KINGDOM OF SAUDI ARABIA
Ministry of Higher Education
Al-Imam Mohammad University
College of Computer & Information Sciences

المملكة العربية السعودية
وزارة التعليم العالي
جامعة الإمام محمد بن سعود الإسلامية
كلية علوم الحاسب والمعلومات

Second term 1441/2020

Software Engineering (CS- 310)
Section: 171

Project-Phase No: 1
Computer Recommendation Software (CRS)
(System Requirement Specification)

Submitted By

Saad BinOnayq (439017145) – Coordinator
Fawzan alhantoshi (439014363)
Mohammed Alkhalifah (439011298)
Tareq Alagel (433000438)
Khaled Almodameeg (438013580)

Supervisor

Sultan Alqahtanie

Date: 2020\2\10

Table of Contents


CRS

1. Introduction

This section will cover scope description and overview of everything included in this SRS document, And it also describes all definitions and abbreviations.

1.1 Purpose

The purpose of SRS is to describe the goals of "Computer recommendation software" and its pros and cons for the first version in detail.

1.2 Scope

The "Computer Recommendation Software" is a mobile application which helps people to find computers that fit their field of interests. By recommending a system based on the user's specification, and combine all components in a single convenient page. It also suggests some applications to the user based on the type of the recommended system. The application should also be free and downloadable through a mobile phone application store or similar services.
All items will be provided from websites (such as Newegg[3]) through API. Then, this application will simply process through all the data and display them to the user.
Furthermore, this software requires internet connection to be able to connect to websites to get all the needed data for this application. It also has the capability to get all items by their best value based on the user's region. And for each item it also includes brief description, summary and pros and cons of the recommended system.

1.3 Definitions, Acronyms, and Abbreviations

Term	**Definitions**
SRS	*System Requirement Specification*
CRS	*Computer Requirement Software "This Application"*
API	*Application Programming Interface*
IEEE	*Institute of Electrical and Electronics Engineers*
PC	*Personal Computer*
RAM	*Random access memory*
workstation	*Special computer designed for technical or scientific applications*
Customer	*The end user.*
MySQL	*An open source database management system (RDBMS)*
Dynamic data structure	*A **dynamic data structure** (DDS) refers to an organization or collection of **data** in memory that has the flexibility to grow or shrink in size, enabling a programmer to control exactly how much memory is utilized.[7]*

Binary search algorithms	***Binary search**, also known as half-interval **search**, logarithmic **search**, or **binary** chop, is a **search algorithm** that finds the position of a target value within a sorted array. **Binary search** compares the target value to the middle element of the array.[8]*

1.4 References

[1] IEEE Software Engineering Standards Committee, "IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications", October 20, 1998, http://www.cse.msu.edu/~cse870/IEEEXplore-SRS-template.pdf.

[2] Adobe photoshop, Februray 19, 1990, Adobe, https://www.adobe.com/products/photoshopfamily.html.

[3] Newegg, 2001, Newegg Inc., https://www.newegg.com/.

[4] Adobe Illustrator, January 1987, Adobe, https://www.adobe.com/mena_en/products/illustrator.html.

[5] GIMP, February 15, 1996, The GIMP Development Team, https://www.gimp.org/.

[6] CAD "Computer-aided design", Wikipedia, https://en.wikipedia.org/wiki/Computer-aided_design#Software.

[7] What is dynamic data structure?, 1996, Webopedia, https://www.webopedia.com/TERM/D/dynamic-data-structure.html.

[8] Binary search algorithms, Wikipedia, https://en.wikipedia.org/wiki/Binary_search_algorithm.

1.5 Overview

In section 2 we will describe the system functionality and the user characteristics and the general constraints in 17the system will be mentioned. The third section provides all the functional and non-functional requirements of the system.

2. General Description

This section will give an overview of the whole system. The system will be explained in its context to show how the system interacts with other systems and introduce the basic functionality of it. It will also show how we interact and use websites with our application.

2.1 Product Perspective

The CRS is a mobile system. Through API, this system gets its data from websites and processes them into a single combined product.

By relying on websites to provide data, such as components of a PC, laptops, phones and software . This data will be stored in a local database to be quickly accessible later.

The mobile application has some restrictions about the resource allocation. To avoid problems overloading the operating system the application is only allowed to use 30 megabytes of memory while running the application. The maximum amount of non-volatile memory is 50 megabytes.

