Prathamesh Tarde



Finance Program

**Table of Contents**

|  |  |
| --- | --- |
| Page | Title |
| 2 | **Problem** |
| 2 | **Client Needs** |
| 2 | **Objectives** |
| 2 | **Issues** |
| 3 | **Boundaries** |
| 3 | **Quality Assurance** |
| 4 | **Feasibility Report** |
| 6 | **IPO Chart** |
| 8 | **Data Flow Diagram** |
| 10 | **Structure Charts** |
| 12 | **Data Dictionary** |
| 13 | **Storyboard** |
| 15 | **Gantt Chart** |
| 19 | **Log Book** |
| 21 | **Flowchart** |
| 23 | **Pseudocode** |
| 25 | **Source Code** |
| 31 | **Checking Software Solutions** |
| 32 | **Testing** |

**Defining and Understanding**

**Problem:**

You have been employed by a client to develop a program that will be implemented on their computer for them to use during their free time and when required. This software should be suitable for the client’s age. Social and ethical issues should be applied within the program.

**Client Needs:**

* Create a bank program which can only be accessed by me and replicates a real-life bank
* Within the bank program you should be able to make deposits and withdrawals
* Savings and checking type of accounts
* Superannuation calculator (both monthly and annually)
* Loan calculator (both monthly and annually)
* Program should be extremely easy to use
* Complete the program within the specified time period
* There should be no errors
* There should be a good user interface
* Program should be free of malware
* Be able to view transactions

**Objectives:**

* Fulfil all the needs of the client which are stated as above
* Comply to social and ethical issues such as
* No advertisement in the program
* No gender discrimination (even though the program is intended for one user)
* Allow user privacy by providing a username and password only for them
* Not plagiarise anyone else’s work

**Communication Issues:**

The needs of the client should be considered at all stages of the software development cycle and especially whilst defining and understanding the problem. In order to provide the client with the best possible software it is vital to maintain communication at a high standard. The forms of communication that was used with the client of the Finance Program were:

* Meetings
* Emails
* Concept prototypes
* Phone Calls
* Storyboards

**Compatibility Issues:**

Software of various types runs on a variety of operating systems, browsers, hardware and a range of different devices. When designing, software developers must ensure their products are compatible with the client needs. There will be no issues with the Financial Program with the operating system and device the client has requested.

* This software will not have any problems with different versions of the intended operating system
* There will be no issues when the underlying application is updated

**Performance Issues:**

Often the specifications of the computers used by developers far exceed those likely to be present in a typical user’s computer. In addition, multi-user applications and applications that access large files and databases will perform very differently under real world conditions. Testing environments and actual tests should simulate real world conditions.

* The computer responds after a function has been initiated and therefore there will be no issues
* Due to the small size of the program there are no issues while there are many applications open and therefore response times are extremely quick

**Boundaries of the Problem**:

Boundaries define the limits of the system to be developed. A boundary that the software has is:

* Whether the client actually has the funds that he transfers. Since the banking system uses mock money and is a way for the client to keep up with their own transactions. The client in some cases may not actually have the money they withdraw/deposit.

**Quality Assurance:**

Quality Assurance is a set of processes that occur throughout the development and manufacturing of any type of product in order to ensure standards and requirements are met and continue to be met such that the final products are of a high quality. The criteria that has been considered and met through the software development approach are:

* Efficiency
* Integrity
* Reliability
* Usability
* Accuracy
* Maintainability
* Testability
* Re-usability

**Feasibility Report**

**Executive Summary:**

The aim of this report is to take into consideration the problem, assess the different options that can be taken, provide the risks that may arise and provide recommendations that will help prevent the risks in order to determine whether the proposed solution is practical.

**Problem Statement:**

You have been employed by a client to develop a program that will be implemented on their computer for them to use during their free time and when required. This software should be suitable for the client’s age. Social and ethical issues should be applied within the program.

**Risk Assessment:**

**Technical:**

The hardware that is being used by the client is a 2016 Lenovo Yoga with its operating system being Windows 10. The software solution will be built using Visual Basic and will be an effective way to create GUI and code efficiently. If there were to be a change in hardware or software it will not have any effect on the efficiency of the program due to Visual Basic being adaptable to different software and hardware’s.

**Economic:**

The solution is affordable to both the client and the developer as Visual Basic is a free programming language and the client is being offered the solution for free.

**Scheduling:**

The developer has been given 6 weeks to complete the task which is a decent amount of time as the programmer is able to finish the task while not rushing which may affect the end result.

**Operational:**

The software will be usable by the client due to the use of a simple user interface which will benefit the user.

**Legal:**

The software solution will not plagiarise and will not break any Copyright Laws which are in place.

**Social/Ethical Considerations:**

The software will not

* Place advertisements within the program
* Gender discrimination (even though the program is intended for one user)
* Deny user privacy by providing a username and password only for them

**Recommendations:**

The program is feasibly and is worth solving and this was made clear through the use of the feasibility report. Recommendations for the developer are:

* Manage time efficiently in order to deliver the highest quality product
* Provide documentation which explores the different aspect of the program in detail

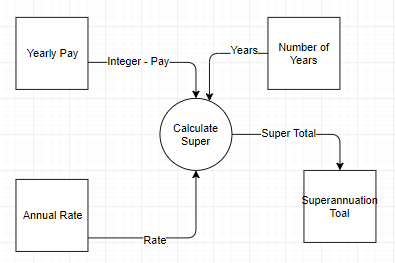
**Planning and Designing Software Solutions**

