Program description

**Identify the problem: - what are you trying to do?**

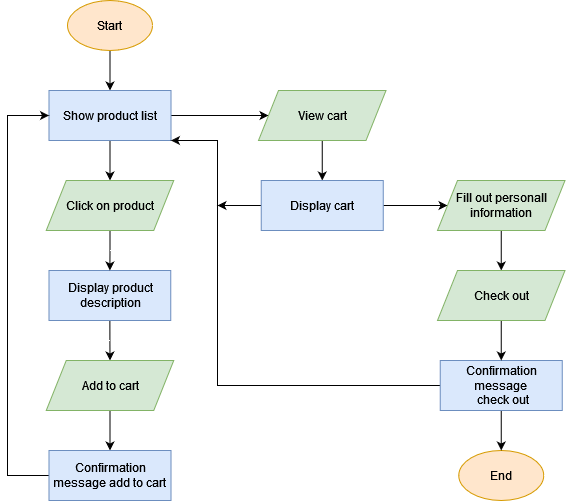
The first stage of developing the program is to identify the problem and solution. In this case the problem presented is the ability to purchase product(s) through a webstore in the shortest amount of time possible. The two steps to find the solution are identifying the requirements and specifications. The requirements consist of the necessary components of the program to reach the goal. The specifications consist of the steps of the program to fulfill the requirements.

* Requirements:
  + Display product list
  + Display product description
  + Add product(s) to cart
  + Check out
* Specifications:
  + List all the available product with name and possible image
  + Select a product to view detailed information
  + Add the selected product to the cart
  + View the cart with the products that have been added
  + Fill out information about full name and address
  + Check out
  + Display confirmation messages after actions take place

**Designing a solution: - how is it going to be done?**

The second stage is designing a solution for the problem. In an effort to implement the solution, the steps the program has to take to reach the goal are formulated. This is done in pseudocode and a flowchart, see figure 1.1. Presented below are the functional steps the program takes to go from start point to end point. In the most basic scenario, the user will view the lists of products, select a product, add the selected product to the cart, view the cart and finally check out. The main actions the program takes are displaying, reading, and storing product information.

1. **BEGIN**
2. **DISPLAY** product list
3. **READ** product selection
4. **DISPLAY** product description
5. **READ** product name
6. **STORE** product to cart
7. **DISPLAY** confirmation
8. **DISPLAY** cart
9. **READ** full name and address
10. **DISPLAY** confirmation
11. **END**



**Writing the program: - what are the specifications? Can we run it?**

For the program to be developed accordingly, it must be coded, compiled, and debugged. Coding the program will depend on the platform and act in accordance with the designed solution. However, it is crucial that the programmer follows guidelines: write as few lines as possible, use appropriate naming, split code into decent size sections and progress minimal error handling. The program must be compiled before it can be run. The computer does not understand the high-level programming language and must be converted into low-level binary language, made up of 0’s and 1’s. Finally, the code must be debugged before the program is finished. This indicates the removal of any existing or potential errors inside the code. If any errors occur in the compiled code, the program might potentially behave unexpectedly or crash. It is of utmost importance to prevent any delays in the testing portion of the thesis project.

Every step the user takes to navigate through the program has to be simple and clear. This implies that the instructions must be formulated properly so that the user can understand its meaning and purpose. Visual components must be displayed with relevant size and position in accordance with web standards. The use of buttons and text fields must be of average size with text size between 18 - 23 px. Each page or section must be clearly divided either by containing an appropriate title or spacing.

* Specify what the layout should look like, interacting should be like – reference

