interface testing, which is both white box and black box testing. The research gathered shows that the best way to test any mobile application that is created within WPF, is to use both of these testing techniques. There are other techniques that are important and that can be used to make sure the mobile application works the best it can and one of these techniques is the security testing. Security testing is very important when it comes to keeping the information of the user safe. Even though this technique was not mentioned within the design and implementation part, it is still a really good technique that should be used in order to make sure that the information of the user is kept safe, where no hackers can break into your code and take personal information about the user.

The objectives within this project were to have the best methods of software testing within a mobile application that was created within WPF. Even though the objectives of this project were met, there could still be room for improvement. There could be a few more techniques that were covered that could have been used, such as security testing, which was previously mentioned.

The aim of this project was to find out which methods of software testing were the best to use within a WPF mobile application. The aim of the project has been met to an extent. There could be room for improvement, maybe even a few more software testing techniques that could be used.

Overall, the answers that were expected from this project were correct to a certain point. Yes, the techniques that were used worked correctly and they did stop all bugs and errors within the application, but there was room for more complex testing such as security testing, but that can go into more of the networking side, rather the mobile application testing side.

If this project was to be done again, the only thing that would be done differently is adding one or two more software testing techniques, such as performance testing and tests to check to see if the application is responsive and runs on every device. With these techniques, there are also very important testing methods such as security testing that will need to be used just to make sure that no one without authorisation has access to personal information of the user.

Conclusions and Recommendations

This project has gone through what bugs, verification and validation are, as well as the different techniques that can be used to test mobile applications that are created within WPF. It has been through the overall best methods of software testing within a mobile application created in WPF, which were both unit and user interface testing. There were two development methodologies that were used within this project, which were both the waterfall and agile methodologies. After that was the requirements analysis, which is about the different types of hardware and software to use in order to build the application.

Then the research methodology was next, which was the research onion and it is about the different layers of research that can be used in order to find out information about software testing mobile applications that were created in WPF.

The next part was the design and implementation, which the design was about what processes that were used, as well as the different factors that were going to be used and what the interfaces were going to look like and what they were going to do. The design was also about building the application, while at the same time building the multiple unit and user interface tests that would make sure that the application runs without any errors or bugs.

The implementation was next, which is about putting everything together and making decisions on the application, like what the interface is, what systems models were created, and also any problems or successes within creating the code.

Lastly then was the evaluation which sums up everything that the application had and all of the tests that were done within the application, and also if there is any room for improvements. The improvements mentioned was about

There could be a few areas of research within this project that would be ideal for future improvements. There could be more research based around security testing, which will make the application a lot more secure from people trying to hack into the program to gain access to personal information. There could also be more research done on how software testing works with screen sizes of mobile applications, because there could be a hidden bug or error within the mobile application that could make things difficult for the user. An example of this would be if the user had the application working on most mobile devices, but there could be a hidden bug that messes with the screen sizes of certain mobile devices. The application could work perfectly for an older generation iPhone, but the application might look all over the place for a newer generation of an iPhone, with the textboxes and buttons placed everywhere around the screen. This causes a lot of trouble for the user because they might even overlay certain objects within the interface, such as a textbox appearing over most of the button. This means that the user will not be able to click that button.

Reflection

While in the process of creating the project, there were a few things that were learned. At the beginning of this project the aim was to quite simply find some of the best methods of software testing for a mobile application that has been created within WPF. The research had shown that some of the best methods of software testing were both unit and user interface testing. Unit testing was found out to be one of the methods that form a part of something called white box testing. Whereas the user interface testing was found out to be one of the methods that form a part of something called black box testing. These two forms of testing were found out to be a couple of the best forms of software testing when it comes to mobile applications.

There were a few other forms of testing that was not used within this application, which could have been put into place to make this application better. Security testing was one of the methods of testing that was very important when it comes to building professional mobile applications. This is because it stops a lot of threats such as hackers, from easily accessing the application in order to get personal information about the user, such as their bank account details, if they have been stored within the database of that application.

There is another form of testing that could have been implemented was the ability to check to see if the application itself is responsive and works for all devices. If the application will only work for some devices then that would not be good for the user. The application could work great on one device, but if there was a newer mobile device that had come out which has a different screen size, then the application might not fit to that screen size. If it does not fit, then the interface within the application might not even work. The buttons might be overlaid with the textboxes, which the user might not be able to click the button.

Performance testing is another form of mobile application software testing that should be used, because not only will it test the client applications performance, but it also tests the server and network performance to see if everything is running smoothly. This form of testing can also run responsive tests to see if the application can work across any platform, whether it is a mobile, tablet or a laptop.

