**Student Template for Software Architecture Skills Demonstrations 1-7**

**Section A: Skills Demonstration 2 \_ Section A is to be completed individually.**

**Describe the role of modern methodologies in the software development process**

1. Discuss the limitations of traditional SDLC methods such as Waterfall and describe the evolution of Agile methodologies with regard to how they overcome these problems. You may use relevant diagram(s) to assist your answer.

|  |
| --- |
| Discuss the limitations of traditional SDLC methods such as Waterfall |
| In the traditional SDLC method format there was limitations that hampered the development of a complete project because there was too much emphasis the planning stage and not on creating a fully operational product. If you have a big product it will take a long period of time to produce the product and this could be not sufficient because of the evolution of new software and programs. If the demands on the product changes it does not give you a chance to change it to comply with the new demands.  E:\Laura\Software Architecture\ASSIGNMENTS\SuperServicesLTD\Waterfall Diagram.png  Another limitation is the lack of testing the product while you are designing it could leave you with a product that has a lot of problems based on a few things early on in its design process but were not found due to lack of testing. These limitations in the method led to a new style of methodology to be created called Agile. |
| Describe the evolution of Agile methodologies with regard to how they overcome these problems. |
| The Agile SDLC method does not put a big emphasis of planning but on the design process of the product itself and the frequent testing leads to bugs in the product being discovered early on, thus making it easy to rectify. The process has more meetings with the customer to get the customer more involved in the process to make sure that the product being created is what he/she wants.    The split up the product into a few iterations and each iterations have-a Planning stage, a requirement analysis stage, a building phase, a designing stage and a testing stage. Each iteration usually last up to three weeks. This also means that the product will not take a long time to create and lets the customer know what he is buying and spending money on producing at frequent intervals. Each point here counteracts the limitations in the traditional SDLC method listed above. |

1. Briefly discuss the main principles behind the Agile manifesto.

|  |
| --- |
| Main Principles behind the Agile Manifesto |
| We are uncovering better ways of developing software by doing it and helping others do it?  Through this work we have come to value;   * “Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.”   It values the customer’s satisfaction over any other thing. It believes the customer must be happy for the project to be a success.   * “Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.”   It welcomes the customers input even if it changes the software because why build something that the customer does not want.   * “Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.”   The new method wants to build the project at a fast pace and not waste time spending up to 1 year building a program that could be obsolete when you deliver it.   * “Business people and developers must work together daily throughout the project.”   It wants continuous input from the customer because at the end of the day you are building it for the customer and they need to like and agree with the program for it to be a successful working program.   * “Build projects around motivated individuals.  Give them the environment and support they need, and trust them to get the job done.”   Do not work with people who are not enthused with the project or it will not work and it will not create a good, vibrant working environment. Make sure you encourage your fellow workers.   * “The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.”   It believes that the most effective way of getting information of value from a customer is by having a face to face conversation. It believes email or by phone does not give off the same vibe and will not have the same effect as a face to face meeting would.   * “Working software is the primary measure of progress.”   Unlike the traditional method the most important aspect of progression in the project will be have working software for each stage.   * “Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.”   It wants the work of each member of the project to coincide with one another to make sure you are developing the project at a pace that does not change or vary. The process of development is done over a long period of time with complete focus and the developers have no drop off in production.   * “Continuous attention to technical excellence and good design enhances agility.”   The design phase of the project is an important feature and nothing short of an excellent design will prove a success for the project.   * “Simplicity--the art of maximizing the amount of work not done--is essential.”   Only code when you have to. Do not write lines of code that do not have any impact on the resulting project.   * “The best architectures, requirements, and designs emerge from self-organizing teams.”   Being able to self-organise will give you the best tool for learning and thus make the team have the best members for their role.   * “At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly.”   There must be regular meetings between each team member to make sure the team is functioning well and completing the project on the correct way. |

1. Briefly describe the Scrum and Extreme Programming (XP) methodologies.

|  |  |
| --- | --- |
| Description of Scrum Method | Description of Extreme Programming Method |
| The Scrum method has people have the awareness to be self-managed. They know what they must do, they do not need a major plan to complete the work needed for the project. It relies heavily on team interaction, communication and self-organization. | Extreme Programming method is when the results are more important than what you code.Designed primarily for benefiting the business’s above all else. Uses continuous coding implementations and frequent testing in order to meet demands sooner and to improve the projects overall qualities. |

