# **Botany Downs Secondary College**

**Internal Assessment**

**Level 3**

**91906- Use Complex Techniques to Develop a Computer Program-6 credits**

**91907- Use complex processes to develop a digital technologies outcome** **-6 credits**

|  |  |  |
| --- | --- | --- |
| **Achieved** | **Merit** | **Excellence** |
| · Use complex programming techniques to develop a computer program. | · Use complex programming techniques to develop an informed computer program. | · Use complex programming techniques to develop a refined computer program. |

Due Date ; July 28th 2023

**Student Declaration:**

I hereby declare that I have completed the assessment for 91906-907, independently and to the best of my abilities. This assessment represents my own work and is based on my own research, practice and understanding of the subject matter.

I confirm that all sources used in this assessment, including but not limited to books, articles, online resources, and any other references, have been appropriately cited and acknowledged according to the prescribed referencing style.

I further affirm that I have not engaged in any form of academic dishonesty, such as plagiarism or unauthorized collaboration, in the completion of this assessment. The ideas, arguments and content presented in this assessment are my own and have not been copied or reproduced from any other source.

I understand that any act of academic misconduct or violation of the academic integrity may result in disciplinary actions, which could include penalties such as grade reduction, course failure or other consequences as determined by the institution.

I take full responsibility for the authenticity and originality of the assessment and acknowledge that my work will be subject to scrutiny and evaluation by my instructors or assessors.

Signed : …………Nikhil……………….

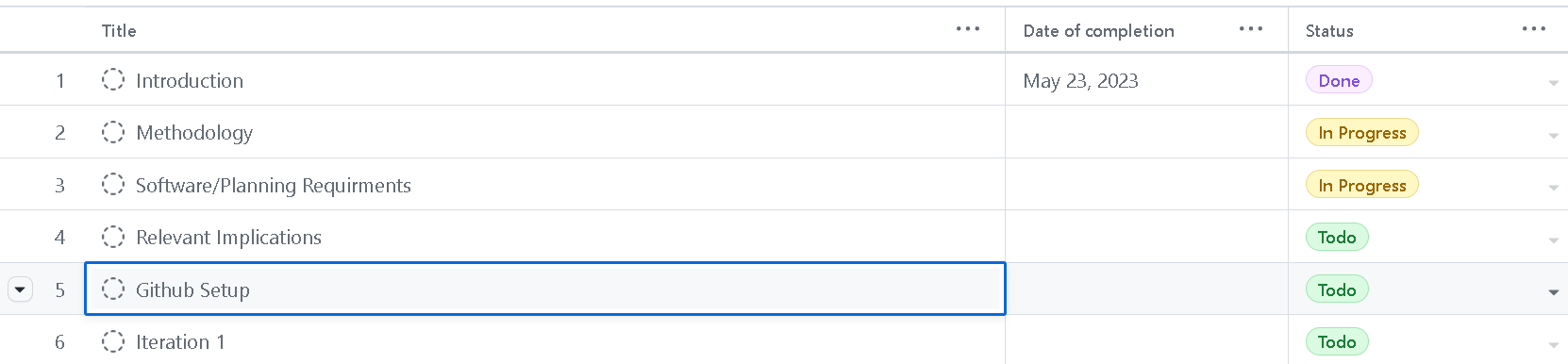
Date: ………………22/05/23…………………..

Signed:

Date:

**Introduction to the chosen task:**

The name of my app will be ‘**Smart Calculator**’ this app will have many different types of calculators. For example, there will be a **basic math calculator, a tax calculator and a date to day calculator.**

I can see this app being used by many people. People who work will need to calculate their taxes. And the tax calculator will help them do that. Also people might need to use a calculator for everyday use such as measurement and cost calculations. 

**Methodology:**

I will start with opening a tkinter window. Then I will use a menu function to set up the different parts of my app (calculator, tax and date).

Then for the calculator I will use a function to calculate the input. I am thinking about using the eval function so that the user input is verified by the code and then is calculated. I am also thinking about making the calculator so that it displays the user input, I am unsure on how to do that as of now.

For the tax calculator I will create an entry box and then use if statements to divide all the tax brackets. Then I will create a label which will print the salary after tax

For the date to days calculator, I will likely have to import the datetime module from the python library. Then I will have to use dropdowns so that the user can select date, month, and year. Then the datetime module will calculate the days between the two days.

**Software/Planning Requirements**

I will need to use a code editor; I will be using Visual Studio Code for this app because I have used it in the past and I am familiar with its features. I chose VScode because it gives me guidance on where I made errors. It shows me the exact line and reason for an error if I have one. I find that to be helpful when I code so that is why I will be using VScode.

I will also need to use GitHub to manage my project and to keep track of all the code changes that I make during the project. GitHub is a well-known and widely used platform for programmers. I can use GitHub also to go back and undo any changes that I might make. I will also use GitHub to track my progress, I will use the project feature on GitHub to set milestones and create a task decomposition board on GitHub.

I will need to find a way to draw my wireframes and flowcharts. For this I will use mock ups, this is a good website since it allows me to use drag-and-drop techniques to make my wire frame. Drag and drop is a quick and easy way for me to create my wireframes. I have also used mock ups for last year’s wireframes, so I am familiar with the features. I will Make my flowcharts using lucid charts. The lucid chart is like mock ups and it is also drag-and-drop type of website. But lucid chart is more code based and it has shapes which are used for flowcharts. That is why I’ll be using lucid charts for my flowcharts.