If this project could be re-done, then these would be the options to put in place with the two other forms of software testing that has already been used. With these other options put in place it would make sure that not only is the application running without any errors or bugs, but it is also very secure and is running at a good speed on any device.

References

- [1] Patton, R. (2001). *Software Testing. Zhurnal Eksperimental'noi i Teoreticheskoi Fiziki*. Retrieved from <http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:No+Title#0>. Page 19
- [2] Patton, R. (2001). *Software Testing. Zhurnal Eksperimental'noi i Teoreticheskoi Fiziki*. Retrieved from <http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:No+Title#0>. Page 41
- [3] Patton, R. (2001). *Software Testing. Zhurnal Eksperimental'noi i Teoreticheskoi Fiziki*. Retrieved from <http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:No+Title#0>. Page 15
- [4] http://www.sqa.org.uk/e-learning/SDPL03CD/page_16.htm, verification and validation, 5/10/2015
- [5] http://www.tutorialspoint.com/software_testing_dictionary/verification_testing.htm, Verification Testing, 6/10/2015
- [6] Selby, R. W. R. W. R. W., & Basili, V. R. (1987). Comparing the Effectiveness of Software Testing Strategies. *IEEE Transactions on Software Engineering*, SE-13(12), 1278–1296. <http://doi.org/10.1109/TSE.1987.232881>. Pages one and two
- [7] Patrick Fan, <http://asmarterplanet.com/mobile-enterprise/blog/2014/09/five-essential-types-mobile-app-testing-help-woo-user.html>, Five essential types of mobile app testing in order to woo the user, September 15th 2014
- [8] Patrick Fan, <http://asmarterplanet.com/mobile-enterprise/blog/2014/09/five-essential-types-mobile-app-testing-help-woo-user.html>, Five essential types of mobile app testing in order to woo the user, September 15th 2014
- [9] A.Malini, N.Venkatesh, Dr.K.Sundarakantham, Dr.S.Mercyshalinie, Of, T. (n.d.). Mobile Application Testing on Smart Devices Using MTAAS Framework in Cloud. Page two
- [10] Mustafa, K. M., Al-Qutaish, R. E., & Muhairat, M. I. (2009). Classification of Software Testing Tools Based on the Software Testing Methods. *2009 Second International Conference on Computer and Electrical Engineering*, 229–233. <http://doi.org/10.1109/ICCEE.2009.9>. Pages one and two

- [11] Mustafa, K. M., Al-Qutaish, R. E., & Muhairat, M. I. (2009). Classification of Software Testing Tools Based on the Software Testing Methods. *2009 Second International Conference on Computer and Electrical Engineering*, 229–233. <http://doi.org/10.1109/ICCEE.2009.9>. Page two
- [12] Mustafa, K. M., Al-Qutaish, R. E., & Muhairat, M. I. (2009). Classification of Software Testing Tools Based on the Software Testing Methods. *2009 Second International Conference on Computer and Electrical Engineering*, 229–233. <http://doi.org/10.1109/ICCEE.2009.9>. Page two
- [13] [https://msdn.microsoft.com/en-us/library/aa292197\(v=vs.71\).aspx](https://msdn.microsoft.com/en-us/library/aa292197(v=vs.71).aspx), Unit Testing, 08/10/2015
- [14] [https://msdn.microsoft.com/en-us/library/aa730844\(v=vs.80\).aspx](https://msdn.microsoft.com/en-us/library/aa730844(v=vs.80).aspx), Guidelines for Test-Driven Development, 08/10/2015
- [15] <https://www.versionone.com/agile-101/agile-software-programming-best-practices/refactoring/>, Refactoring, 08/10/2015
- [16] Margaret Rouse, <http://www.softwaretestingclass.com/regression-testing-definition/>, regression testing, 08/10/2015
- [17] <http://softwaretestingfundamentals.com/test-plan/>, TEST PLAN Fundamentals, April 16, 2011
- [18] http://www.tutorialspoint.com/software_testing_dictionary/white_box_testing.htm, White box Testing, 10/10/2015
- [19] <http://www.softwaretestingclass.com/white-box-testing/>, What is a White Box Testing, 11/10/2015
- [20] <http://www.softwaretestingclass.com/white-box-testing/>, What is a White Box Testing, 11/10/2015
- [21] <http://www.softwaretestingclass.com/white-box-testing/>, What is a White Box Testing, 11/10/2015
- [22] <http://softwaretestingfundamentals.com/black-box-testing/>, BLACK BOX TESTING Fundamentals, December 19, 2010
- [23] http://www.webopedia.com/TERM/B/Black_Box_Testing.html, Black Box Testing, 13/10/2015
- [24] <http://www.softwaretestinghelp.com/black-box-testing/>, Black Box Testing: Types and techniques of BBT, November 3, 2015
- [25] <http://technologyconversations.com/2013/12/11/black-box-vs-white-box-testing/>, Black-box vs White-box Testing, 10/11/15
- [26] <http://technologyconversations.com/2013/12/11/black-box-vs-white-box-testing/>, Black-box vs White-box Testing, 10/11/15

