**Quick PC**

*PC Building Assistance Mobile Application*

****

**Requirement Documentation**

CECS 491A T/TH 12:30 PM

# **Table of Contents**

[**Table of Contents**](#_id9gyep3b8ui) **1**

[**Executive Summary**](#_isgncmom5jao) **2**

[Context](#_o2cg48sitiu1) 2

[Motivation](#_1domrt37jjcv) 2

[Contribution](#_omyx162wpid6) 2

[Impact](#_qkz4b3864e) 2

[**Stakeholder Model**](#_x95xncon9f2b) **3**

[**Goal Model**](#_up6vzumboxzz) **6**

[**System Vision**](#_97lcu82jr94z) **7**

[**Usage Model & Use Cases**](#_sab2ll54ldb1) **8**

[View Splash Screen](#_5pymnhwu6r4u) 9

[Creating an Account](#_6kfhrz9x1on6) 10

[Login](#_r1dbjz7i48hw) 11

[View Home Menu](#_gak7mcwgkars) 12

[Search for parts with prices, features, etc](#_n4c3hpdjbtgo) 13

[Managing budget of list parts](#_k19idjb6d3y) 14

[Cloud Database Storage: User Login Information](#_a44lbrbkixqd) 15

[Cloud Database Storage: PC Component Information](#_cwqkfzwkc7f2) 16

[Product Alert Sign up](#_n85wt3h4atjw) 17

[Push Notification](#_hmgpoipfxnjq) 18

[Permissions](#_p06tf2d2lcp7) 19

[Component Barcode Scan](#_6q8bpptjpxez) 20

[View Component Reviews](#_dnx9415l7znh) 21

[View Component Comparison](#_1ukc2n5xn9em) 22

[Graphs for Price History](#_zfl1ahc7wpqi) 23

[Suggest PC build](#_fxm1piql39fh) 24

[View a PC Information Page](#_4awk5yf82ru7) 25

[Build Guide](#_92zvdnn1k7fg) 26

[**Functional & Non-Functional Requirements**](#_qo6zg64u967i) **27**

[Functional Requirements](#_g54pj0d7om2x) 27

[Quality Requirements](#_oyz48ewhq1ws) 27

[Constraints](#_nt7vd2fk8nk9) 28

[Development Process](#_m1a53k4zgnuk) 28

# **Executive Summary**

## Context

Quick PC is a mobile application written in Dart using Flutter. The application is designed to help users compare prices and compatibility of PC components. Our app can recommend parts based on the user’s budget and consider what they want to use the PC for. The app can also compare prices from different retailers to find the best deal for users looking for parts.

## Motivation

Due to the covid pandemic, many people have gained interests in building a computer from scratch. Our mobile application intends to assist people in learning the different types of PC components and their respective prices through the use of our mobile application. No such applications are widely available for mobile devices at the moment and the best version of this type of software is typically in the form of a mobile website which is unintuitive to use.

## Contribution

Quick PC will start up asking the user to login to their already registered account or register for a newly made account. Once a user has signed into their account they will arrive at the home screen which is the launch screen. This main page acts as the entry point into our application.

This screen features a guide on how to assemble a PC, a page on what each part of a PC does, performance comparison on certain parts, a search bar for looking up parts, a system builder, and the ability to browse by parts.