I will be using complex techniques and processes which will help me achieve a better result here is a list of some of the complex techniques and processes that I’ll consider be using:

* Menu bar
* Functions
* Tkinter
* Dropdown (Option Menu)
* Classes and other features of classes
* Object oriented programming

A picture containing text, line, screenshot, font

Description automatically generated

**Relevant Implications: (list the implications that are only relevant to your program development.)**

|  |  |  |
| --- | --- | --- |
| **Relevant Implication** | **Describe** | **Explain** |
| Usability | The end users are the ones that are most effected by the usability implication. And because of this this is one of the most important implications for me. The end users are very influential since they are the ones who decide whether the app is usable or not. So, because of this the usability implication is important. This implication will be in effect throughout the entire app. For an app to be usable the user must be able to navigate through the app with little to no help. They also must not face any bugs and glitches that may cause the app to stop working. The user also must be able to see any buttons or texts that the app shows. | I will ensure that this app is usable by looking at the usability heuristics. I can look at what I need to do to make my app usable. The heuristics I found useful were the consistency, user help, user freedom and a few more. I now know how I can make my app usable. I will have to probably add a help section to my app where I can have like a FAQ page, or I can add a link to a form which the user can fill to give me feedback on the app. I will also have to keep the whole app consistent throughout the page. To do this I will have to ensure that the whole app works and that there is a smooth transition between the different features. And to make the app more usable I will try to give the user as much freedom as possible. I will do this by setting up a menu bar as I have stated before. Having a menu bar will let the user choose where they want to go next. |
| Functionality | The functionality implication requires me to code the program in a way where everything works as it is intended and so that the purpose of the app is met. The purpose of this app is to have multiple apps in once place, so the user doesn’t have to keep switching. This implication will also require me to ensure that there are no flaws in the code which will make the app stop being functional. | I will reduce the chances of crashes and bugs by putting my code through a code checker. This will check the app for any problems. And I will refine my app using this. I will use **feedback** from my teacher and clients by having them look at my program. Since they will be the end users, they can tell me which features I need to add to make my program more **functional**. By asking others for feedback, I can also confirm whether I am doing the right thing and meeting the purpose of my outcome. I will also use **testing and trialling** to check if the app is working myself. |
| Aesthetics | This implication is about how good an app looks. Also, about how the colour scheme and theme work together to make the overall app deliver its purpose better. This implication will need to be consistent throughout the entire app since having only one page with colour will ruin the entire aesthetic of the app. The end users will be affected the most by this implication. | I will ensure that my app is aesthetically good by analysing colour schemes, images, content placement and readability. I will seek feedback and look at existing websites/apps for great ideas. By doing this I will get a good idea of the aesthetics that I need to apply to my app. |
| Copyrights | Copyrights is a legal implication that I will be discussing. This implication requires me to not use any copyrighted images, texts, or code without reference/ buying rights monetarily. If I do this without their consent, I can be fined. This implication impacts me the most since I will have to take responsibility for any problems that may arise with this implication. | I will ensure that I am not copying anyone’s work by using GitHub, this will allow the teacher to see how I have updated code throughout the weeks. The teacher will also be reviewing the code with me to see how much of the code I understand, and I can show that I haven’t copied code by showing my understanding of the code. And for images If I use them I will use websites such as pixabay, these websites have thousands of copyright free images, meaning I can use any image from there without worrying about copyright. |

**A picture containing text, font, line, screenshot

Description automatically generated**

**Links to Project Management:**

|  |  |
| --- | --- |
| **Project Management Tools** | **Link** |
| **Github – for project management and also to save and commit changes in repository.** | <https://github.com/NikhilKamath2006/AS91906-907.git> |
| **Draw.IO** | N/A I have created a folder named mock ups where I have put images of wireframes. |

**(Screenshot of Task Decomposition from Github)**

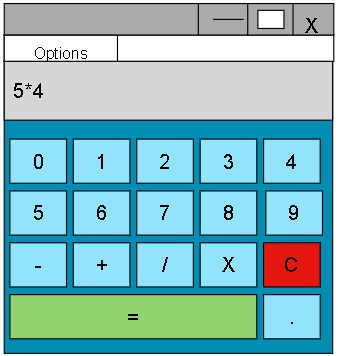
**A screenshot of a computer

Description automatically generated with low confidence**

**Iteration1: Explain what actions will happen in your first iteration.**

**(**This should include the wireframe of your GUI, code structure, classes used, functions used. Highlight the key sections of the code that are of significance.**)**

**Wireframe of GUI with Annotations:**

****Here is the wireframe for my first iteration. As we can see here it Is a basic calculator. It has all the buttons, and it even has a decimal point. I may change the colour scheme later. I need to research about aesthetic colours and then I will design this.