**IPO Chart**

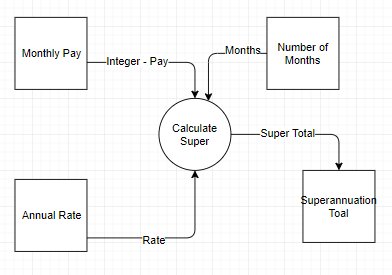
|  |  |  |
| --- | --- | --- |
| Input | Process | Output |
| Username and Password | Checks Username and Password to see if it’s correct | If username and password are correct it takes you to the next page and if they are not correct a message box pops up indicating that it is incorrect |
| Exit Button | Checks the code for the button | Ends the program |
| Superannuation Button | Checks the code for the button | Takes you to the superannuation page |
| Loans Button | Checks the code for the button | Takes you to the loan page |
| Bank Button | Checks the code for the button | Opens the bank page |
| Annually Button | Checks the code for the button | Opens the annually page |
| Monthly Button | Checks the code for the button | Opens the monthly page |
| Textboxes for Superannuation | Checks whether input is numerical | If the input is numerical it allows it if not a message box pops up indicating that a numerical value must be entered |
| Calculate Button | Runs the algorithm | Displays the end value |
| Check Boxes | Determine whether they have been selected and if only one has been selected | If neither has been selected, then a message box pops up indicating that the user needs to select one  If both have been selected, then a message box pops up indicating that only one can be chosen  If one has been selected, then it allows the calculate button to function |
| Logo | Checks the code one clicked upon | Takes the user to the homepage |
| Deposit Button | Checks the code | Adds the value into the balance and notes the date, deposit value into a grid |
| Withdrawal Button | Checks the code and checks if there are sufficient funds in the balance | If there aren’t sufficient funds in the balance a message box pops up indicating that there are insufficient funds  If there are sufficient funds, then the value is subtracted from the balance and is noted in the grid |
| Transfer Button | Checks the code and checks if there are sufficient funds in balance | If there aren’t sufficient funds in the balance a message box pops up indicating that there are insufficient funds  If there are sufficient funds, then the value is subtracted from the balance and is added to the other account balance and is noted in the grid |
| Sign Check | Checks the code and checks if there are sufficient funds in balance | If there aren’t sufficient funds in the balance a message box pops up indicating that there are insufficient funds  If there are sufficient funds, then the value is subtracted from the balance and is noted in the grid |
| Drop Down Box | Checks whether savings or checking has been selected | If savings has been selected the “check” section is not visible to the user and the balance and grid for checking appears  If checking has been selected the checking balance and grid is visible |