**Section B: Skills Demonstration 1 \_**

**Engage a modern methodology such as the Agile Scrum to aid the design of a system from initial requirements**

1. Identify the software development methodology used by your group:

**The software methodology used for this group project is the Agile Scrum method.**

1. Identify the role fulfilled by each team member e.g.:

|  |  |  |
| --- | --- | --- |
| **Student Name** | **Development role** | **What did the role entail** |
| **Steven** | Product Owner | I gave an overview of what is needed and wanted for the new program to be a functioning model. List the requirements that needed to be met for it to reach the standard required. |
| **Gavin** | Scrum Master | As the Scrum Master I had to organise the team and make what the product owner wanted a reality. As a scrum master I took on a leadership role in the group. |
| **Laura** | Development Team member 1 | Working together with team members in order to create a functional and correctly working program based on the needs of the product owner. |
| **Robert** | Development Team member 2 | Work as a team in order to create a functioning program that has all the requirements listed based on the needs of the product owner. |

1. Create a product backlog

|  |  |  |
| --- | --- | --- |
| **Product Backlog** | | |
| **User Story ID** | **User Story** | **Priority** |
| **U001** | Initial analysis phase | 1 |
| **U002** | UML Use Case | 8 |
| **U003** | UML Class Diagram | 2 |
| **U004** | Generate the code from UML Class Diagram | 3 |
| **U005** | GUI build | 4 |
| **U006** | Calculate functionality | 5 |
| **U007** | Clear Form | 6 |
| **U008** | Buy Quote | 7 |

1. Record the work undertaken during each stage of the software development (i.e. each sprint requires analysis, development and testing). The skills demonstration should include at least 4 sprints each with its own sprint backlog.

**Sprint One: An initial analysis phase to analyse the existing systems and to produce UML (use cases and class diagrams) for the overall system**

**Record of Scrum Meeting for Sprint One**

|  |  |
| --- | --- |
| **Date of meeting:** | **23rd January 2017** |
| **Names of those present at meeting:** | Gavin , Laura and Steven |
| **Name of student writing report for this Scrum meeting:** | Gavin |
| **User Story documents for the sprint / Deliverables** | Produce UML(use cases and class diagrams)  Assigned Roles  Assigned estimated deadlines for sprints |
| **Daily Scrum Meeting Summary** | Steven is the product owner. Gavin is the scrum master. Laura and Robert are team members. Gavin and Robert will do the use cases diagram and Laura and Steven will do the class diagrams. We decided to give 1 week for sprint one, 3 weeks for sprint 2, a week and a half for sprint 3, a week and a half for sprint 4. |
| **Planning and Analysis of user stories** | To have the UML Class and case diagrams prepared for the next sprint meeting |
| **Outcome of Development** | See Sprint backlog |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sprint 1 Backlog: Compiled by: Gavin** | | | | | |
| Requirement | Task | Estimate (hours) | Who | Status | Remaining Work |
| Analyse initial Programs | Look at Munster Motor Insurance program | 1 | Laura | Completed | 0 |
| Look at the original Slainte Health Insurance Ltd | 1 | Steven | Completed | 0 |
| UML, USE case | Choose Program | 1 | Gavin | Completed | 0 |
| Design Use cases | 2 | Gavin | Completed | 0 |
| Link up Use cases to customer and relevant cases | 1 | Gavin | Completed | 0 |
| UML, CLASS Diagram | Choose Program | 1 | Steven, Laura | Completed | 0 |
| Create Customer class | 2 | Steven | Completed | 0 |
| Create Policy class | 2 | Laura | Completed | 0 |
| Create individual policy | 2 | Steven | Completed | 0 |
| Create corporate policy | 2 | Laura | Completed | 0 |
|  |  |  |  |  |  |
| Total Estimate & Remaining: | | 15 |  |  | 0 |

**Sprint Two: Create java project with classes generated from UML. Design and develop the Swing form with all the necessary GUI elements**