**Note: This table of objects is for all iterations**

**Table of objects (**Add more rows if required.**)**

|  |  |  |
| --- | --- | --- |
| **Objects/Variables/Storage structures such as lists/dictionaries/CSV/JSON** | **Datatype** | **Purpose and its relevance to the outcome development** |
| **main\_menu** | Menu | It creates a menu so user can switch between programs within the app |
| **Class Calculator** | Class | Class for the calculator feature |
| **Live\_display** | Function | This function shows a live display of input/result |
| **calculate** | Function | Calculates the input |
| **Button 1-9** | widget | Buttons for the calculator |
| **Button x/+-** | Widget/operator | They are the operation signs for the calculator |
| **clear** | Button | It clears the calculation that is currently in the display |
| **Tax** | Class | A class for the tax feature |
| **Tax\_label** | Label | A label asks the user salary |
| **Salary** | Entry | Entry for user to enter salary |
| **Taxed\_label** | Label | Shows tax to pay and amount left after tax |
| **Tax\_button** | Button | Calculates tax when clicked |
| **Date** | Class | Class for the date feature |
| **Day/month/year** | Option Menu | Dropdowns so user can choose the dates |
| **Question** | Label | Question asking user to choose dates |
| **Days** | Label | Shows days between dates |
| **Find** | Button | Initiates the calculation of days |
| **Welcome/title/info** | Labels | Labels for when user opens program |
| **Calc\_feature,tax\_feature,**  **day\_feature** | Buttons | Buttons leading user to different parts of app |
| **All frames** | Frames | Frames so the different features don’t overlap with one another |

**Screenshot of Version 1 Flowchart:**

**A diagram of a company

Description automatically generated**

**Iteration 1: Component Testing Table: Include Screenshots and give reason which component is selected and why?**

**Note:**

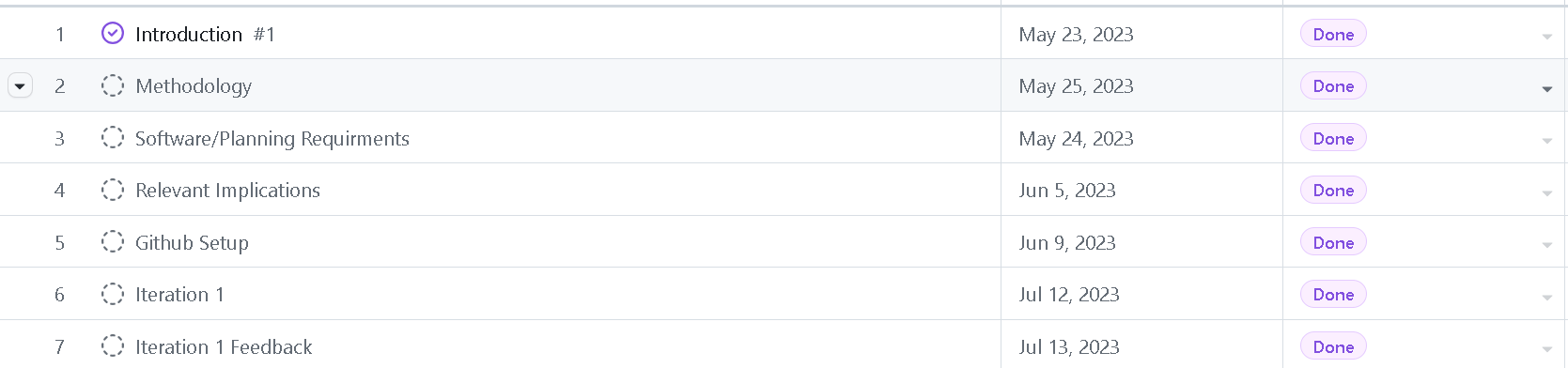
**There are no boundary cases in this iteration so I have only tested invalid and valid cases.**

|  |  |
| --- | --- |
| **Test Case** | **Expected** |
| For the first test I tested a simple equation using only addition. I tested this because it is an easy equation which will allow me to know if the code is working or not. **I entered 17+6. (Valid case)** | **I expected an answer of 23** and this is what the result was:    This is the code that runs this test:    And for the buttons this is the code (I have put code for one button but all the other buttons are the same) |
| For this test I chose to test if this program can do BEDMAS. This is important because if it can not to BEDMAS the end answer will be wrong. **I entered 31–5+15. (Valid case)** | I solved the question given using BEDMAS, I also used a calculator to confirm the answer. And the **expected answer should be 41** here is the result:    And this is the function that makes this work(I have not included an image of the buttons since I have done that in previous cases, the buttons are the same): |
| In this test I chose to use negative numbers, negative numbers can cause problems in the code since the code might confuse the negative number for a misplaced minus sign. **I tested -7\*-2 (Valid Case)** | This is the result:  The **expected answer was 14.**    The code that runs this is the same as the code from the previous test case. |
| Now I will try an invalid statement, By doing this I will be able to see whether the program breaks or it shows the error message that I have coded. **The invalid statement I entered was 0/0. (Invalid)** | Here is the result. This did not go as expected. I expected to get error straight away but instead I got a confirmation then I got the error:  It didn’t matter what I clicked I got this after I clicked one of the buttons: |
| This is the final case I tested for this iteration. I tested a decimal calculation. I tested this since decimal points aren’t whole numbers and I wanted to know whether the code was understanding the decimal input. **I tested 7.81\*2.42 (Valid Case)** | And this was the result**. I expected a result of 18.9** and this is what I got:    After testing this yes this works with decimal points as well. |