**Data Flow Diagram**

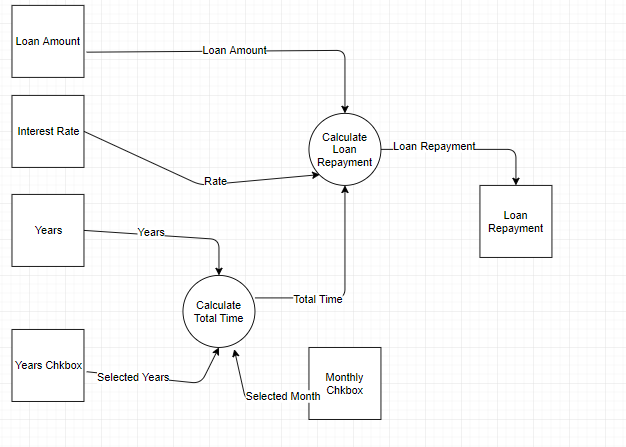
**Yearly Superannuation**



**Monthly Superannuation**

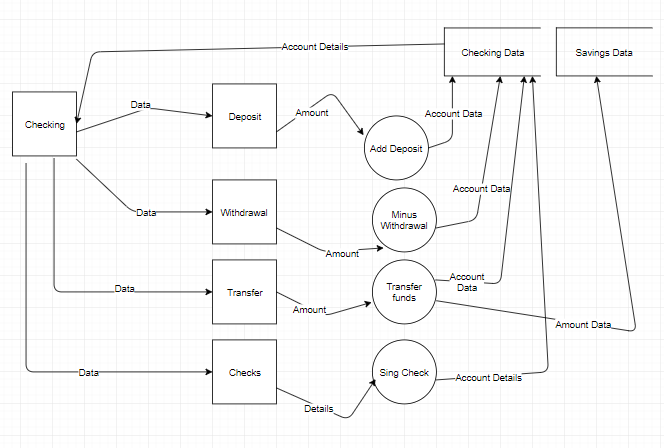


**Loans**

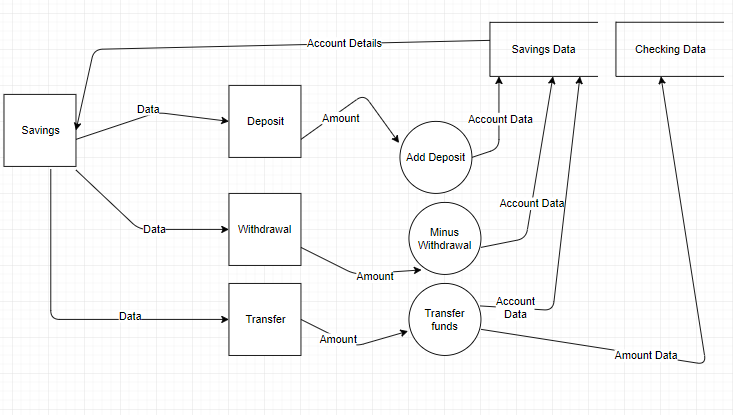


**Bank**

**Checking Account**

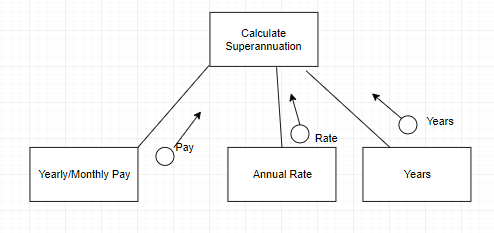


**Savings Account**

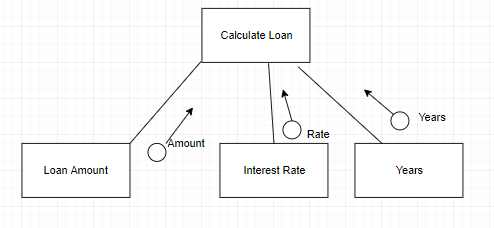


**Structure Chart**

**Superannuation**

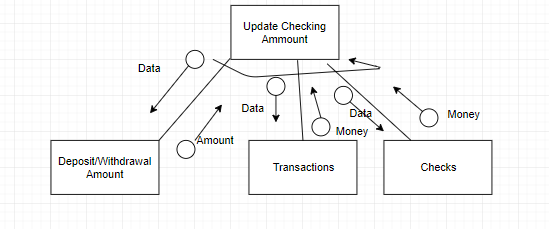


**Loans**

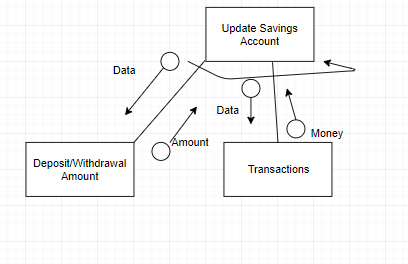


**Bank**

**Checking Account**



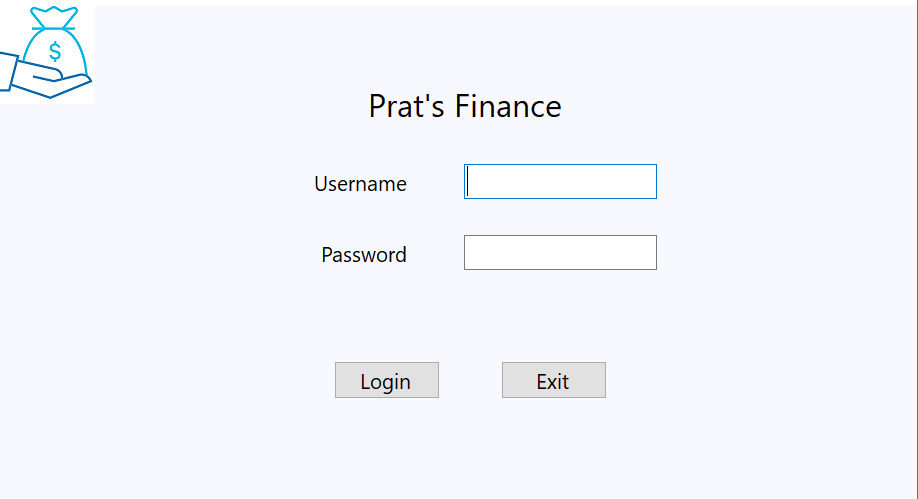
**Savings Account**



**Data Dictionary**

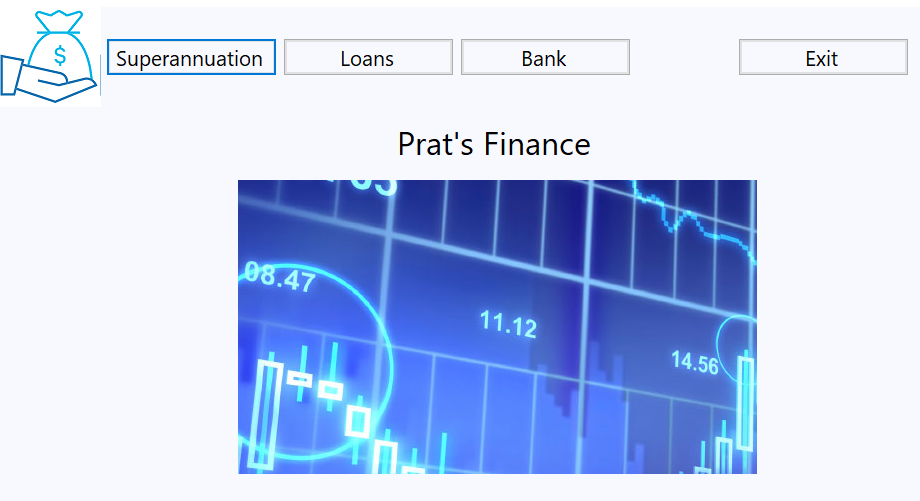
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Identifier | Data Type | Number of Storage Bytes | Description | Example |
| Deposit | Floating Point | 4 | Input parameter | 70,50 |
| Withdrawal | Floating Points | 4 | Input parameter | 70.6 |
| Transaction | Floating point | 4 | Input parameter | 50 |
| Checks | Floating Point | 4 | Input Parameter | 49.2 |
| Checking or Savings | Boolean | 8 | An option to chose | True |
| Monthly or Year | Boolean | 8 | An option to chose | False |
| Username or Password | String | 255 | User input | Prathamesh |

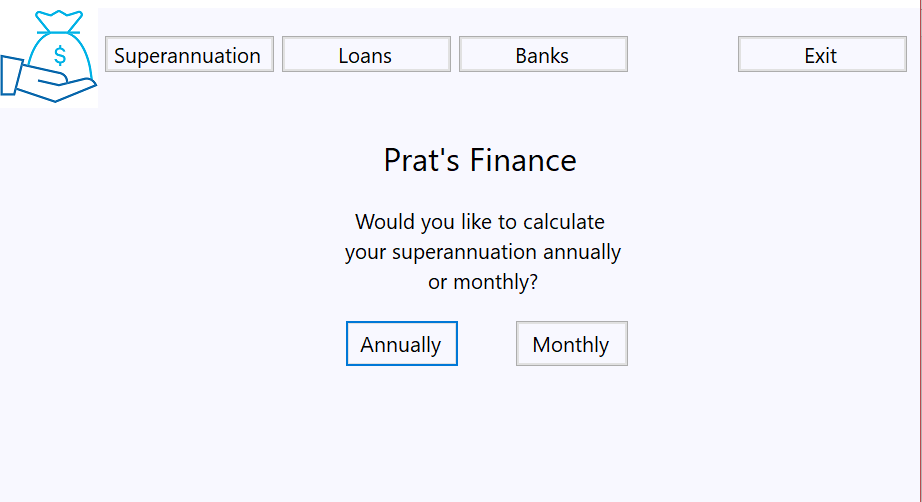
**Storyboard**



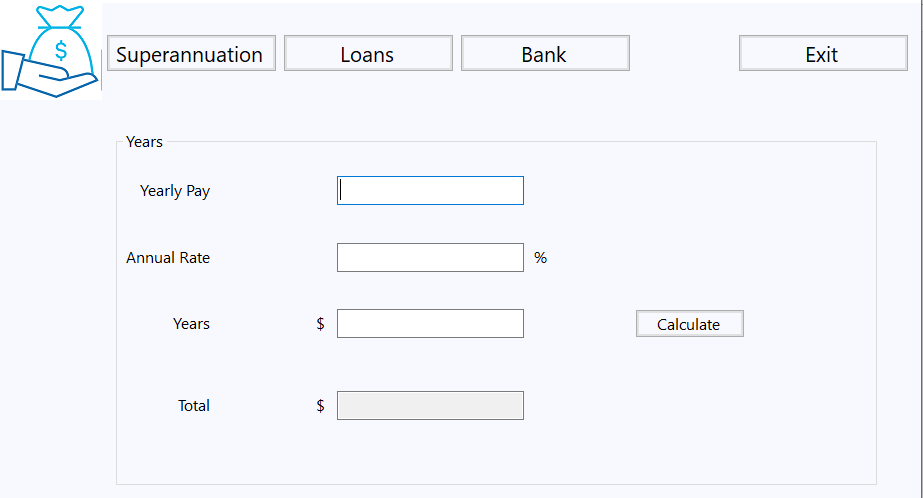
Three buttons on top are 3 different options for the client. If the logo is clicked on another page it will bring you to this page

If the username and password are correct it will go onto the next page otherwise a message box will pop up