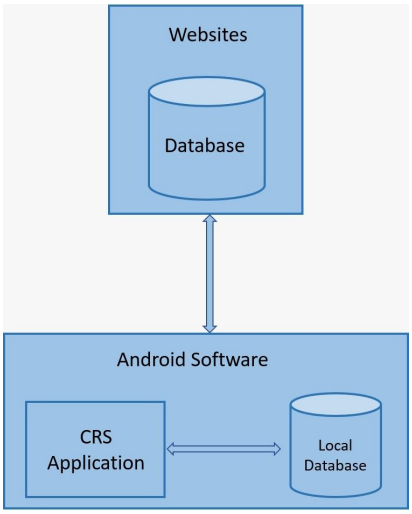

Figure 1

2.2 Product Functions

The "Computer Recommendation System" allows users to search for a specific computer that fits their needs. The result is based on the type of computer the user inputs. If the user's field requires a high performance computer, then the application will focus on workstation computers.

CRS does not only recommend computers, it also suggests software based on the user's profession. An artist would get programs like Adobe photoshop[2], Adobe Illustrator[4] or GIMP[5]. A 3D designer would be CAD[6] or any similar softwares and so on.

The results will be showcased on the result page, with each item listed in any order, by price or name. Clicking any of the items will send the user to the product's page where they could order it.

Manual build is also available in CRS as an alternative for advanced users that want to meticulously build their own computers. Or find other products that fit their niche hobby.

2.3 User characteristics

The two main groups of CRS users are customers, and store personnel. A customer is anyone who is Interested in computers, and quest for computer specifications that match his requirements.

The customer can only use the application to search for products from computer parts, laptops and programs. In order for the customers to get a relevant search result there are multiple criteria the customers can specify

and all results matches depend on their requirement. The amount of product training needed for a customer is none since the level of technical expertise and educational background is unknown. The only skill needed by a customer is the ability to use a phone.

The application personnel will not use the mobile application but instead they maintain the application to update product inventory from set new product, customer service. Furthermore, contact with websites to be provided with items.

2.4 General Constraints

We will focus on the performance and reducing the program size to become more professional, and make the user interface simple as much as possible to be more easier to use the program even for non professional users.

2.5 Assumptions and Dependencies

One of the assumptions is that the application works quickly and efficiently with most modern Android phones that have enough memory, if the phone does not have enough memory or RAM the application will not work unless there is sufficient space in the phone

3. Specific Requirements

This section will contain all the functional and non-functional details of the system requirements.

3.1 External Interface Requirements

This section provides a detailed description of all inputs into and outputs from the system. software and communication interfaces and provides basic prototypes of the user interface.

3.1.1 User Interfaces

Initially the user will see the home page which contains all components, see Figure 2. On this page if the user wants to search for a specific product, he should type the product name in the search bar at the top and will move it to the results page, see Figure 3.
The application could also automatically recommend a system. To do this, the user could simply answer all questions on the main page, then click on the "result" button, see figure 2. After that, the program will process the user's answers, then it will send the user to the result page, see figure 3.
At the end of the main page, there are several suggestions for the most requested products, see Figure 2.

Figure 2

Figure 3

3.1.2 Hardware Interfaces

Since the software does not use any designated hardware, it does not have any direct hardware interfaces. Data will be stored locally and is managed by the underlying operating system on the mobile phone.

3.1.3 Software Interfaces

The mobile application communicates with websites through API in order to get data and to process this data to create a visual representation to the user. The communication between CRS and the local database consists of only reading operations, see Figure 1.

3.2 Functional requirement

This section includes the requirements that specify all the fundamental actions of the software system.

3.2.1 FR1/Update notification/Functional requirement 1

ID: *FR1.*
Title: *Update notification*
Description: *When a new version or update is released, the application should notify the user. Updating the application should be as simple as downloading the application through a mobile application store.*
Inputs: *None*
Source: *None*
Outputs: *None*
Destination: *None*
Action: *First when the user starts the application for the first time a window of notification authorization will pop out and after the user press allow after that update notification will notify them whenever a new version is released.*
Requirement: *User authorization.*
Pre-Condition: *None.*
Post-Condition: *None.*
Side effects: *None.*