**Feedback on Version 1: include a testing video here.**

|  |  |
| --- | --- |
| **Stakeholder** | **Feedback** |
| **SH1(Dale Evans Parker)** | This is a good program. I like the fact that you have added and confirm clear button. By doing this you will help users a lot. I also like that you have tried making a confirm exit page. I hope that it works in the next iteration. The calculator works as it should. By looking at other calculators I can see that your calculator does not have exponents or square roots. This can maybe be an idea for future iterations. |
| **SH2(Enrique Macwan)** | Nikhil, I think you have done a good job for your initial development of your project. The most important part, the calculator works as it is supposed to. I did find a few problems when looking at your program though. One of them was the fact that the exit button wasn’t working. Another one was the fact that I was able to type into the box where the answer was supposed to go. It would be good if you could fix those issues. I think you should try making the tax calculator for your next iteration. |
| **Teacher’s Feedback** | Calculator is done well. However, try to improve the aesthetics. **Now try using the class and create the calculator program.** The code is working well. Comments are describing the functionality of the code. |

**Updated Trello Screenshot after Version1**

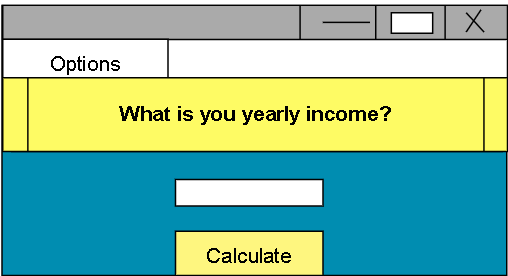
****

**Summary of Feedback and intended changes to make in Version 2:**

After listening to the feedback given to me by my stakeholders I have come up with some additions, changes, and fixes to my program. I am planning to fix the exit button confirmation; I am also planning to fix the invalid cases on the calculator. I also thought about what else I can implement, and I thought about adding a tax calculator. I also want to change the properties of the display without breaking the code. As of now the user can type into the display box and this is not a good thing since it can stop the code from working. Therefore, I will try to lock the display box.

**Iteration2: Explain what actions will happen in your second iteration.**

**(**This should include the GUI Wireframe,code structure, classes used, functions used. Highlight the key sections of the code that are of significance. Include the Task screenshot from Trello**)**

****As we can see here this is the wireframe for the second iteration of my project. I have added a tax calculator to my program. It matches the aesthetic of the previous iteration. There is one label, entry box and one button. The user will enter their salary into the entry box and the result will show up on a label.

**Screenshot of V2 Flowchart:**

**A diagram of a tax process

Description automatically generated**

Flowchart inside of red box is for iteration 2, the rest of it is same as iteration 1 flowchart. I have cut off that part since the flowchart would be too big and I will not be able to clearly show the flowchart for iteration 2.

**Iteration 2: Component Testing Table: Include Screenshots: Include Screenshots and give reason which component is selected and why?**

|  |  |
| --- | --- |
| **Test Case** | **Expected** |
| For the first test I will look at the lowest tax bracket to see whether the tax I get is correct, (I entered 12000, Valid test case): | I checked whether if the tax calculation is correct on the official IRD website. The expected answer was $1260 tax to pay:  And this is the code that makes it work: |
| For the next test I tried using a negative salary, this is not taxable. I entered -21234 (Invalid Test): | This was the result:    (Nothing happened)  I realised that I did not have any precautions for invalid statements so I went back and added a message box which would let the user know about the invalid input.  Here is the result after the fix:    So now when the user enters an invalid number they will be guided.  And here is the code that I made to fix this problem:    I added an else statement after the elif statements from the tax brackets. By doing this, the code runs all the if and elifs first and if it doesn’t match that category, it will display this message box. |
| Now I will test my menu. I want to know if I can switch between the calculator and tax. I first opened the calculator then clicked tax (Valid Test): | This is what happened next:  It looks like the calculator opened on top of the tax feature. I will need to go back and find a way to close the tax feature then open the calculator.    After fixing it the two features did not open on top of one another, I added frames so that it wouldn’t happen. Here is the code I used to fix the problem:      And before I ran the function, I cleared all frames and then packed the one which was needed: |
| Now I will try entering floats. Users may have a salary which ends with decimal points. So, I need to check for that **I entered a salary of 14000.65**(Valid Case): | I got an error message in the terminal saying **‘Value Error: invalid literal for int() with base 10 ‘14000.65’.’**  I will need to go back most likely and convert the user entry into a float before I goes through the calculate function.  This was the result after fixing and the code:    I changed the conversion from integer to float |
| Finally, I will try to enter letters and symbols, this test will check to see whether the program works with these characters **I entered $five thousand five hundred** (Invalid Case): | Like the decimal test I got another value error this time the terminal said **‘ValueError: could not convert string to float: '$five thousand five hundred’’.**  Here is the fixed result and code:  And after clicking ok:    I used a try and except keywords to solve this issue. I tried to see whether the input is a float or not. And if it wasn’t I wrote an exception value error where a message box would pop up, here is the code: |
| Finally, I will test the boundary value for the tax calculator**. I will enter 14000 as the salary**, this is the boundary between 10.5% and 17.5% tax. (Boundary Test) | Here is the result for this case and this was the expected result:  And here is the code that is responsible for this result:  The <= is the reason the boundary values are correct. |

**Feedback on Version 2: include a testing video here.**

|  |  |
| --- | --- |
| **Stakeholder** | **Feedback** |
| **SH1(Dale-Evans Parker)** | I like that you have added a tax feature to your app. This gives your app more uses. After using this app, I found it easy to use. I would get error messages if I did something wrong. I like that there are not too many things that the user can do. You only let them have one entry box and one button to click. By doing this you are making this app more user friendly. |
| **SH2-Enrique Macwan** | This iteration adds a lot to your overall app. I find this useful since it can help people who struggle with maths. But most importantly I think you should optimise your code more. I can see that right now you are only using functions to run your app. I think you should start using classes. By using classes, you will be able to sort out the different parts of your app. |
| **Teacher’s Feedback** |  |