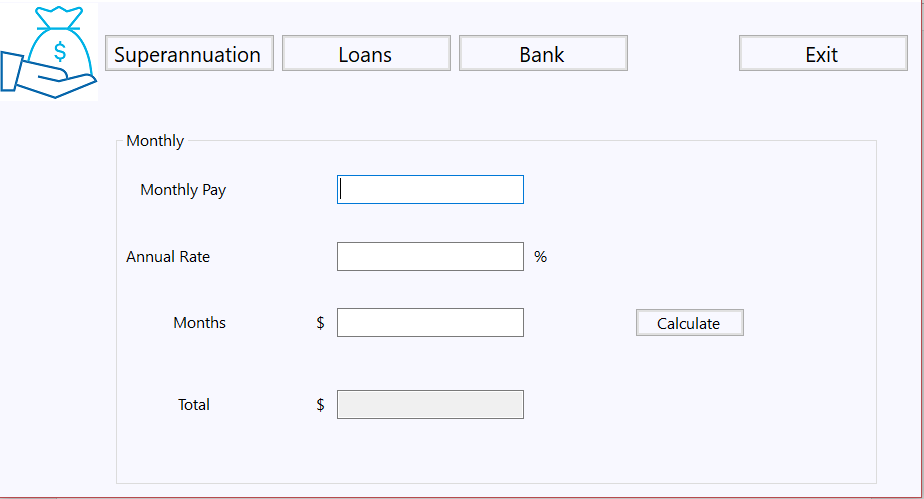


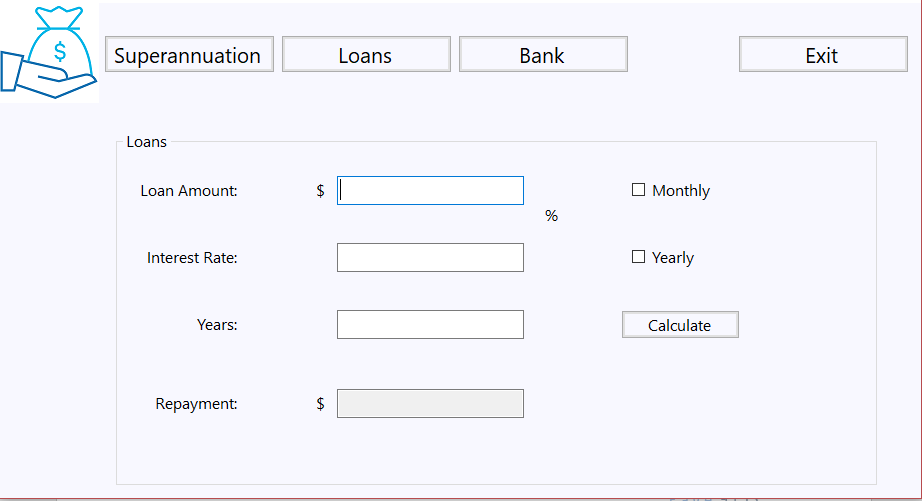


The user is given a choice between calculating annually or monthly.

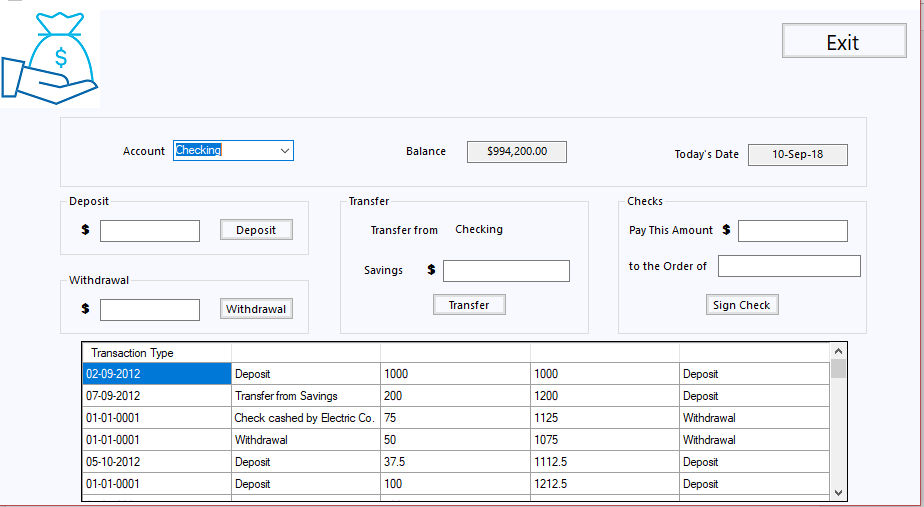


Textboxes only allow numerical input and the coding has a formula which calculates the superannuation total





Only difference between the super and loan form is that the monthly and yearly is given as checkboxes. You can only use one at a time



Extensive use of textboxes, drop down menus, buttons and grid views

**Gantt Chart**

**June Gantt Chart**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **June Gantt Chart** | **Estimated Time** |  | **Actual Time Taken** |  |
| **Dates** | **26th** | **28th** | **29th** | **30th** |
| Log In |  |  |  |  |
| Banking Transactions |  |  |  |  |
| Superannuation |  |  |  |  |
| Loans |  |  |  |  |
| Feasibility |  |  |  |  |
| Report |  |  |  |  |
| Data Flow Diagrams |  |  |  |  |
| Structure Chart |  |  |  |  |
| System Flow Charts |  |  |  |  |
| Pseudocode |  |  |  |  |
| Objectives |  |  |  |  |
|  |  |  |  |  |
| IPO Chart |  |  |  |  |
| Storyboard |  |  |  |  |
| Data Dictionary |  |  |  |  |
| Source Code |  |  |  |  |
| Hardware Testing |  |  |  |  |
| Software Testing |  |  |  |  |
| Data Desk Checks |  |  |  |  |
| Personnel and Procedural |  |  |  |  |

**July Gantt Chart**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **July Gantt Chart** | Estimated Time |  |  | Actual Time Taken |  |  |  |  |  |  |
| **Dates** | **3rd** | **5th** | **10th** | **12th** | **17th** | **19th** | **24th** | **26th** | **28th** | **31st** |
| Log In |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| Banking |  |  |  |  |  |  |  |  |  |  |
| Transactions |  |  |  |  |  |  |  |  |  |  |
| Superannuation |  |  |  |  |  |  |  |  |  |  |
| Loans |  |  |  |  |  |  |  |  |  |  |
| Feasibility |  |  |  |  |  |  |  |  |  |  |
| Report |  |  |  |  |  |  |  |  |  |  |
| Data Flow |  |  |  |  |  |  |  |  |  |  |
| Diagrams |  |  |  |  |  |  |  |  |  |  |
| Structure Chart |  |  |  |  |  |  |  |  |  |  |
| System Flow Charts |  |  |  |  |  |  |  |  |  |  |
| Pseudocode |  |  |  |  |  |  |  |  |  |  |
| Objectives |  |  |  |  |  |  |  |  |  |  |
| IPO Chart |  |  |  |  |  |  |  |  |  |  |
| Storyboard |  |  |  |  |  |  |  |  |  |  |
| Data Dictionary |  |  |  |  |  |  |  |  |  |  |
| Source Code |  |  |  |  |  |  |  |  |  |  |
| Hardware Testing |  |  |  |  |  |  |  |  |  |  |
| Software Testing |  |  |  |  |  |  |  |  |  |  |
| Data Desk Checks |  |  |  |  |  |  |  |  |  |  |
| Personnel and Procedural |  |  |  |  |  |  |  |  |  |  |