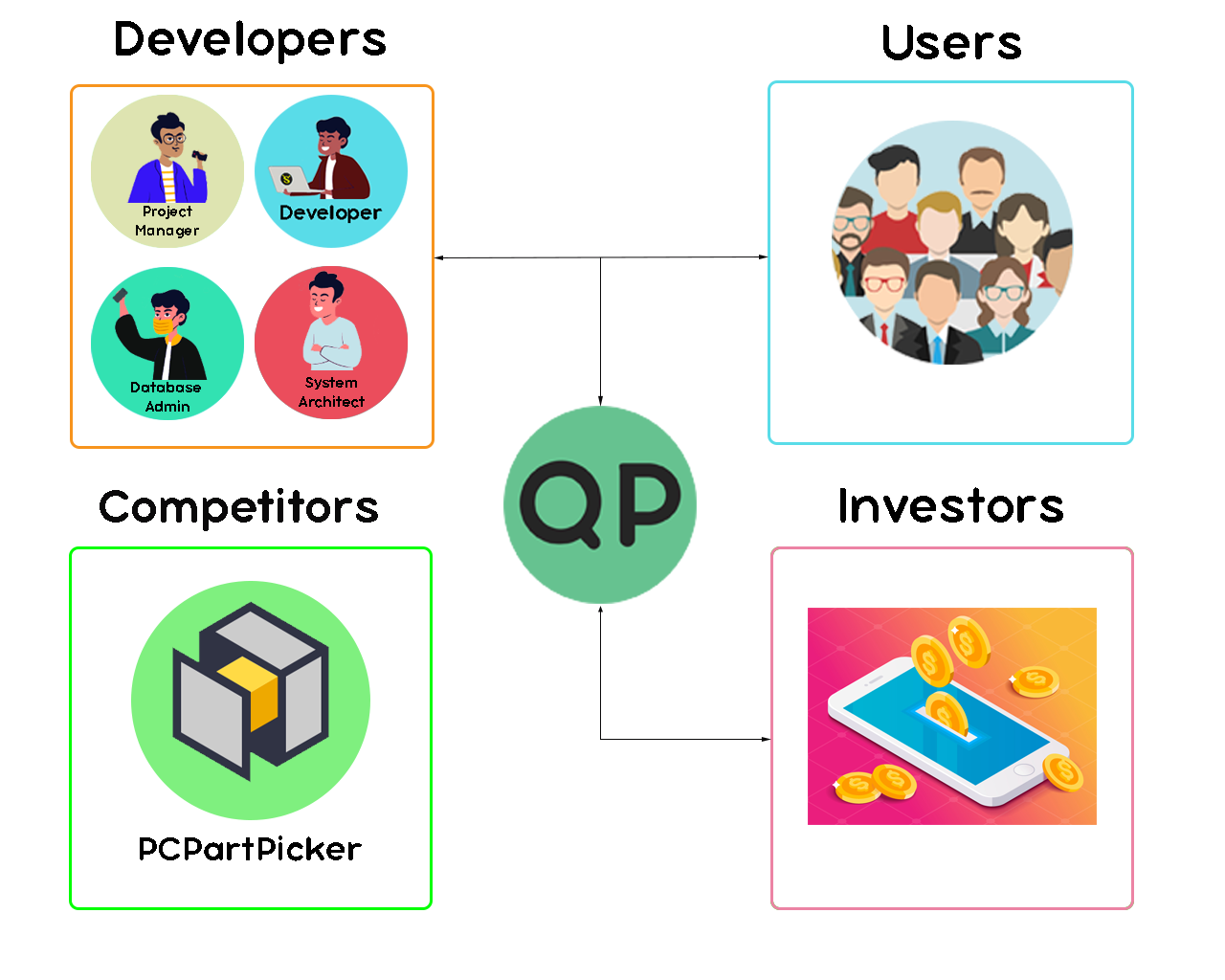
## Impact

The purpose of Quick PC is to give users an accessible and informative way to build and assemble a computer that fits their needs. The app can act as a gateway to learning about PC components and understanding the functions of each part.

# 

# **Stakeholder Model**

The stakeholders in our system all play a large role in the development of our mobile application. The developer group has members such as the project manager who helps manage the project timeline and the database admin who manage our database of PC parts. There is also the system architect who defines the architecture for our system. The users in our stakeholder model are those that register and use QuickPC and they can provide feedback for issues and suggestions for our application. Investors are those that have an interest in our applications success and provide money to develop the project. Lastly PCPartPicker is a website that is a possible competitor to our application and they are indirectly related to our software.



**Developers**

|  |  |
| --- | --- |
| **Representatives** | Anthony Dawson, Gustavo Cisneros, Jose Vasquez, Kevin Vu, Osman Khan |
| **Description** | Software Developers |
| **Type(s)** | Project Manager, Database Admin, System Architect, Developers |
| **Responsibilities** | Develop mobile application |
| **Success Criteria** | Releasing a functional and efficient mobile application that assists users in PC assembly |
| **Involvement** | Maintain and develop mobile application |
| **Deliverables** | Source code, mobile program, documentation |

**Users**

|  |  |
| --- | --- |
| **Representatives** | Users |
| **Description** | People registered with an account on QuickPC |
| **Type(s)** | Active and inactive users |
| **Responsibilities** | Use of application to build computers and find parts suitable for a user’s budget |
| **Success Criteria** | Build a PC or find a part for their budget |
| **Involvement** | Provide feedback about issues, bugs, or suggestions about QuickPC |
| **Deliverables** | N/A |

