**Q: Appropriate stages of the software development life cycle were followed and includes a description of each stage (requirements analysis and solution design, solution development, testing)**

**Software Development Cycle**:

Planning

Requirements Analysis (or Business Analysis)

Solution Design

Detailed Design

Construction

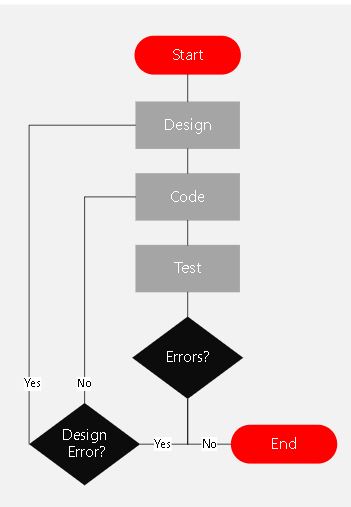
System Testing

Deployment

Operation

Maintenance

The software development life cycle (SDLC) consists of several stages that guide the development and maintenance of software. Here is a summery and how I used it help develop my project.



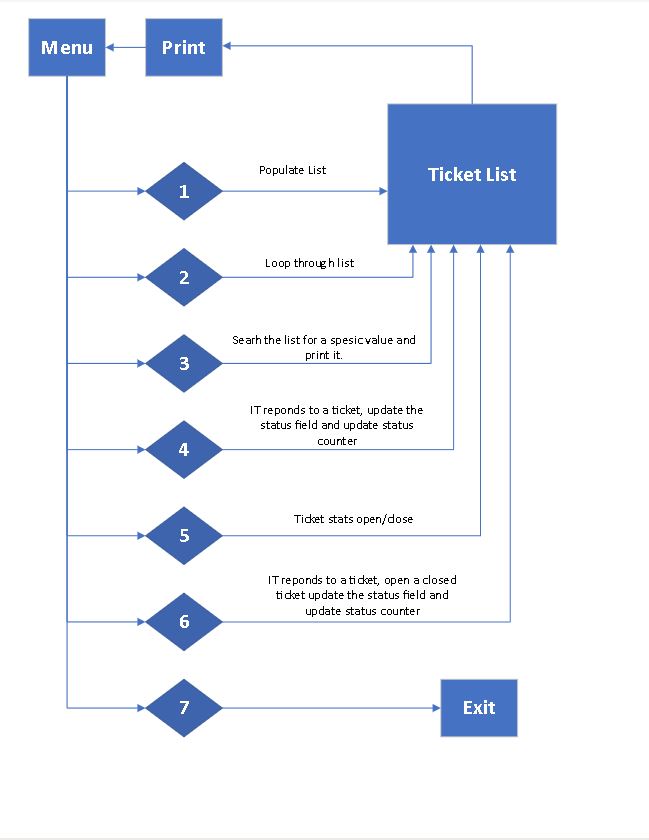
**Requirement Analysis and solution design:**

**This stage involves the gathering and analysing of the program to be developed. It includes the requirements of the stakeholder, understanding the problem and what the desired outcome (functionality and constraints) should look like. The requirements are documented and are used to design a solution.**

**During the solution design stage, the system architecture, modules and user interface are planned. In this stage we aim to have a full understanding of the project scope and a roadmap to develop it.**

I used the Assignment document in Canva and get all the information regarding the project. I did a couple sketches with pen and paper to visualize what the outcome should look like. I watched the video solution provided and made notes on the desired outcome. Then I made a flow diagram on how I see the solution. That has not changed to much from the beginning to the end. Some functionalities were added or moved.

Realizing that at that stage I can implement all the requirements but that I did not have the knowledge to implement all the technical requirements, I had a roadmap to develop my program. I moved onto Solution design with the idea of having a working program I can clean up the code.



**Solution Design:**

**In this stage we code and the implement the software. A working code is derived from the previous stage. Programming standards and best practices are followed to write code. This stage involves collaborating among developers, regular code reviews and version control to ensure the quality and maintainability of the software.**

I jumped ahead to module 4 in course to learn about classes and how to implement them. Lastly, I put in some data validation. I am using IntelliJ IDEA and found the debugging very helpful. Moving onto Module 3 after my program was running.in hind sight I can see that I already used some of the principles and that the rest would have been useful to know before I started. The same can be said about module 5.

I made sure to add my project to Github, although there was no collaboration in this project it proved to be very useful to roll back changes that did not work and to access the code from anywhere.

I tried to use KISS – keep it simple stupid, to avoid complexity and to ensure that I can understand it when I look back at it. I had to repeat code a lot knowing that where ever I repeat code I will be able to come back and create a method. Doing it the long way and then removing repeated code helped me learn and taught me to think like that when designing the code.

I tried to use, Composition > inheritance, by creating an object of the ticket class and then creating instances of the objects.

I also tried to follow the Yagni model, only add features that was requested. I did add a variation of the STAT function.

I tried hard not to optimize prematurely, get a working program first, then optimize. I tried to refactor and kept the code clean.

I tried to avoid S.T.U.P.I.D. design principles and focused on S.O.L.I.D. principles.

I gave my ticket class a single responsibility with methods and functions.

Before I had all the knowledge, I went through this diagram a couple of times. At about vs15 I got my design errors eliminated. From vs15-19 was mostly coding errors.

**Testing:**

**Testing is a very import part of the SDLC as it verifies and validates the software system. It involves running test cases and comparing it to the expected outcome. It ensures that the software meets the defined requirements, works correctly and handles various inputs correctly.**

**There is different test, unit tests, integration testing, system testing and user acceptance testing. If anything is not working as it should, it will be fixed in the next iteration of SDLC.**

I repeated the testing un countable times. Sometimes I had to go back to the solution design and other times all the way back to requirement analysis as I misunderstood a requirement.

I started with just creating a list and testing it. Then adding a menu and creating a list. Later, I added how to change the list values. In the end I created a class and object that I add to a list.

Then it is easy to loop through the objects in the list, change the values or print the values.

Once I had a working program, I started testing what happens when the user enters a wrong selection on the menu, or forgets to type in his name or last name. I added code to validate the user input and to handle errors. Lastly, I tested when user does not put in a valid email address and added code to help remind the user.

**Conclusion:**

**It is important to remember that the SDLC has more stages such as deployment and maintenance. The three stages (requirement analysis and solution design, solution development and testing) are the foundation for the development process. It ensures that the software is built to the requirements, is of high quality and meets the needs of the stake holders.**

In the first versions of my project, I used the waterfall model. This had a sequence that followed from the top to the bottom. This was simple and easy to understand and easy to manage. I could not test any other part of my project until the previous one was working.

After that I moved on to Agile model. As long as that ticket list was created, I could build and test any of the other steps without affecting the others.