**Update the Trello and place a screenshot showing completion of above task.**

**A screenshot of a computer

Description automatically generated**

**Summary of Feedback and intended changes to make in Version 3:**

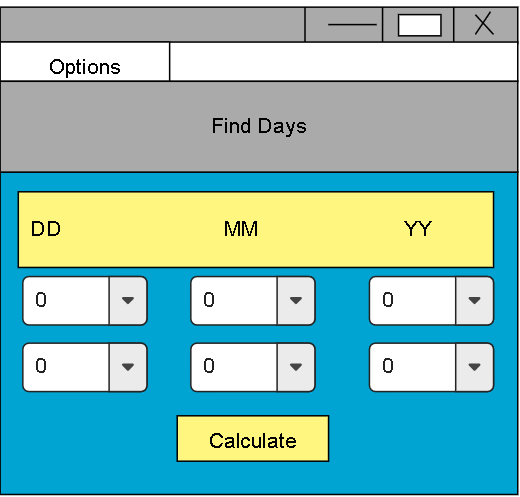
After listening to my stakeholders, I thought that I should start using classes starting from this iteration. I will have to go back and change the code of the previous two iterations. I am still yet to investigate the aesthetics of the app since I first want to finish all the functioning parts of the app. I will likely fix the aesthetics in the final iteration.

**Iteration3: Explain what actions will happen in your 3rd iteration.**

**(**This should include the GUI Wireframe, code structure, classes used, functions used. Highlight the key sections of the code that are of significance.**)**

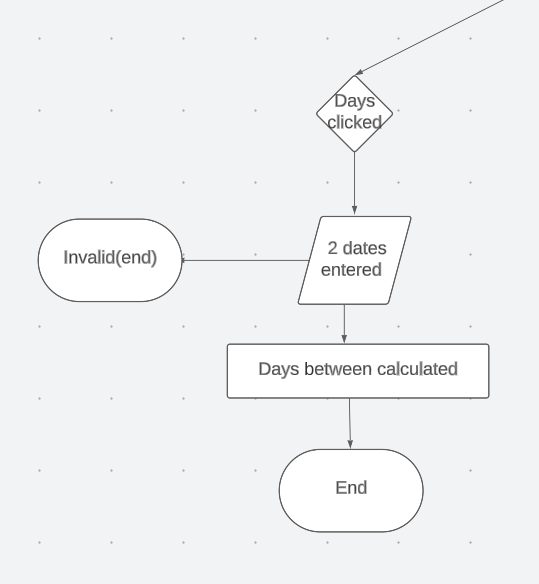
I am planning to make one class for the calculator, another class for the tax calculator and another class for the day calculator. In this iteration I will be adding a day feature to my app. Essentially it will find the number of days between two given dates. I will use option menus so the user can choose the year, day, and month. Then I will use the datetime module to find the number of days in between.

Here is the wireframe for the third iteration:



As we can see on the wireframe there are going to be 6 dropdowns. The user can enter 2 dates using the 6 dropdowns in day/month/year format. Then the click on the calculate button. And then the result will be shown on a label.

**Screenshot of V3 Flowchart:**

**Note:** I have only shown the part, which is relevant to iteration 3, but the rest of the flowchart is the same as the iteration 2 flowchart. This part branches off from the start of the program. I have the full flowchart in the flowchart folder.

**Iteration 3: Component Testing Table: Include Screenshots : Include Screenshots and give reason which component is selected and why?**

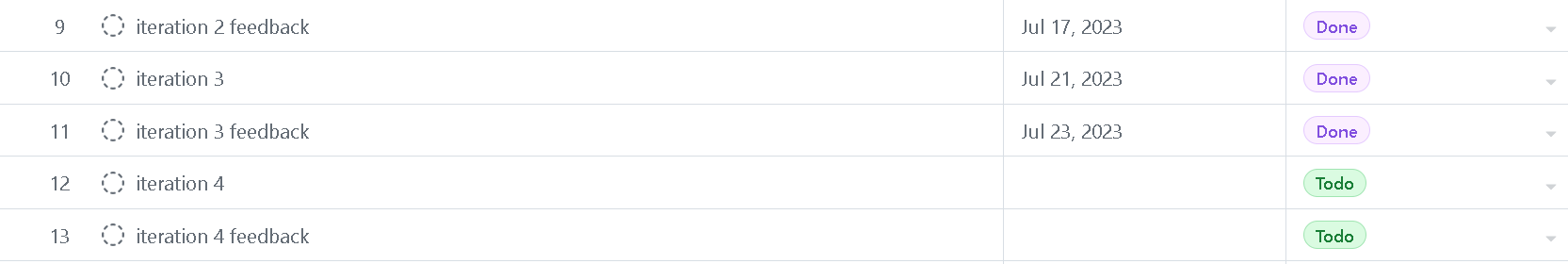
**Once again there are no boundary cases for this iteration so I will only test invalid and valid cases.**