**Competitors**

|  |  |
| --- | --- |
| **Representatives** | PCPartPicker |
| **Description** | A website that somewhat provides a similar service |
| **Type(s)** | Website |
| **Responsibilities** | Can influence us and provide ideas to improve our own app |
| **Success Criteria** | Gaining a large user base |
| **Involvement** | Acts as a competitor to our mobile application |
| **Deliverables** | N/A |

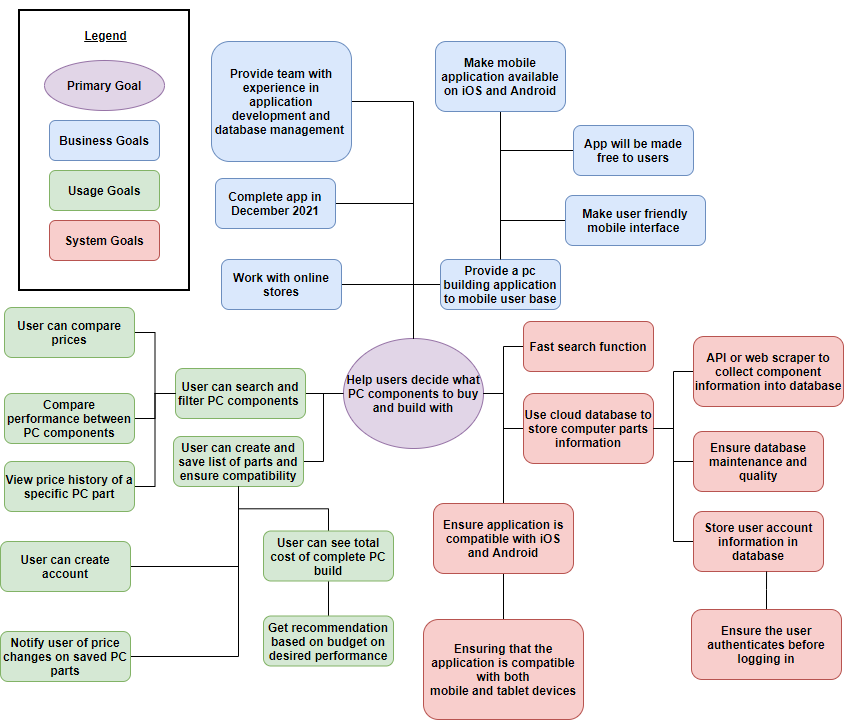
**Investors**

|  |  |
| --- | --- |
| **Representatives** | Developers |
| **Description** | Provide financing for software needs |
| **Type(s)** | Subscription for cloud computing |
| **Responsibilities** | Contributes money for software |
| **Success Criteria** | Ensure success of the mobile application |
| **Involvement** | Finance needs of the software |
| **Deliverables** | N/A |