**August Gantt Chart**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **August Gantt Chart** | Estimated Time |  |  | Actual Time Taken |  |  |  |  |  |  |
| **Dates** | **1st** | **3rd** | **7th** | **9th** | **14th** | **17th** | **21st** | **23rd** | **28th** | **30th** |
| Log In |  |  |  |  |  |  |  |  |  |  |
| Banking |  |  |  |  |  |  |  |  |  |  |
| Transactions |  |  |  |  |  |  |  |  |  |  |
| Superannuation |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| Loans |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| Feasibility Report |  |  |  |  |  |  |  |  |  |  |
| Data Flow |  |  |  |  |  |  |  |  |  |  |
| Diagrams |  |  |  |  |  |  |  |  |  |  |
| Structure Chart |  |  |  |  |  |  |  |  |  |  |
| System Flow Charts |  |  |  |  |  |  |  |  |  |  |
| Pseudocode |  |  |  |  |  |  |  |  |  |  |
| Objectives |  |  |  |  |  |  |  |  |  |  |
| IPO Chart |  |  |  |  |  |  |  |  |  |  |
| Storyboard |  |  |  |  |  |  |  |  |  |  |
| Data Dictionary |  |  |  |  |  |  |  |  |  |  |
| Source Code |  |  |  |  |  |  |  |  |  |  |
| Hardware Testing |  |  |  |  |  |  |  |  |  |  |
| Software Testing |  |  |  |  |  |  |  |  |  |  |
| Data Desk Checks |  |  |  |  |  |  |  |  |  |  |
| Personnel and Procedural |  |  |  |  |  |  |  |  |  |  |

**September Gantt Chart**

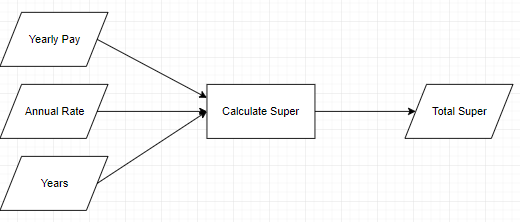
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **September Gantt Chart** | Estimated Time |  |  | Actual Time Taken |  |  |  |  |  |  |
| **Dates** | **1st** | **2nd** | **3rd** | **4th** | **5th** | **6th** | **7th** | **8th** | **9th** | **10th** |
| Log In |  |  |  |  |  |  |  |  |  |  |
| Banking |  |  |  |  |  |  |  |  |  |  |
| Transactions |  |  |  |  |  |  |  |  |  |  |
| Superannuation |  |  |  |  |  |  |  |  |  |  |
| Loans |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| Feasibility Report |  |  |  |  |  |  |  |  |  |  |
| Data Flow |  |  |  |  |  |  |  |  |  |  |
| Diagrams |  |  |  |  |  |  |  |  |  |  |
| Structure Chart |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| System Flow |  |  |  |  |  |  |  |  |  |  |
| Charts |  |  |  |  |  |  |  |  |  |  |
| Pseudocode |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| Objectives |  |  |  |  |  |  |  |  |  |  |
| IPO Chart |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| Storyboard |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| Data Dictionary |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| Source Code |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| Hardware |  |  |  |  |  |  |  |  |  |  |
| Testing |  |  |  |  |  |  |  |  |  |  |
| Software |  |  |  |  |  |  |  |  |  |  |
| Testing |  |  |  |  |  |  |  |  |  |  |
| Data Desk |  |  |  |  |  |  |  |  |  |  |
| Checks |  |  |  |  |  |  |  |  |  |  |
| Personnel and |  |  |  |  |  |  |  |  |  |  |
| Procedural |  |  |  |  |  |  |  |  |  |  |

**Log Book**

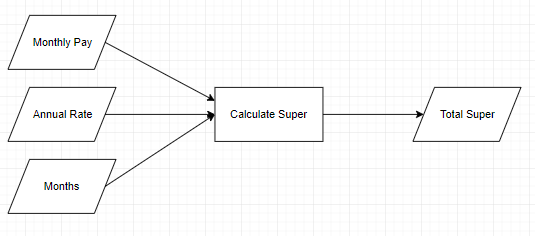
|  |  |
| --- | --- |
| Date | Explanation |
| 26th June | The project is handed out. What is specified:   * Time given for project * Search for client |
| 29th June | First meeting with client. Discussed items:   * What the idea for the program will be * How many functions within the program * Feasibility report is completed, and the project begins * Objectives and needs are set out which are * Banking system with log in |
| 1st July | * Log in system commences * Banking system commences * Data flow diagram commences |
| 17th July | * Log in system is finished * A second meeting is held and the log in is shown. * Approval of the log in system results in further needs for the client which are: * Superannuation * Loans |
| 3rd August | * The superannuation system commences * The banking system if difficult to work with and therefore there is a need to overcome the problem * Email is sent to client asking whether there are any more client requirements * The superannuation system is shown to client and client satisfaction is gained however the UI needs to improve |
| 21st August | * The loan system commences * The problem with superannuation system is solved * A meeting with client states that there is no need for a tutorial to the program |
| 23rd August | * The loan system commences * The system is quite easy however other commitments force the task to go longer than required |
| 3rd of September | * The loan system is finished * The Banking system is continuing to be a problem due to its difficulty * The algorithms for the * The loan system is shown to the client and approval is gained for the GUI and efficiency of the program |
| 4th September | * Structure chart commences * Pseudocode commences * System flow charts commence |
| 5th September | * IPO Chart commences * Source Code commences * Banking System is finished after a period of difficulty * IPO chart finished * Source Code finishes |
| 6th September | * Storyboarding commences * Storyboarding finishes * Hardware testing commences * Software testing commences * Data desk check commences * Personnel and procedural commences |
| 7th September | * Data Dictionary Commences * Client travels overseas for personal reasons |
| 8th September | * Data dictionary finishes * Hardware testing finishes * Software testing finishes * Data desk check finishes * Personnel and procedural finishes |
| 9th of September | * Review of the documentation and software commences and finishes |