**Record of Scrum Meeting for Sprint Two**

|  |  |
| --- | --- |
| **Date of meeting:** | **Thursday 26th January 2016** |
| **Names of those present at meeting:** | **Laura, Gavin, Steven** |
| **Name of student writing report for this Scrum meeting:** | **Steven** |
| **User Story documents for the sprint / Deliverables** | Generating the code from the UML Class Diagram.  Building the base of the software. |
| **Daily Scrum Meeting Summary** | Organising who will do what job on the next Sprint.  Generate the code-  Design layout of GUI-Gavin  Policy- Everyone  Individual Policy-Steven  Corporate policy- Steven  Slainte Health-Robert  Customer- Laura |
| **Planning and Analysis of user stories** | Gavin |
| **Outcome of Development** | See Sprint backlog |
| **Testing** | Steven and Gavin |

**User Story *{GUI build}***

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

As an *{Employee}*,

I want *{A form that will store customer information}*

So that *{the company can provide policy quotes faster.}*.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sprint 2 Backlog: Compiled by:** | | | | | |
| Requirement | Task | Estimate (hours) | Who | Status | Remaining Work |
| Generate the  Code from UML Class Diagram | Create folder for code | 1 | Steven | Done | 0 |
| Generate code from UML | 1 | Steven | Done | 0 |
| Customer | 8 | Laura | Done | 0 |
| Slainte Health | 3 | Robert | Done | 0 |
| Policies | 3 | Steven | Done | 0 |
| Testing | 6 | Gavin/Steven | Done | 0 |
|  |  |  |  |  |
| Create layout of  the GUI | Open up a Java Swing GUI Form- JFrame | 1 | Gavin | Done | 0 |
| Title the program | 1 | Gavin | Done | 0 |
| Create 4 Jpanels, name them | 1 | Gavin | Done | 0 |
| Insert correct palette items Into proper position. | 3 | Gavin | Done | 0 |
|  |  |  |  | 0 |
|  |  |  |  |  |  |
| Total Estimate & Remaining: | | 28 |  |  | 0 |

**Sprint Three: Design and develop the 'calculate quote' functionality.  (Including form validation and test case for over 70s).**

**Record of Scrum Meeting for Sprint Three**

|  |  |
| --- | --- |
| **Date of meeting:** | **9/03/17** |
| **Names of those present at meeting:** | **Steven, Gavin, Laura, Robert** |
| **Name of student writing report for this Scrum meeting:** | **Laura** |
| **User Story documents for the sprint / Deliverables** | Fixing up the front end of the Jpanel form, and completing the calculate button coding |
| **Daily Scrum Meeting Summary** | deciding to split the panel segments equally  Robert and Laura will fill in details for Personal Details and Health Details.  Gavin and Steven will fill in Scheme Details and calculate button functionality |
| **Planning and Analysis of user stories** | Laura |
| **Outcome of Development** | See Sprint backlog |
| **Testing** | Robert and Laura- Personal details and health details testing.  Gavin and Steven- Scheme and calculate buttons testing. |

**User Story *{Calculate Functionality}***

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

As an *{Employee}*,

I want *{a button that will calculate all the calculations within policies with one click}*

So that *{policy calculations are faster, and more accurate}*.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sprint 3 Backlog: Compiled by:** | | | | | |
| Requirement | Task | Estimate (hours) | Who | Status | Remaining Work |
| Writing the code for the front end | Coding the Personal Details | 1 | Robert | Completed | 0 |
| Coding the Scheme details | 1 | Gavin | Completed | 0 |
| Coding the Health Details | 1 | Steven | Completed | 0 |
| Coding the calculate button | Validate all inputted form data | 2 | Laura | Completed | 0 |
| Create a customer | 5 | Laura | Completed | 0 |
| Input data from form into the customer | 1 | Laura | Completed | 0 |
| Get the calculations from policy file | 2 | Laura | Completed | 0 |
|  |  |  |  |  |  |
| Total Estimate & Remaining: | | 13 |  |  | 0 |