# **Goal Model**

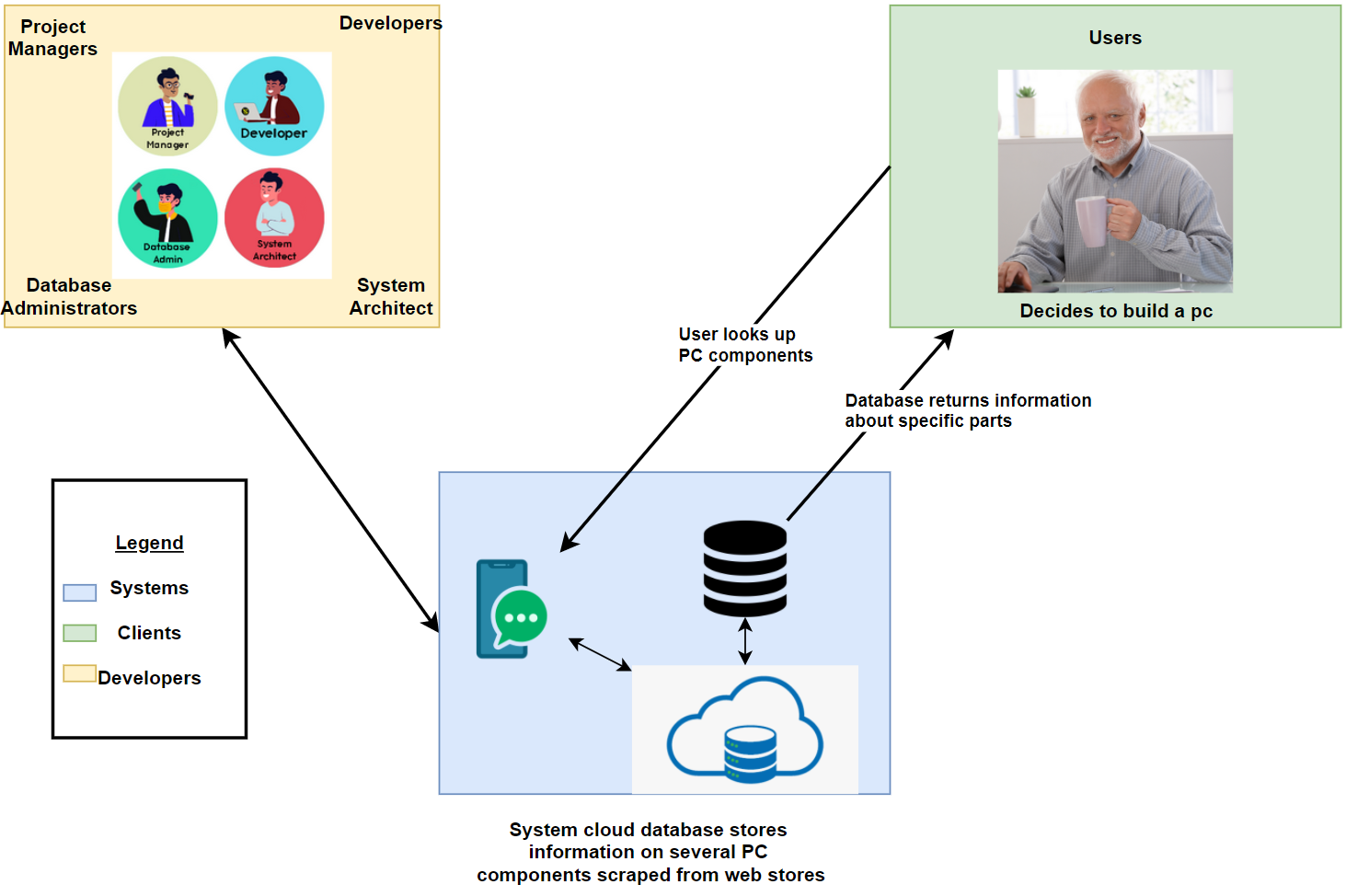
The primary goal of this application is to help users make decisions when purchasing components for building a PC. The user should be able to compare component prices, check component compatibility, and get suggestions for their desired budget or performance.

1. Business Goals: How the application will be present usable.
2. Usage Goals: How users will interact with the app.
3. System Goals: How the application will perform its task