**System Flow Chart**

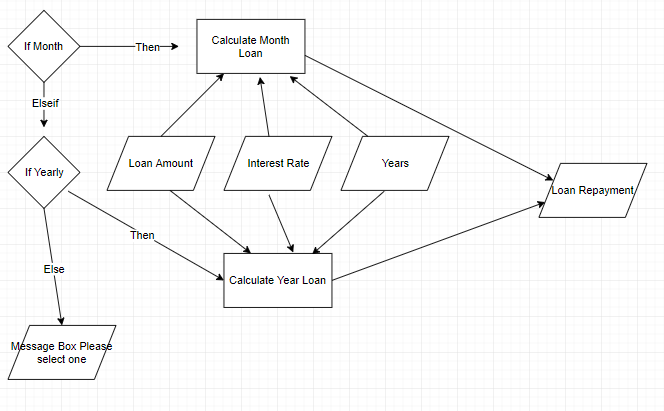
**Superannuation Yearly**



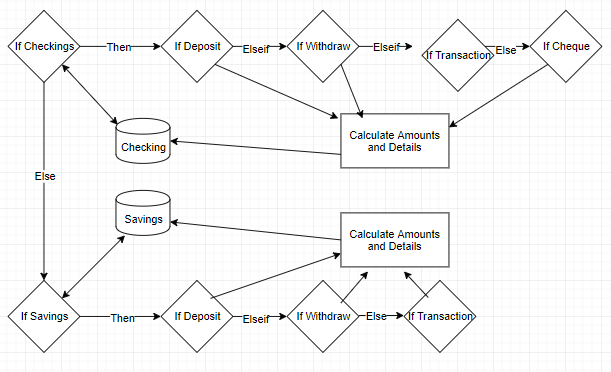
**Superannuation Monthly**



**Loans**



**Bank**



**Pseudocode**

BEGIN Superannuation  
 Year Pay = Integer  
 Rate = Floating Point  
 Years = Integer  
 WHILE Calculate = True  
 TextBox1 = ( Year Pay x 0.1) x ((1+Ratex0.1) ^ Years) - 1) / (Rate\*0.1)  
 ENDWHILE

END Superannuation

BEGIN SuperannuationM

Monthly Pay = Integer

Rate = Floating Point

Months = Integer

WHILE Calculate = True

A = Monthly Pay x 0.1

B = ( 1 + ( Rate / 12 ) )

C = ( B ^ Months )

ENDWHILE

END SuperannuationM

BEGIN Loan

LoanAmount = Integer

Interest Rate = Floating Point

Years = Integer

WHILE Calculate = True

A = LoanAmount

B = Interest Rate x 0.01

N = Years

If Month = True THEN

R = B/12

Y = N x 12

C = (1 + R) ^ Y

Textbox3 = A / ((C – 1) / (R x C))

ELSEIF Yearly = True THEN

Textbox4 = A / ((C – 1) / (B x C))