|  |  |
| --- | --- |
| **Test Case** | **Expected** |
| First, I tried two normal dates one year apart**.** This test was done to see if the code can run at least the most basic part. **I entered 6/6/6 and 6/6/7. (Valid case)** | I expected a result of 365 days here is the result I got:    And here is the code which makes it work (I have only shown the code for the function not the widgets): |
| Next, I will test an invalid case. **I entered the first date as 1/2/3. And I entered the second date as 1/2/2**. I want to see whether I get an error or a negative result. | I got a negative result:    I thought about this test for a bit and then I thought that the calculator shouldn’t consider in which order the dates are entered. So, I tweaked the code, so I always get a positive result:    Before I didn’t have the abs function, by adding it I can ensure I always get the answer in its absolute value, meaning always positive. |
| Now I will test to see if the program can tell if the date is valid or not. **I will enter 1/2/3 and 30/2/3.**  Since 30 February doesn’t exist nothing should happen, or I should get an error. Invalid | When I clicked the calculate button I got:  **‘ValueError: day is out of range for month’.**  So, I had to find a method which I could use which would check for invalid dates. So, like previous iterations I used the try and except method. Here is the fixed result and code:    As we can see here, I used the try and except. I tested this with leap years as well and it worked, so I do not need to test for that again. |

**Feedback on Version 3: include a testing video here.**

|  |  |
| --- | --- |
| **Stakeholder** | **Feedback** |
| **SH1-Dale Evans Parker** | So far this is my favourite part of the app since it is quite innovative and because I do not see this kind of calculator much. I struggled to use it a bit though. This was because there were no labels telling the user what to enter. I think you should add labels such as dd/mm/yy so the user will know what to enter the option menus. Other than that, the app worked with no bugs for me. For your next iteration I think you should focus on aesthetics |
| **SH2-Enrique Macwan** | I liked this iteration; it was simple to use, and I couldn’t find any bugs. Like dale mentioned I think you need to add labels to guide the user. I only knew what to enter because you were next to me. Also, I see that you have started to use classes. That is good but I see that your comments are lacking. I think you can add more detail to your code comments. |
| **Teacher’s Feedback** |  |

**GitHub Project Management:**

****

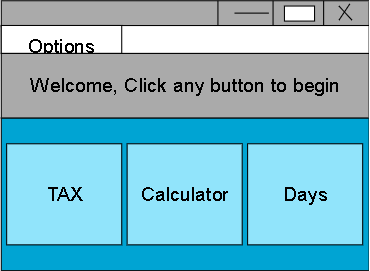
**Summary of Feedback and intended changes to make in Version 4:**

After looking at the feedback I noticed that both my stakeholders wanted me to upgrade the aesthetics of the app. Even my teacher said this back in iteration 1 feedback. So, I think I will be upgrading the aesthetics of my app for this iteration. I am also planning to add a welcome page for the user where they can access all the different parts of the app. I will also make quality improvements. When testing iteration 1 I had found that doing 0/0 would confuse the program and a clear message will pop up. I want to fix this and other things similar to it.

**Iteration4: Explain what actions will happen in your fourth iteration.**

**(**This should include the code structure, classes used, functions used. Highlight the key sections of the code that are of significance.**)**

For my fourth iteration will add a welcome page and I will make aesthetic improvements. I will also be fixing any bugs or inconveniences in the program. Here is the G.U.I wireframe for the welcome page of my app:

As we can see here there will be three buttons on the welcome page. And there will be a title welcoming the user. And if the user clicks on a button they will be taken to that feature. Then they can click on the option menu to return or go to a different feature.

**Screenshot of V4 Flowchart:**

**A screenshot of a diagram

Description automatically generated**The only new addition to the flowchart is the welcome page clicked part. The rest is the same as previous iterations. The full flowchart is in the folder.

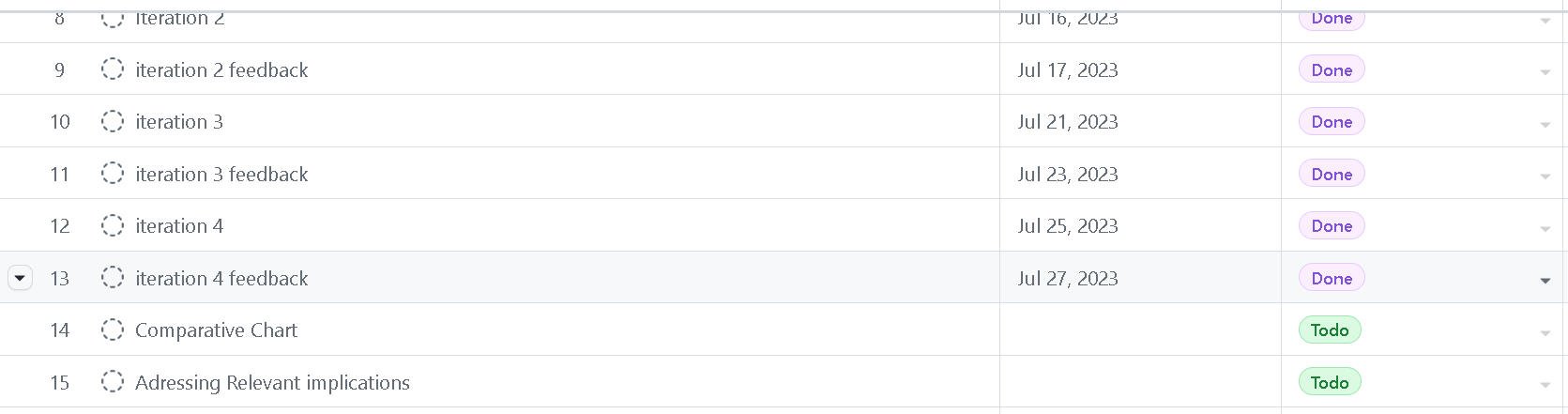
**Iteration 4: Component Testing Table: Include Screenshots : Include Screenshots and give reason which component is selected and why?**