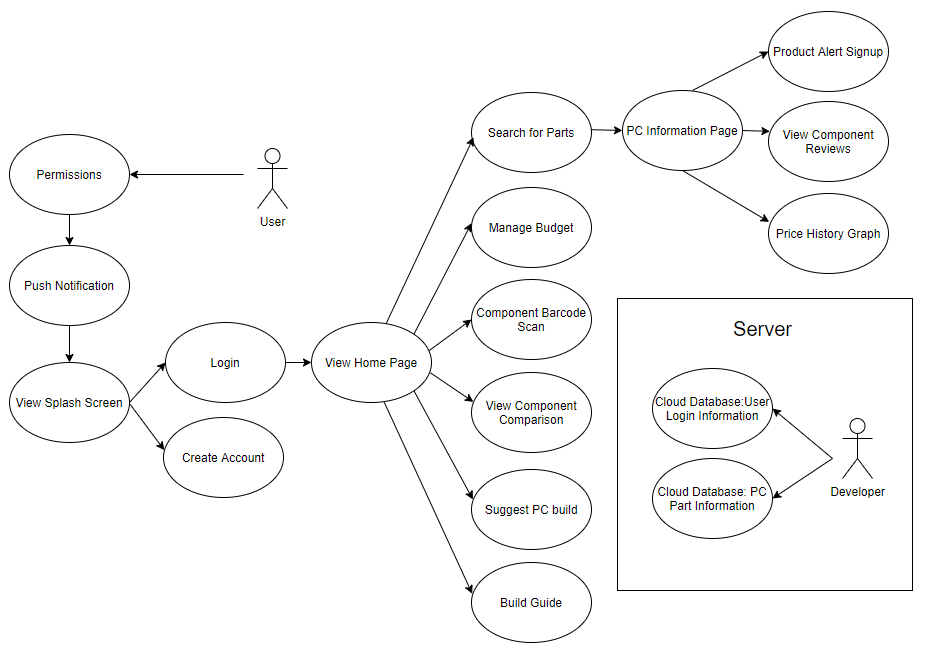


# **System Vision**

The System Vision provides a description of the structure and process of our project. It outlines the features and relationship between the stakeholders, developers, clients, and system. The developers are responsible for any tasks in the development of the application, such as designing, developing, and keeping the system updated. The clients are responsible for using the application itself: creating accounts, forming PC lists, and giving reviews to different parts. PC information is scraped from online retailer websites and is stored in a database for efficient retrieval of information.



# **Usage Model & Use Cases**



|  |  |
| --- | --- |
| **Use Case # 1** | View Splash Screen |
| Author | Kevin Vu |
| Goal in Context | The splash screen will consist of the company logo with the intention to identify/distinguish our company from another |
| Preconditions | User opens the application |
| Success End Condition | The user is directed to the login interface, allowing the user to create an account or login with an existing account |
| Failed End Condition | Application crashes and is not directed to the correct layout. In this case, the login interface |
| Primary, Secondary Actors | Users |
| Trigger | User clicks on the application through their mobile device |
| Description | 1. User clicks on the QuickPC application 2. Logo splash screen is displayed for a brief second 3. User is then directed to the login page, allowing the user to create an account or login to an existing account. |

|  |  |
| --- | --- |
| **Use Case # 2** | Creating an Account |
| Author | Kevin Vu |
| Goal in Context | Allow the user to create an account |
| Preconditions | User does not have an existing account |
| Success End Condition | The newly created account is added to the cloud database that’ll allow the user to login into their newly created account. |
| Failed End Condition | Account already exist and an error message will be displayed |
| Primary, Secondary Actors | Users |
| Trigger | User clicks on create an account |
| Description | 1. User clicks on create an account 2. User fills in their:    1. Username    2. Password    3. Retype Password    4. Email    5. Phone Number 3. Account is added into the cloud database |

|  |  |
| --- | --- |
| **Use Case # 3** | Login |
| Author | Kevin Vu |
| Goal in Context | Allow the user to login into their existing account |
| Preconditions | User has a registered account |
| Success End Condition | User is logged in, allowing the user to search for parts and manage budget list |
| Failed End Condition | User is prompted with an error message saying that the account does not exist or password is incorrect |
| Primary, Secondary Actors | User |
| Trigger | User types in username/password and then clicks “Login” |
| Description | 1. Login panel is displayed 2. User types in the username and password 3. User clicks “Login” 4. User is directed to the menu screen allowing the user to search for parts, manage budget builds, and manage user profile |

|  |  |
| --- | --- |
| **Use Case # 4** | View Home Menu |
| Author | Osman Khan |
| Goal in Context | Acts as launcher page for accessing the rest of the app |
| Preconditions | The user has logged in |
| Success End Condition | The user is viewing the home menu page |
| Failed End Condition | The home page fails to load or launch |
| Primary, Secondary Actors | Users |
| Trigger | After the user has logged into their account they are directed to the home page |
| Description | 1. The user logs into their account 2. The user is not directed to the home page and menu |

|  |  |
| --- | --- |
| **Use Case # 5** | Search for parts with prices, features, etc |
| Author | Gustavo Cisneros |
| Goal in Context | To retrieve a list of similar PC components with prices, and where they are available |
| Preconditions | Application is open, user is viewing the current list of computer parts |
| Success End Condition | User can see the list of computer parts for a particular component, view prices, and select each part to view features |
| Failed End Condition | No parts are returned upon searching for a component list, or not all available information is found/returned when searching for list of parts |
| Primary, Secondary Actors | The user |
| Trigger | When the user clicks the “View Parts” button next to the designated computer component |
| Description | 1. A user is viewing the list of parts currently being assembled 2. The user clicks on “View Parts” 3. A list of currently available parts by different companies is generated 4. The list of available parts is shown to the user as a scrollable list |