**Sprint Four: Design and develop ‘new quote/ clear form’ and ‘buy quote’ functionality (including testing)**

**Record of Scrum Meeting for Sprint Four**

|  |  |
| --- | --- |
| **Date of meeting:** | **23/03/2017** |
| **Names of those present at meeting:** | **Robert, Gavin, Steven, Laura** |
| **Name of student writing report for this Scrum meeting:** | **Robert** |
| **User Story documents for the sprint / Deliverables** | Making clear form and buy policy buttons to work. |
| **Daily Scrum Meeting Summary** | Clear form button not working. Robert Laura and Gavin fixed the clear form button to make it working. Laura got ID No. to work. |
| **Planning and Analysis of user stories** | Clear Form and Buy Policy buttons to be done for next scrum meeting and to check if program is fully functioning. |
| **Outcome of Development** | Robert and Steven |
| **Testing** | Laura tested the clear form button.  Steven tested the buy Policy button. |

**User Story *{Buy Policy}***

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

As a *{customer}*,

I want *{to buy a policy type}*

So that *{By clicking once on the button I get the policy type that suits my information.}*.

**User Story *{Clear Form}***

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

As a *{customer}*,

I want *{to clear a form with the button}*

So that *{I can re-enter the information if I made a mistake}*.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sprint 4 Backlog: Compiled by:** | | | | | |
| Requirement | Task | Estimate (hours) | Who | Status | Remaining Work |
| Clear Form | Coding for clearing the personal Details | 1 | Robert | Completed | 0 |
| Coding for clearing the Health Details | 2 | Laura | Completed | 0 |
| Coding for clearing the scheme Details | 2 | Laura | Completed | 0 |
| Coding for clearing the Final Quotation | 2 | Robert | Completed | 0 |
| Buy Policy | Design the GUI buy policy window | 3 | Gavin/  Steven | Completed | 0 |
| Coding the buy policy window | 3 | Steven/  Gavin | Completed | 0 |
|  |  |  |  |  |  |
| Total Estimate & Remaining: | | 13 |  |  | 0 |

1. Identify the main tools used by your team: Netbeans- GUI class form, Java Application, easyUML, UML Class Diagrams project and starUML.

**Section C: Skills Demonstration 3**

**Apply the main tools and techniques used in the gathering, recording and analysis of information relating to an existing information system**

**Observation:**

1. Record your main observations regarding the existing health insurance program below:

* The existing health insurance program is not designed in a GUI format which means it has no link to object oriented programming.
* It is all on one page meaning there is not a customer or policy class.
* All the data in inputted in the output box that reads in the code while you run your program.
* The program is not designed in a creative manner. It is not an appealing program to look at and the format is not designed in a way to draw in the user.
* It is not easy to input data into the program because of its limited design.

1. *Record your main observations regarding the existing GUI motor insurance program below:*

* The GUI Layout is simplistic and easy for customers to fill out.
* The program checks for input as it displays dialog windows if a text box remains empty when clicking calculate.
* New quote increments the Quote number at the top of the program.
* Summary opens up a list of previous customers.
* Customers who choose the 80s age group have a dialog window explaining that there’s no cover for this age group.

**Document Analysis:**

1. *Analyse the User Manual Documentation provided for the Munster Motor Insurance GUI Program and record your main findings below:*

* The existing health insurance program is not designed in a GUI format which means it has no link to object oriented programming.
* It is all on one page meaning there is not a customer or policy class.
* All the data in inputted in the output box that reads in the code while you run your program.
* The program is not designed in a creative manner. It is not an appealing program to look at and the format is not designed in a way to draw in the user.
* It is not easy to input data into the program because of its limited design.
* The amount of cars listed is not a broad overview of the car industry and does not show a variety of car makes or models.

**Section D: Skills Demonstration 5**

**Use straightforward UML use‐case diagrams to depict the interaction between an actor and system**

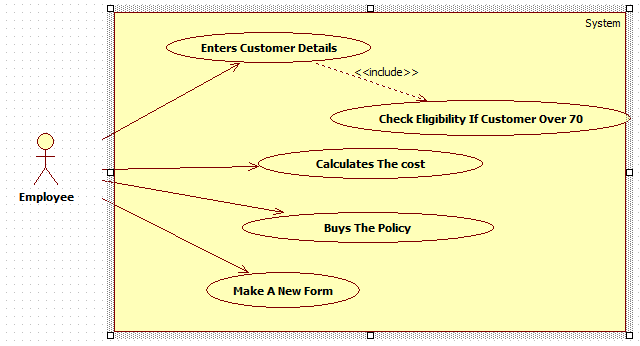
1. Briefly explain what a Use Case is:

A series of steps used to plan a course of action and to define the roles of each individual segment. It usually has one or more actors and a system for values to be pre-defined and achieved.