**There were no boundary cases for this iteration.**

|  |  |
| --- | --- |
| **Test Case** | **Expected** |
| **I tested 0/0** on the calculator, back in iteration 1 the code got confused and ran the wrong thing. 0/0 is invalid so maybe my try and except is wrong. | This is how I fixed the problem:    I added two exceptions to the try block. I added the divide by zero exception. And because of it I will get an error message when dividing by zero. This is what I get now: |
| Next, I tested the new welcome page, I tested all the buttons to see if they take me to the right place. This was a valid test. | I have only shown me testing one of them but I actually tested all the buttons:    And after I clicked tax I got this: |
|  | The code that runs this is: |

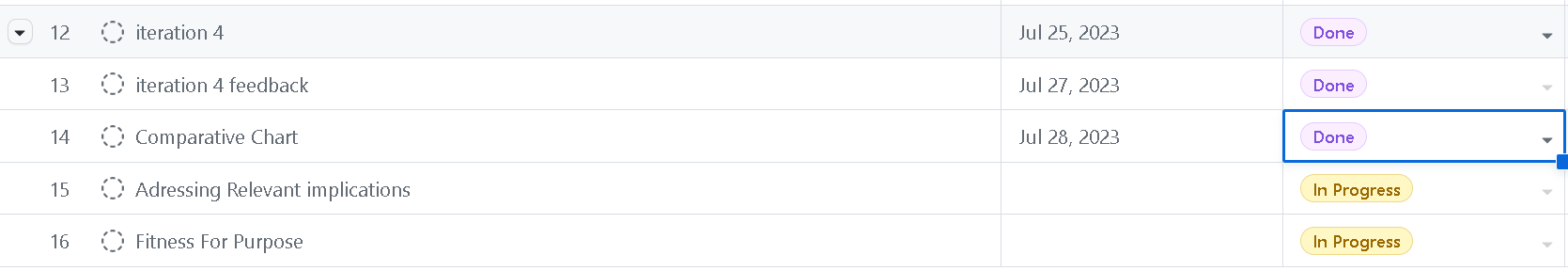
**Feedback on Version4: include a testing video here.**

|  |  |
| --- | --- |
| **Stakeholder** | **Feedback** |
| **SH1** | Nikhil, think you have made your program more appealing to the eye. I also see that you fixed bugs that have stayed since previous iterations. I think your end outcome Is good, it serves its purpose which is important. I like how you used ttk.optionmenu instead of the normal ones. I think using that made your day calculator good looking |
| **SH2** | I also think that your outcome is good. I like that you listened to our feedback and actually improved the aesthetics of the app. I also liked that you used classes to organise your code. You also added comments since the last time I asked you which is good. |
| **Teacher’s Feedback** |  |

****

**A comparative chart of Version GUI showing iterative improvement**

|  |  |
| --- | --- |
| **Version1 Description** | **GUI Design** |
| For version 1 I just had a calculator that would do basic math equations. And the buttons that were clicked would be shown on a live display. It was just a simple and basic design with not many designs. |  |
| **Version2 Description and Feedback** | **Gui Design** |
| For this version I added the tax calculator. My teacher told me to look at different aesthetics. So when coding this iteration I was looking at aesthetics for future iterations. So the only design change was the addition of the tax calculator |  |
| **Version 3 Description and Feedback** | **GUI Design** |
| For iteration 3 I added the date to days calculator. My stakeholders also told me that I should try change the colour scheme of the program. I told them I was researching it. I didn’t make any visual upgrades other than adding the date to days calculator for this iteration. |  |
| **Version4 Description and Feedback** | **GUI Design** |
| This was my final iteration and for this iteration I chose to finally upgrade the aesthetics of this app. I changed the colour scheme, added padding, and made the app look better overall. I also fixed any outstanding bugs that might have made it past the testing and trialling. And I have put images of all 4 parts of my app. |  |

****

**Discuss how you addressed the Relevant Implications you described and explained earlier. Please provide screenshots where you applied the implication.**