|  |  |
| --- | --- |
| **Use Case # 6** | Managing budget of list parts |
| Author | Gustavo Cisneros |
| Goal in Context | To manage the total price of all parts in the current list |
| Preconditions | User logged in, a pc part list must be created, the PC part list must be open |
| Success End Condition | The total price of all parts is added up correctly and is displayed to the user under the list |
| Failed End Condition | The total price of all parts is not added up correctly, or the total price is not displayed |
| Primary, Secondary Actors | The user |
| Trigger | The user creates a list  The user adds a part to their list of parts |
| Description | The user creates a new list profile prior to the budget management  After the user has a new list profile, it will be empty and the budget will be displayed below with the total price being $0 (no parts exist in the list yet)  A user searches for parts under a specific category/component, and adds a particular component to the list  The budget value is re-calculated after adding a part with the total sum of all parts combined. |

|  |  |
| --- | --- |
| **Use Case # 7** | Cloud Database Storage: User Login Information |
| Author | Gustavo Cisneros |
| Goal in Context | The main goal is to store critical user information (user login info) on a cloud database |
| Preconditions | The application must be active, on the login screen, the cloud database must be running and online |
| Success End Condition | A user’s login information is saved onto the cloud database and can be retrieved upon logging in to the application |
| Failed End Condition | A user’s login information is not saved onto the cloud database, or it is uploaded to the database incorrectly somehow, or the data cannot be retrieved from the database |
| Primary, Secondary Actors | Primary Actors: The developers |
| Trigger | The user clicks “Create Account” on the intro screen (log-in screen) |
| Description | 1. The user opens the application 2. The user clicks/taps “Create Account” on the login screen 3. The user enters their email address in the “Email” text box 4. The user enters their password in the “Password” text box 5. The user re-enters their password on the “Confirm Password” 6. The user clicks/taps “Create Account” at the very bottom of the page 7. The user’s account details is uploaded and saved to the database 8. The user is taken back to the login screen |

|  |  |
| --- | --- |
| **Use Case # 8** | Cloud Database Storage: PC Component Information |
| Author | Gustavo Cisneros |
| Goal in Context | The main goal is to store PC component information in a structured database |
| Preconditions | The application must be active, on the login screen, the cloud database must be running and online, the user must be viewing the PC component list |
| Success End Condition | A database table is created/updated with a list of similar components for a particular PC part category, such as name, price, etc. |
| Failed End Condition | A database table is not created/updated with a list of similar components for a particular PC part category, or not all relevant information is uploaded to the database, or incorrect parts are added to the database |
| Primary, Secondary Actors | Primary Actors: The developers |
| Trigger | The user clicks the “View Parts” button next to the component category in the PC part list screen |
| Description | 1. The user is viewing the part list 2. The user taps “View Parts” for a particular PC component 3. A query is generated to search for each part under the specified category 4. The database is populated with each component and their information 5. The list of computer components and any relevant information is displayed to the screen |

|  |  |
| --- | --- |
| **Use Case # 9** | Product Alert Sign up |
| Author | Jose Vasquez |
| Goal in Context | Allows users to subscribe to product in-stock alerts |
| Preconditions | The user is on the reference page of the component |
| Success End Condition | The user adds the specified product to their notification list after clicking the notification button. |
| Failed End Condition | The component is not added to the notification list |
| Primary, Secondary Actors | The user |
| Trigger | The user clicks the “ notification” button |
| Description | 1. The user is on the reference page for the specified product 2. The user clicks the notification button under the product image. 3. The product is added to the notification list |

|  |  |
| --- | --- |
| **Use Case # 10** | Push Notification |
| Author | Jose Vasquez |
| Goal in Context | Allows the user to activate push notification on their device |
| Preconditions | Device permissions are granted by the user |
| Success End Condition | Notification are activated |
| Failed End Condition | Device permissions are not allowed therefore push notifications are denied |
| Primary, Secondary Actors | The user |
| Trigger | User clicks the checkmark box to activate push notifications. |
| Description | 1. The user clicks on the “Options” tab under the drop menu 2. The user clicks on the “Activate Notification” checkmark box |

|  |  |
| --- | --- |
| **Use Case # 11** | Permissions |
| Author | Jose Vasquez |
| Goal in Context | Allows the user to grant access to permissions on their device |
| Preconditions | Application updated to the latest version |
| Success End Condition | Permissions are granted by the user |
| Failed End Condition | Permissions are denied by the user |
| Primary, Secondary Actors | The user |
| Trigger | Initial app startup will prompt the user with a device permissions prompt |
| Description | 1. Updated application is launched for the first time 2. The user is prompted with a grant permissions screen 3. The user accepts/declines requested permissions |