1. Briefly explain the purpose of a Use Case Diagram is:

The purpose of use case diagrams is to display the main goals achievable by the system and to show the users of this system.

1. Create a straightforward **UML use-case diagram** using a suitable tool to model the workings of your software.

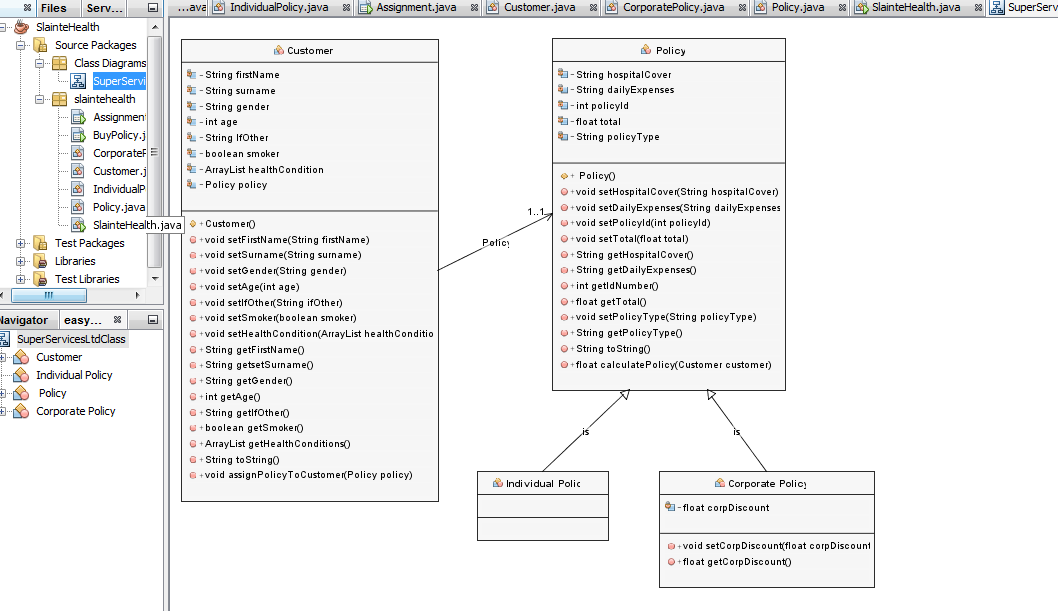


**Section E: Skills Demonstration 4**

**Generate Java (or suitable Object‐ Oriented language) classes from UML class diagram specifications**

In this skills demonstration you are required to design and develop a piece of Software using an Agile Software Development Methodology.

1. Create a straightforward **UML class diagram** using a suitable tool to model the design for your software based on the problem statement provided. Make sure to show the relevant relationship e.g. inheritance between the classes



1. Using your UML class diagram **forward engineer** to generate and save Java Code from your UML class diagram specification. Print out a copy of this unedited generated code. Once this task is completed.

See printed code.