ELSE

PRINT Pick Month or Year

ENDIF

ENDWHILE

END Loan

BEGIN Bank

CBalance = CBalance

IF Checkings = True THEN

IF Deposits = True THEN

DepositInput + CBalance

Store CBalance

ELSEIF WithDraw = True THEN

CBalance – WithDrawInput

Store CBalance

ELSEIF Transaction = True THEN

CBalance – TransactionInput

Store CBalance

SBalance + TransactionInput

ElSEIF Cheque = True THEN

CBalance – ChequeInput

Store CBalance

ENDIF

ELSE Savings = True THEN

SBalance = SBalance  
 IF Deposits = True THEN

DepositInput + SBalance

Store SBalance

ELSEIF WithDraw = True THEN

SBalance – WithDrawInput

Store SBalance

ELSE Transaction = True THEN

SBalance – TransactionInput

Store SBalance

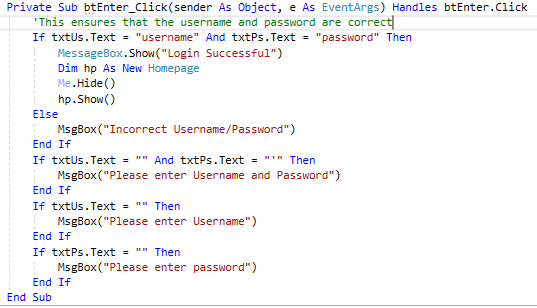
CBalance + TransactionInput

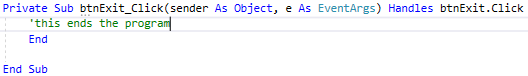
ENDIF

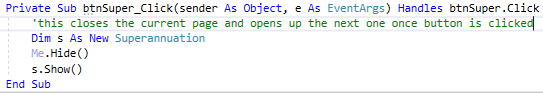
ENDIF

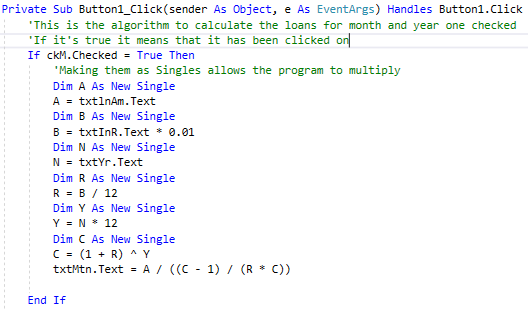
END Bank

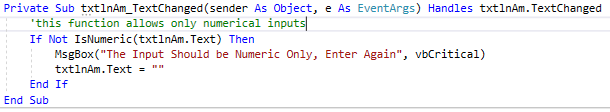
**Source Code**

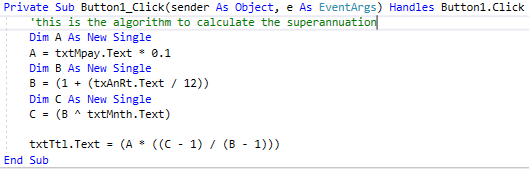


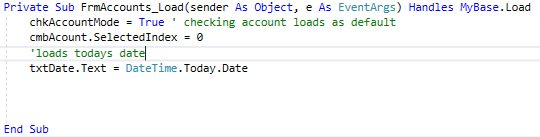


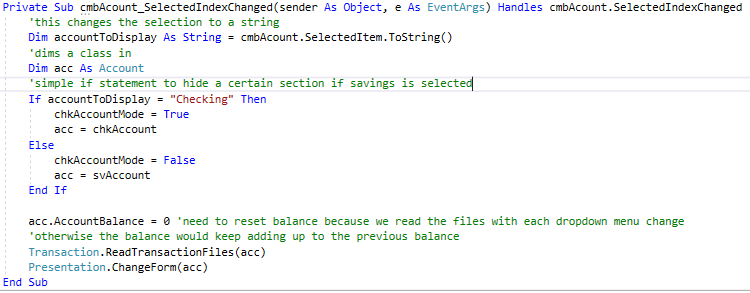


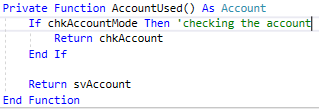


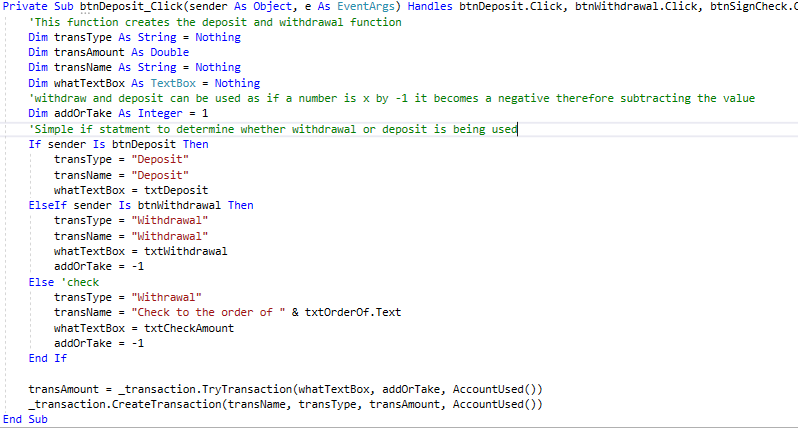


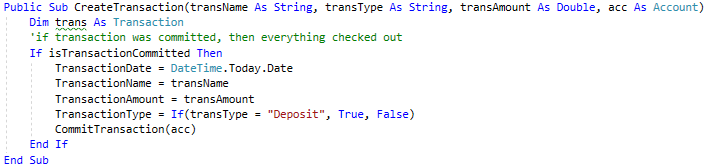
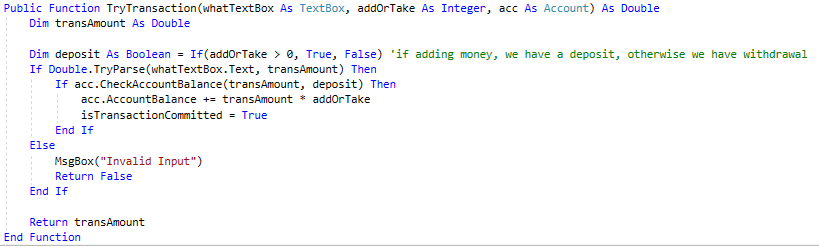
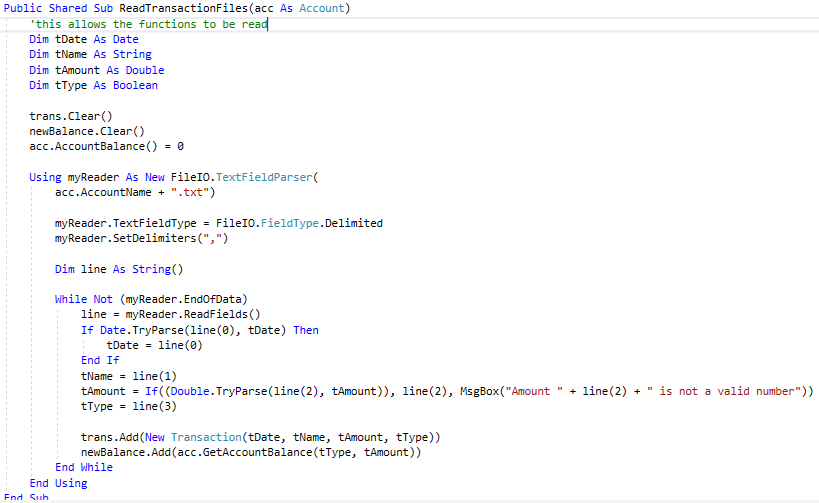
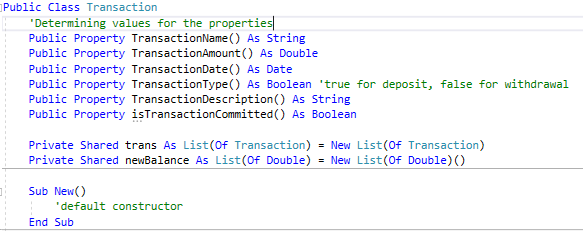
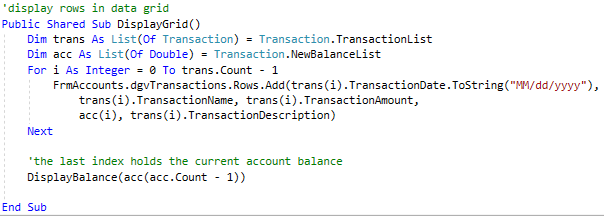
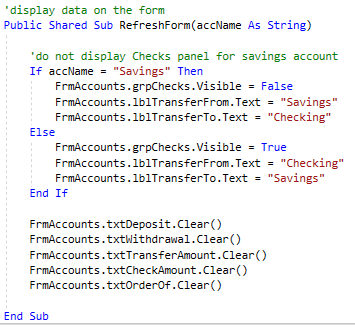
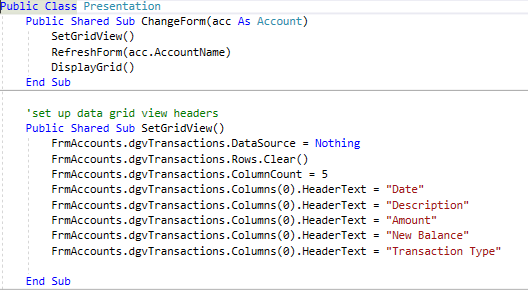
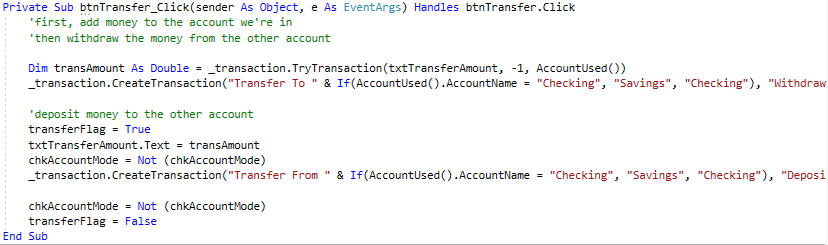
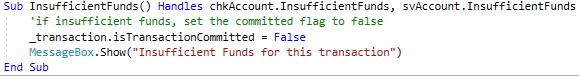
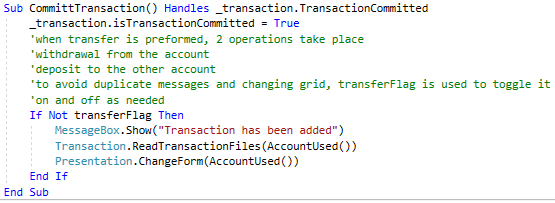












**Checking Software Solutions**

There are different testing processes for software solutions. They are:

**Alpha Testing:**

Alpha Testing refers to the testing of the final solution by personnel within the software development company prior to the products release

**Beta Testing**

Beta testing refers to the testing of the final solution by a limited number of users outside the software development company using real world data and conditions.