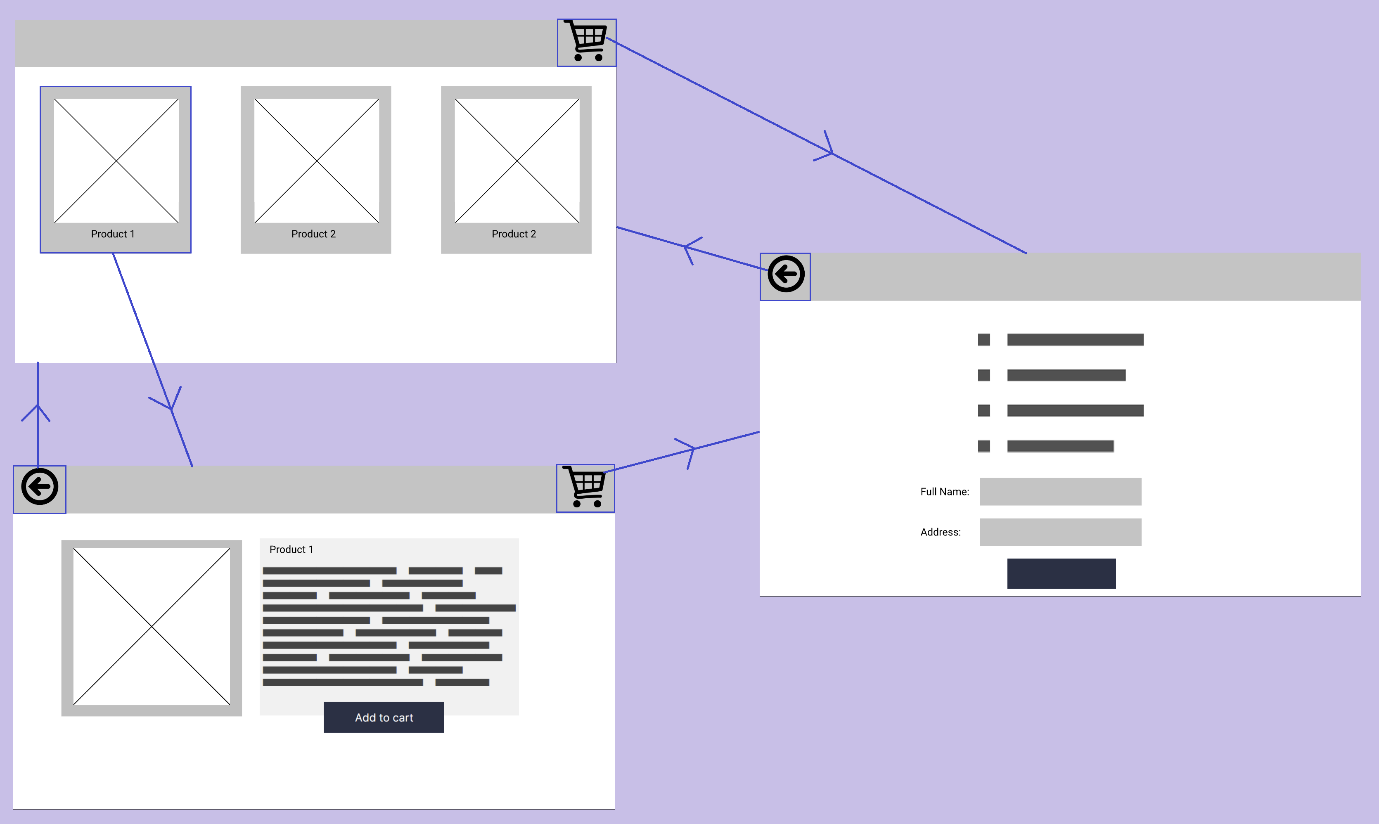
**Check the solution: - is it solving the problem?**

Every listed step the program performs must be tested by the programmer. The next step is not pursued until the previous step is completed. After each step is completed, the programmer tests the finished program to perceive if the requirements meet. The main question declares if the program is serving a solution to the problem.

* Confirm that the program has a solution
* Match requirements to the solution – make comparison

**Technical development web**

The development for the program in relation to the graphical user interface starts by the creation of wireframes. Wireframes are a screen blueprint for the basic functionality and structure of the layout. The frames contain visual representation of pictures, text, buttons, menu bar and context area. Figure 1.2 displays the three frames and the connections between them. The first frame, top-left, represents the product listings each with a picture and title of the product. The user can click on the product div to be redirected to the second frame, bottom-left, to add the product to the cart and view the product description. To view the third frame, middle-right, the user can click on the shopping cart icon located in the top right inside the navigation bar. The icons used are representative of the function performed when clicked. For example: the return button, made from an arrow key pointing backwards with a circle to indicate it is a button. The product description and cart list fill a portion of the page, making sure they are readable and gain attention of the user by the placement in the center of the page.



The languages used for developing the program for the graphical user interface are HTML (Hypertext Markup Language), CSS (Cascading Style Sheets), and JavaScript. HTML is used for the page structure: containing its plain text, title, pictures, buttons, and so forth. CSS is the styling for these elements, giving them shape and color. Finally, JavaScript is used to change the page contents, handle button clicks, and make calculations. The application contains three pages: for the product listings, product description and the cart. The development and design of the program matches the requirements as well as the wireframes.

* How to run the program
* Compiling, debugging
* Creation of a website – steps, reference

**Command Line interface**

The command line interface consists of a single Java class. Each method in the java class performs a unique task requested by the user. Each of these methods can be called several times, for example to return to the products list after adding a product to the cart, or when the user enters an invalid command.

* Debugging, Compiling
* How to run the program
* Option selection methods – references.

Table

Description automatically generated