- [27] [https://msdn.microsoft.com/en-us/library/aa970268\(v=vs.100\).aspx](https://msdn.microsoft.com/en-us/library/aa970268(v=vs.100).aspx), Introduction to WPF, 10/11/15
- [28] Patton, R. (2001). *Software Testing. Zhurnal Eksperimental'noi i Teoreticheskoi Fiziki*. Retrieved from <http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:No+Title#0>. Page 34
- [29] <http://www.base36.com/2012/12/agile-waterfall-methodologies-a-side-by-side-comparison/>, Agile & Waterfall Methodologies – A Side-By-Side Comparison, 05/02/16
- [30] <http://istqbexamcertification.com/what-is-waterfall-model-advantages-disadvantages-and-when-to-use-it/>, What is Waterfall model-advantages, disadvantages and when to use it?, 05/02/16
- [31] <http://istqbexamcertification.com/what-is-waterfall-model-advantages-disadvantages-and-when-to-use-it/>, What is Waterfall model-advantages, disadvantages and when to use it?, 05/02/16
- [32] http://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm, SDLC – Waterfall Model, 05/02/16
- [33] http://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm, SDLC – Waterfall Model, 05/02/16
- [34] <http://www.base36.com/2012/12/agile-waterfall-methodologies-a-side-by-side-comparison/>, Agile & Waterfall Methodologies – A Side-By-Side Comparison, 10/02/16
- [35] <http://istqbexamcertification.com/what-is-agile-model-advantages-disadvantages-and-when-to-use-it/>, What is Agile model-advantages, disadvantages and when to use it?, 10/02/16
- [36] <http://www.my-project-management-expert.com/the-advantages-and-disadvantages-of-agile-software-development.html>, The Advantages and Disadvantages of Agile Software Development, 10/02/16
- [37] <http://www.base36.com/2012/12/agile-waterfall-methodologies-a-side-by-side-comparison/>, Agile & Waterfall Methodologies – A Side-By-Side Comparison, 10/02/16
- [38] <http://www.my-project-management-expert.com/the-advantages-and-disadvantages-of-agile-software-development.html>, The Advantages and Disadvantages of Agile Software Development, 10/02/16
- [39] UK Essays. November 2013. Explanation Of The Concept Of Research Onion Psychology Essay. [online]. Available from: <https://www.ukessays.com/essays/psychology/explanation-of-the-concept-of-research-onion-psychology-essay.php?cref=1> [Accessed 4 April 2016].

[40] UK Essays. November 2013. Explanation Of The Concept Of Research Onion Psychology Essay. [online]. Available from: <https://www.ukessays.com/essays/psychology/explanation-of-the-concept-of-research-onion-psychology-essay.php?cref=1> [Accessed 4 April 2016].

[41] UK Essays. November 2013. Explanation Of The Concept Of Research Onion Psychology Essay. [online]. Available from: <https://www.ukessays.com/essays/psychology/explanation-of-the-concept-of-research-onion-psychology-essay.php?cref=1> [Accessed 4 April 2016].

[42] UK Essays. November 2013. Explanation Of The Concept Of Research Onion Psychology Essay. [online]. Available from: <https://www.ukessays.com/essays/psychology/explanation-of-the-concept-of-research-onion-psychology-essay.php?cref=1> [Accessed 4 April 2016].

[43] UK Essays. November 2013. Explanation Of The Concept Of Research Onion Psychology Essay. [online]. Available from: <https://www.ukessays.com/essays/psychology/explanation-of-the-concept-of-research-onion-psychology-essay.php?cref=1> [Accessed 4 April 2016].

[44] UK Essays. November 2013. Explanation Of The Concept Of Research Onion Psychology Essay. [online]. Available from: <https://www.ukessays.com/essays/psychology/explanation-of-the-concept-of-research-onion-psychology-essay.php?cref=1> [Accessed 4 April 2016].

Appendices

There is a use case diagram below that shows how the user will interact with the interface and how the software tests that were created are used to show how everything is being tested.