3.2.2 FR2/Results page/Functional requirement 2

ID: *FR2.*
Title: *Result page.*
Description: *All items will be listed in order by price or name on the result page. Programs are also suggested alongside the recommended computer. These programs are based on the user's profession. Each item and program is connected to its product page.*
Inputs: *Output from multiple categories (FR4).*
Source: *Choices from multiple categories (FR4).*
Outputs: *The recommended computer and some suggested programs.*
Destination: *Result page.*
Action: *Once the user finished with multiple categories (FR4), the application starts to process the results. Firstly, it updates the local database, then it takes all the data from the newly updated local database and processes them. Once it is done processing, the information will be showcased at the result page.*
Requirement: *multiple categories (FR4).*
Pre-Condition: *The data needs to be available.*
Post-Condition: *The resulted recommended system.*
Side effects: *none.*

3.2.3 FR3/Show Price/Functional requirement 3

ID: *FR3.*
Title: *Show price.*
Description: *There will be a price for each product.*
Inputs: *None.*
Source: *Website's* items.

Output: *Price under the products pictures.*
Destination: *None.*
Action: This function will automatically take the product price from the websites we are currently syncing with, then show it under the product picture, if the price changed in the main website it will change in our application.
Requirement: Item's availability.
Pre-Condition: *The item must be available, otherwise it would show the item as not available.*
Post-Condition: *Shows the product's price successfully.*
Side effects: *None.*

3.2.4 FR4/Multiple categories/Functional requirement 4

ID: *FR4.*
Title: *Multiple categories.*
Description: *to make sure that the user will be served will, we added many categories such as Gaming, Designing, and Student, etc.*
Inputs: *User's inputs.*
Source: *none.*
Output: *none.*
Destination: *Results in result page (FR2).*
Action: *In the main page of the application, each question is followed by checkboxes or radio buttons. Once the user has answered all the questions, then click the result button, after that the application will send the user to the result page (FR2) once it finished processing all the data.*
Requirement: *Available data for each category.*
Pre-Condition: *none.*
Post-Condition: *none.*
Side effects: *none.*

3.2.5 FR5/Discount/Functional requirement 5

ID: *FR5.*
Title: *Discount.*
Description: *If there is a discount it will appear on the right of the price.*
Inputs: None.
Source: *The websites items that we syncing with them.*
Outputs: *Shows the discounted price.*
Destination: none.
Action: Discount will appear automatically if there is a discount on the website's item, and it will change if the discount on the main product in the website changed.
Requirement: Items should be discounted on the main website.
Pre-Condition: *Item is discounted.*
Post-Condition: *none.*
Side effects: *none.*

3.2.6 FR6/Show the best sales/Functional requirement 6

ID: *FR6.*

Title: *Show the best sales.*
Description: *Show the best products to the customer in terms of specifications, quality and price as well.*
Inputs: *The main website's product.*
Source: *Databases of the websites that we syncing with them.*
Outputs: *None.*
Destination: *None.*
Action: *Checking the website sales to compare the previous best selling items with the new ones and replace them with each other and show it in the home page in the app.*
Requirement: *Access to the website sales so we can determine the best selling product.*
Pre-Condition: *None.*
Post-Condition: *The items on the bottom of CRS application home page should be the best selling items.*
Side effects: *None.*

3.2.7 FR7/Sales notification/Functional requirement 7

ID: *FR7.*
Title: *Sales notification.*
Description: *Notifying the user if there is a sale.*
Inputs: *Sales on product in the main websites.*
Source: *None.*
Output: *None.*
Destination: *None.*
Action: *After we get authorization from users to notify them whenever there are sales on the main website's products, the sales will appear on the home page under the best sales category (FR6) in a unique category called sales to show the user all the products that are in sale.*
Requirement: *Databases for each website that our application syncing with them.*
Pre-Condition: *None.*
Post-Condition: *None.*
Side effects: *None.*

3.2.8 FR8/Sort item/Functional requirement 8

ID: *FR8.*
Title: *Sort items.*
Description: *Sorting the items so the user can easily order them and our application becomes more practical.*
Inputs: *None.*
Source: *None.*
Output: *None.*
Destination: *None.*
Action: *Items will be sorted in many ways depending on the user's requirement for example if he wants cheap items, expensive items, items with sale, items which are limited or best sellers, and the user can sort the items in the searching page (FR11) or when the user want to see the product in one of the ways that we mentioned before he checked out in the result page (FR2) and go the main website to buy the products.*
Requirement: *None.*
Pre-Condition: *The user should manually choose the type of sorting that he wants.*