**Section F: Skills Demonstration 6**

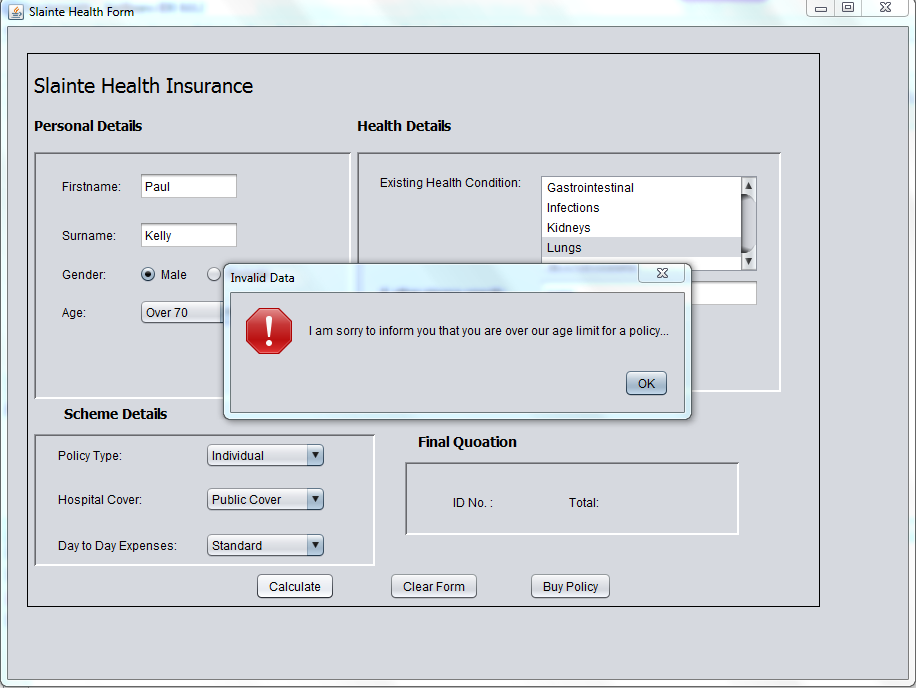
**Test the design specification of a new information system based on a provided use‐case**

1. **Create** and **execute** a test case for your software using the use-case provided in your brief:

|  |  |  |
| --- | --- | --- |
|  | **Project Name: SlainteHealth**  **Test Case ID:** Health\_1.1 | **Test Designed by:** Gavin |
|  | **Test Priority (Low/Medium/High):** Medium | **Test Designed date:** 24/03/2017 |
|  | **Module Name: Software Architecture 5N2772** | **Test Executed by: Steven** |
|  | **Test Title: Test for Over 70’s** | **Test Execution date:** 27/03/2017 |
|  | **Description: Test for quote for over 70’s.** |  |
|  | **Pre-conditions: The user is a registered user.** | |
| **Post Conditions: The user is given no quote and the quotation screen is reset to original values.** | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Step** | **Test Steps** | **Test Data** | **Expected Result** | **Actual Result** | **Status (Pass/Fail)** | **Notes** |
| 1 | Launch Application | Name: Paul Kelly Gender: Male Age:79 Existing Health Conditions: Lungs Other: no Smoker: no Hospital Cover: public Day to Day Expenses: standard Policy Type: corporate |  |  |  |  |
| 2 | input user data |  | System displays message:  "I am sorry to inform you, that you are over our age limit." | Total: 0 | Pass |  |
| 3 | Select Over 70 |  |  |  |  |  |
| 4 | Calculate button selected | Clears all data and creates a new policy id number for the next customer |  |  |  |  |
| 5 | Clear Form |  | All form options revert to original status | All form options are in their original form. | Pass |  |

1. Take relevant screen shots as evidence that you have ran the test case.



**Screen shots taken by:** Laura

**Section G: Skills Demonstration 7 \_ Section G is to be completed individually.**

**Explain the development of an information system using a traditional systems development life cycle model**

1. Use a traditional Software Development Life Cycle (SDLC**) to outline the stages** in the development of the current information system you have just developed using this approach **e.g. Waterfall SDLC**

For each stage in the methodology **explain the main activities that took place**

|  |  |
| --- | --- |
| **Identify name of Traditional Systems Development Life Cycle Model:** | Waterfall |
| **Identify the names of all relevant stages** | **Explain the main activities that took place during each stage** |
| System and Software Requirements | We had a meeting with the product owner to find out what is the system and software requirements. The system need to have a GUI form as well as a jDialog window. The code must be generated from a class diagram, contain customer and policy classes. The policy class must be linked to an individual policy and a corporate policy depending on user input. |
| Analysis | We analysed all the data we received from the owner to understand what we need to undertake as a group on this programme. We also looked at the existing programme and used this as a base to what user input is needed. |
| Design | We used our knowledge of the programme to design a template of what we want it to look like. We designed a UML class diagram and use case diagram.  2 GUI Forms were also created. |
| Coding | We generated code from the class diagram for the back end of the project. We also completed the back end before generating code for the GUI form with the back end. From the buttons, we designed a GUI form that displayed all the customer details. |
| Testing | After each sprint we tested our project. |
| Deployment | Once we tested the project and were satisfied that it was functioning we released it to the owner. |