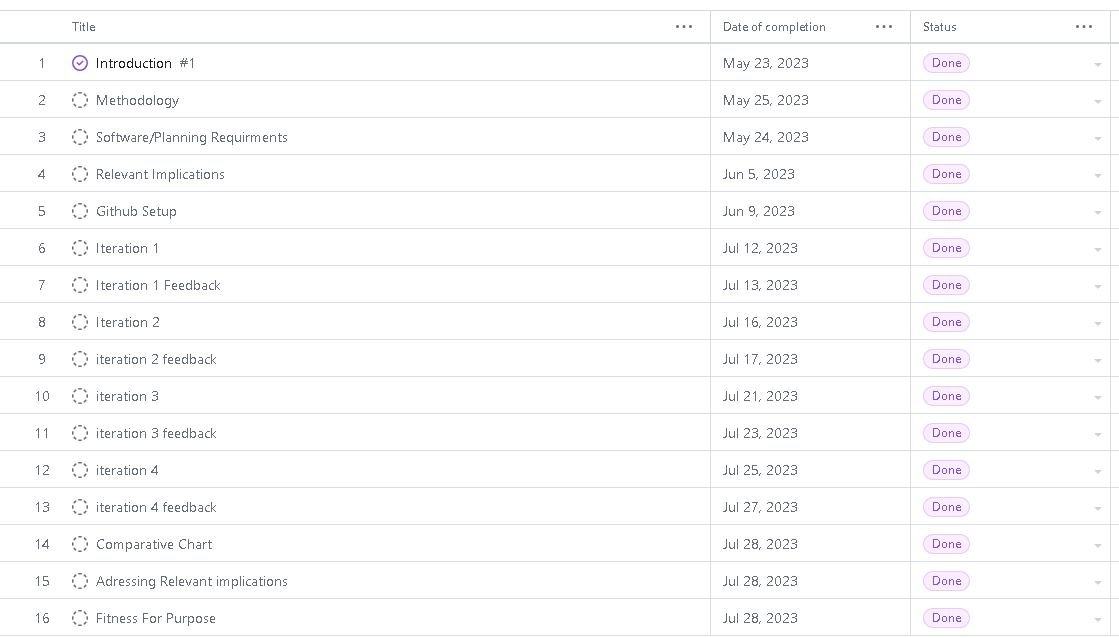
|  |  |
| --- | --- |
| **Relevant Implication** | **How I applied this in the development of my outcome.** |
| Usability | When addressing this implication I first went to a website given to me by the teacher. On that website it had all the information on how to make my app usable.    For my app error prevention, visibility of system status, and efficiency were the most important heuristics.  I made user errors visible using message boxes. I used them in many places such as the calculator, for invalid dates and more:    I made my program more efficient by using menu bar. By doing that the user will efficiently be able to move between the different parts of the app:    And I prevented errors by telling the user what to do for example in the day calculator I put a label informing the user of how to format their date:    By doing all these things I was able to make my app more usable |
| Functionality | I made sure that my app was functional by testing and trialling all the different iterations. By doing testing I was able to reduce bugs in my program, making my app more functional. I also used feedback from my stakeholders to see if there are any bugs in the code. All the evidence of testing and trialling is above this and so is the stakeholder feedback. |
| Aesthetics | To address the aesthetics implication, I first went and looked at what makes an app aesthetic. I found that consistency, user interface and colour schemes are important when it comes to the aesthetics of the app. So, then I found a good colour scheme and designed my program. I have used the same background colour for all the features to show design consistency. Then I used the colours from a palette I liked to have a good colour scheme and finally I used ttk.optionmenu instead of the regular one so the user interface would be better. Image of the use of ttk.optionmenu: |
| Copyright | I addressed this implication in a few different ways. The first way I addressed this was by using GitHub, by using it I was able to commit all my changes. This means that my teacher can see all the changes I have made to the code. Another way I addressed this implication was by giving links to all the videos and websites I used to make this project. By doing this I can show where I got inspiration from, and I will be able to show the developments from watching those videos. Below is a screenshot of my changes in GitHub: |

**Fitness for Purpose:**

**(Discusses how the information from planning, testing and trialling of components assisted in the development of a high-quality outcome. Include how this can be further developed and implemented in the future.)**

Planning was important since it helped visualise how the program would look(wireframe) and how to make the program(flowchart). Without those I would have to spend a lot of time just thinking in my head about how to make the program. By planning everything I just had to write the code. Also, by using GitHub projects I was able to manage my time better. With better time management I was able to achieve a better outcome since all the milestones were planned out for me. I believe that the use of testing and trialling helped me develop a better outcome. This is because if we look at the testing and trialling that I have done we can see the number of bugs that were found in the code. If I hadn’t tested and trialled those bugs would be part of the app and would make the program worse. By testing and trialling I was able to remove a lot of bugs which overall made a higher quality program. I believe that I can develop this app further in the future. I will have to plan out what features to add for future iterations. So once again I will have to use wireframes and flowcharts so I can make the program better. And then I will have to use GitHub to stage the changes which will help others understand the code better.

**Final Trello Board showing all the Tasks Done :**

****

**Make sure you tick all the boxes here.**

|  |  |
| --- | --- |
| **Requirement** | **Status**  **√** |
| Introduction to the project | **√** |
| End users | **√** |
| Project Management Tools such as Trello, Github,Draw.IO used and updated from version to version. | **√** |
| Relevant implications- Described and Explained | **√** |
| Software requirements listed | **√** |
| Program design requirements such as selection, sequence and iteration control structures | **√** |
| Flowcharts | **√** |
| Defined Classes and created objects | **√** |
| GUI wireframes for all versions including annotations | **√** |
| Coding conventions followed | **√** |
| Reads from or writes to files or other persistent storage used | X |
| Defined classes and created objects | **√** |
| Defined and used custom data types | **√** |
| Used complex data structures such as Queues | **√** |
| Trello updated frequently. | **√** |
| Links for Trello, Github and Draw.IO given | **√** |
| Comments written to describe the code | **√** |
| Annotated screenshots or screencast videos showing the testing procedures | **√** |
| Annotated screenshots or screencast videos demonstrating the program is functioning | **√** |
| Documented all the testing procedures using tables provided for each version. | **√** |
| Feedback from end users/stakeholders documented correctly. | **√** |
| Addressed relevant Implications with screenshots | **√** |
| Summary of fitness for purpose | **√** |
| Final Update of Trello board with all tasks done. | **√** |