Post-Condition: *None.*
Side effects: *None.*

3.2.9 FR9/Add item/Functional requirement 9

ID: *FR9.*
Title: *Add item.*
Description: *Add item to Bookmark page (FR12)*
Inputs: *None.*
Source: *None.*
Output: *None.*
Destination: *None.*
Action: *User will be able to see an icon looks like a heart in the right of the products image, if the user click this icon the item will be added to the bookmark page.*
Requirement: *The CRS application should have items so the user can add them to his/her bookmark (FR12).*
Pre-Condition: *User must click on the heart icon so the item could be added.*
Post-Condition: *Item must be added to the bookmark (FR12).*
Side effects: *None.*

3.2.10 FR10/Remove item/Functional requirement 10

ID: *FR10.*
Title: *Remove item.*
Description: *Remove an item from the bookmark (FR12).*
Inputs: *None.*
Source: *None.*
Output: *The product will be removed.*
Destination: *None.*
Action: *In Bookmark page (FR12) there will be all the user's products and this function basically will remove the item from the bookmark page by clicking on the red heart icon in the right of the product's picture and the product will be removed from the bookmark page (FR12).*
Requirement: *Bookmark page must be non empty so the user can remove the products from it.*
Pre-Condition: *User must click the red heart icon so the product can be removed.*
Post-Condition: *The selected item should be removed.*
Side effects: *None.*

3.2.11 FR11/Search item/Functional requirement 11

ID: *FR11.*
Title: *Search item.*
Description: *The search function will be implemented so the user could find a specific item.*
Inputs: *String of words, numbers or special letters.*
Source: *The local database.*
Output: *Items with title which include the word.*
Destination: *none.*
Action: *The search bar is on top of the application, see figure 2. Once the user inputs a string, a box will appear under the search bar showcasing all items with title which include the string.*

Requirement: *The local database must not be empty.*
Pre-Condition: *String of words, numbers or special letters.*
Post-Condition: *none.*
Side effects: *none.*

3.2.12 FR12/Bookmark/Functional requirement 12

ID: *FR12.*
Title: *Bookmark.*
Description: *A page that saves the products that the user is interested in.*
Inputs: *None.*
Source: *None.*
Output: *All user's products are in one page.*
Destination: *None.*
Action: *When the user click on a product There will be a little square, under the square will be a word (Bookmark) in the left of the check out button and when this square clicked a check will appears on the square so the user can know that this product saved in the bookmark page and the user can review it whenever he/she wants.*
Requirement: *None.*
Pre-Condition: *The CRS application should have items and the user should manually pick the items that he/she wants.*
Post-Condition: *None.*
Side effects: *None.*

3.2.13 FR13/Link item/Functional requirement 13

ID: *FR13.*
Title: *Link item.*
Description: Link item from websites such as Newegg to CRS Application.
Inputs: *None.*
Source: Website's database.
Output: *None.*
Destination: CRS database.
Action: *Linking databases to CRS application database with the main websites databases by using MySQL server, but we will also need a simple API. Our application won't connect directly to the database, instead, it will need to send requests to an API that we will write. This simple script will take the request, process it, and respond to the app.*
Requirement: *The developer team must get the authorization from the main websites to sync the databases with each other.*
Pre-Condition: *If there are specific rules for each company the CRS developers should follow and agree on them.*
Post-Condition: *None.*
Side effects: *None.*

3.2.14 FR14/Show pictures of products/Functional requirement 14

ID: *FR14.*

Title: *Show pictures of products.*
Description: *For each product there will be a main picture above its name if the product is on the home page but if the product is in the result page the picture will be on the left of its name .*
Inputs: *Images from the local database.*
Source: *Website's database.*
Output: *Show the image for each item.*
Destination: *CRS database.*
Action: *By using the Link item (FR13) function, each item that is stored in the local database will store the item and its image id. Since each individual item already includes its image id, simply loading the item would render these images. On the main page, see figure 2. All items would render the image above its title. On the other hand, if it is on the result page, see figure 3. The images would be at the left of each item.*
Requirement: *The developer team must get the authorization from the main websites to sync the databases with each other. And each image must be approved to be used on this application.*
Pre-Condition: *Each item's image must include an id.*
Post-Condition: *None.*
Side effects: *None.*