**Black Box Testing**

Also known as functional testing. The inputs and expected outputs are known; the processes occurring are unknown.

**White Box Testing**

Also known as structural, or open box testing. A software testing technique whereby explicit knowledge of the internal workings of the item being tested is used.

**System Level Testing**

System-level testing aims to ensure that the hardware, software, data, personnel and procedures that form the components of the final system are able to work together efficiently, correctly and in the manner intended with the new software product.

**Volume Data (Load Testing)**

Large amounts of data should be entered into the new software system to test the application under extreme load conditions. Muti-user products should be tested with large numbers of users entering and processing data simultaneously.

**Benchmarking**

Benchmarking is the process of evaluating a product in which from a point of reference from which quality or excellence is measured.

**Quality Assurance**

Quality assurance is about evaluating how well the software product meets or exceeds the users’ expectations based on correctness, reliability, efficiency, integrity, useability, maintainability, flexibility, testability, portability, re-usability and interoperability.

**Acceptance Testing**

Acceptance Testing is when formal tests are conducted to verify whether or not a system meets its requirements. Acceptance testing enables the client to determine whether or not to accept the new system.

**Testing**

**Hardware Test:**

The Hardware Test comprises of testing the product on different combinations of hardware ranging from those specified as minimum requirements to those that include any additional hardware devices. The different hardware that the Financial Program was tested on were:

* 2016 Lenovo Yoga
* 2017 Lenovo Yoga Pro
* 2009 Lenovo
* 2010 Toshiba
* 2011 HP

With the different hardware including different levels of functionality the software was able to run on all of them demonstrating its ability to run on different hardware’s.

**Software Test**

Software Test refers to how the program will run while other application is being used in the background of the hardware and how the program will affect other applications. The program was slowed down by a little amount when multiple tabs from Google Chrome and several applications were open. The program due to its small size does not have a major effect on any other application and does not slow down greatly therefore demonstrating its ability to run efficiently.

**Data desk checks**

Data Desk Checks allow the logic to be tested within a program

**Superannuation Yearly**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Line Number | Yearly Pay | Annual Rate | Years | Output |
| 1 | 50000 | 7.5 | 20 | 216523.50 |
| 2 | 70000 | 3.5 | 40 | 591851.4 |
| 3 | 10 | 1 | 1 | 1 |
| 4 | 180000 | 3.5 | 19 | 479051.2 |

**Superannuation Monthly**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Line Number | Monthly Pay | Annual Rate | Months | Output |
| 1 | 50000 | 12 | 90 | 6.1897E+30 |
| 2 | 15000 | 3.5 | 17 | 393663.4 |
| 3 | -9000 | 4 | 9 | Message Box (Please enter only numeric) |
| 4 | 0 | 0 | 0 | NaN |

**Loans**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Line Number | Loan Amount | Interest Rate | Years | Yearly | Monthly | Output |
| 1 | 1000000 | 3.5 | 90 | True | False | 36657.81 |
| 2 | 1000000 | 3.5 | 90 | False | True | 3047.875 |
| 3 | 90 | 15 | 1 | False | True | 8.12 |
| 4 | 90 | 15 | 1 | True | True | Message Box (Please select either month or year) |

**Bank**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Line Number | Checking | Savings | Balance | Deposit | Withdraw | Transaction | Cheque | Output |
| 1 | True | False | $994,100.00 | 100 | 0 | 0 | 0 | $994,200.00  Data to Grid |
| 2 | True | False | $994,200.00 | 0 | 0 | 1000 | 0 | C = $993,200.00  S = $3,590.00 |
| 3 | False | True | $3,590.00 | 0 | 4000 | 0 | False | Insufficient Funds |
| 4 | False | True | $3,590.00 | 900 | 0 | 0 | False | $4,490.00 |

**Personnel and Procedure**

**Alpha Testing**

Alpha Testing refers to the testing of the final solution by personnel within the software development company prior to the products release and was done so.

**Beta Testing**

Also known as functional testing. The inputs and expected outputs are known; the processes occurring are unknown.

|  |  |  |
| --- | --- | --- |
| Name | Positives | Improvements |
| Eric Kim | * Simple and clear instruction and format, to be easily accessed by anyone. * Accurate | * More Functions |
| Parasdeep Bindra | * Easy to navigate * Accurate | * Improvement on graphics |

**Peer Testing**

Peer Testing is when your program is evaluated by your peers in order to gain feedback.

|  |  |  |
| --- | --- | --- |
| Name | Positives | Improvements |
| Samuel Wibawa | * Meets the set objectives. * Good user interface as it is easy to use * Has nice contrast in colour | * Try Adding More Functions |
| Myles Pritchett | * Meets the objectives you set out * User interface is consistent * Easy to use | * Try Adding More Functions |

**Quality Assurance**

Quality Assurance was used throughout the software development cycle and as a result the program operates without any faults and therefore the client needs were met.

**Acceptance Testing**

Acceptance Testing is when formal tests are conducted to verify whether or not a system meets its requirements. Acceptance testing enables the client to determine whether or not to accept the new system.

|  |  |
| --- | --- |
| Client Requirements | Outcome |
| Create a bank program which can only be accessed by me and replicates a real-life bank | Completed |
| Within the bank program you should be able to make deposits and withdrawals | Completed |
| Savings and checking type of accounts | Completed |
| Superannuation calculator (both monthly and annually) | Completed |
| Loan calculator (both monthly and annually) | Completed |
| Program should be extremely easy to use | Completed |
| Complete the program within the specified time period | Completed |
| There should be no errors | Completed |
| There should be a good user interface | Completed |
| Program should be free of malware | Completed |
| Be able to view transactions | Completed |

The program has met all of the client needs and therefore the project has been a success and can be accepted by the client.

**Prathamesh Tarde**