3.2.15 FR15/Dark mode and light mode/Functional requirement 15

ID: *FR15.*
Title: *Dark mode and light mode.*
Description: *An additional feature that Darkens the user interface.*
Inputs: *User must click the lamp icon so the dark mode can be activated.*
Source: *None.*
Output: *User interface theme should be changed.*
Destination: *None.*
Action: *When the user clicks the gear icon "⚙" (FR16) on the bottom left of the user interface there will be an icon and the user will see next to it "Dark mode" if the user clicks it the user interface will turn to dark mode "make the user interface darker", if the user interface is already on dark mode and the user click the button again it will change to the main theme which is light mode.*
Requirement: *Two types of modes which are darke mode and light mode so the user can change the interface theme however he/she wants.*
Pre-Condition: *None.*
Post-Condition: *User interface theme must be changed to dark mode or light mode.*
Side effects: *None.*

3.2.16 FR16/Settings/Functional requirement 16

ID: *FR16.*
Title: *Settings.*
Description: *It's the gear icon "⚙" on the bottom left of the user interface, this function gives the user more options.*
Inputs: *User must click the gear icon so he/she can open the settings page.*
Source: *None.*
Output: *None.*
Destination: *None.*

Action: *If the user clicks the gear icon "⚙" the settings page will appear on the user interface, on this page the user will be able to change the theme of the user interface from dark mode to light mode or the opposite (FR15), and there will be five more options which are contact us, about us, notifications, app feedback, rate the app.*
Requirement: *None.*
Pre-Condition: *None*
Post-Condition: *None.*
Side effects: *None.*

3.2.17 FR17/Filter/Functional requirement 17

ID: *FR17.*
Title: *Filter.*
Description: *It's a feature that lets the user organize the items that been shown in the result page (FR2) by different ways.*
Inputs: *User must click the filter button.*
Source: *None.*
Output: *Organized items by a specific way.*
Destination: *None.*
Action: *In the result page (FR2), there will be an icon on the top right, when the user clicks it he/she can organize the items that been shown by different ways for example the most expensive items or cheap items.*
Requirement: *Result page must have items.*
Pre-Condition: *User must be in result page.*
Post-Condition: *Items have been organized.*
Side effects: *None.*

3.3 Non-Functional requirement

This section includes all the non- fundamental requirements that satisfy the software system.

3.3.1 NFR1/Security/Non-Functional requirement 1

ID: *NFR1.*
Title: *Security.*
Description: *Not showing customer privacy and making sure the customer's private information is secure.*

3.3.2 NFR2/Clear interface/Non-Functional requirement 2

ID: *NFR2.*
Title: *Responsive interface.*
Description: *Widescreen, portrait, multi-screen, and tablets will be supported.*

3.3.3 NFR3/Show colors for the products/Non-Functional requirement 3

ID: *NFR3.*
Title: *Show colors for the products.*
Description: *Display the colors of the products, if any, so that the customer can choose the right color.*

3.3.4 NFR4/Memory usage/Non-Functional requirement 4

ID: *NFR4.*
Title: *Memory usage.*
Description: *By using image compression we will be able to reduce the size of all images by a large amount. Thus saving us lots of space.*

3.3.5 NFR5/Result processing speed/Non-Functional requirement 5

ID: *NFR5.*
Title: *Result processing speed.*
Description: *To deal with large data without affecting the system's operating system. We will use a dynamic data structure. And by using advanced algorithms to search, sort and choose the fitting item(s). This will increase the result processing performance tremendously.*

3.3.6 NFR6/Search processing/Non-Functional requirement 6

ID: *NFR6.*
Title: *Search performance.*
Description: *Since our data are sorted. We could simply use binary search algorithms to search for specific items or items with similar names.*

3.3.6 NFR7/Dark mode/Non-Functional requirement 7

ID: *NFR7.*
Title: *Dark mode.*
Description: *The application ui mode will change depending on the user's device setting.*

5. Team Members Contributions

Person	Role	Github page
Saad Bin Onayq	Project Coordinator Deployment Manager Architecture Reviewer Designer User Interface	Saad-BinMansour.github.io
Fowzan Alhantoshi	Project Reviewer Requirement Reviewer Requirement Specifier	fowzan1.github.io/Fkfalotaibe.github.io/
Mohammad Alkhalifah	Test Analyst Implementer Code Reviewer	mssalkhalifah.github.io
Khaled Almodameeg	System Analyst Software Architect	KhaledM.github.io
Tareq Alagel	Test Designer Test Manager	

6. Conclusion

The purpose of this SRS document is to elaborate what application is CRS and how it will serve the end user. Overview, general description and detailed document has been written and reviewed. Functional and non-functional requirement are written in detail, but more functions may be added in the future. System architecture and models will be written in a different document.