|  |  |
| --- | --- |
| **Use Case # 12** | Component Barcode Scan |
| Author | Jose Vasquez |
| Goal in Context | Scanning component barcodes will provide users with additional part information. |
| Preconditions | Application permissions are granted by the user |
| Success End Condition | Scanning a barcode will launch a third party website with information regarding the specified product |
| Failed End Condition | Scanning the barcode will return a “No results” error message when nothing is found |
| Primary, Secondary Actors | The user |
| Trigger | The user click on the “Scan Barcode” tab |
| Description | 1. The user clicks the “Scan Barcode” tab located under the home page menu 2. Camera view is activated 3. The user taps on screen to focus on the barcode 4. A link is displayed on screen that will provide the user with more information about the product |

|  |  |
| --- | --- |
| **Use Case # 13** | View Component Reviews |
| Author | Jose Vasquez |
| Goal in Context | Displays part review listings. |
| Preconditions | The user is located on the reference page of the specified product |
| Success End Condition | A list of reviews are displayed for the product |
| Failed End Condition | The user will be granted with a “No Reviews” display message if no reviews are found |
| Primary, Secondary Actors | The user |
| Trigger | The user clicks on the “View Review” button |
| Description | 1. The user selects the “View Review” button 2. A drop down of reviews is displayed towards the bottom of the product tab |

|  |  |
| --- | --- |
| **Use Case # 14** | View Component Comparison |
| Author | Kevin Vu |
| Goal in Context | Compare components between each other such as the specification between two different graphics cards, CPU, RAM, etc. |
| Preconditions | Two components are chosen from the parts database |
| Success End Condition | Exact specification are displayed between the two components |
| Failed End Condition | If two different types of components are chosen, an error message will be displayed saying that the same component must be chosen in order to compare |
| Primary, Secondary Actors | User |
| Trigger | User chooses “Component Comparison” in the Menu page |
| Description | 1. User clicks on the “Component Comparison” in the main menu 2. User is directed to the Component Comparison interface 3. Two boxes that contain a list of parts user can choose from to compare 4. Once chosen, specifications of the two parts will be shown |

|  |  |
| --- | --- |
| **Use Case # 15** | Graphs for Price History |
| Author | Osman Khan |
| Goal in Context | Compare and the see what the price for a PC component was over a certain timespan |
| Preconditions | A certain PC component must be picked and viewed in order to bring up its price history |
| Success End Condition | A graph is displayed of the price of a component |
| Failed End Condition | No graph is displayed |
| Primary, Secondary Actors | User |
| Trigger | User scrolls to section about price history on PC component page |
| Description | 1. The user selects a certain PC component to view 2. On the generated page for the PC component the user can scroll to the price history 3. This price history displays the price for the component over a timespan |

|  |  |
| --- | --- |
| **Use Case # 16** | Suggest PC build |
| Author | Osman Khan |
| Goal in Context | Recommend PC parts that fits user budget and fulfills user’s purpose |
| Preconditions | User is on home page and selects suggest a PC build from the menu |
| Success End Condition | Displays suitable builds that the user can view |
| Failed End Condition | No builds displayed for any budget |
| Primary, Secondary Actors | User |
| Trigger | User selects the suggest build option on the home page |
| Description | 1. User selects suggest Pc build on the home page menu 2. The user is asked what their budget range and selects a value 3. Suggest builds are displayed for office or gaming builds |

|  |  |
| --- | --- |
| **Use Case # 17** | View a PC Information Page |
| Author | Gustavo Cisneros |
| Goal in Context | To be able to click on a PC component in a list and view its information |
| Preconditions | A list of PC parts must be open |
| Success End Condition | A page is loaded with an image of the PC part and its appropriate information |
| Failed End Condition | A page is not loaded with the image of the PC part, or not all the information is loaded into the page |
| Primary, Secondary Actors | User |
| Trigger | User clicks on an item list element in a PC part list |
| Description | 1. The user is viewing a list of PC parts 2. The user taps on one of the PC parts in the list 3. The PC component’s information is loaded onto the page |

# 

|  |  |
| --- | --- |
| **Use Case # 18** | Build Guide |
| Author | Anthony Dawson |
| Goal in Context | Provide users with resources to guide them in assembling a PC |
| Preconditions | User is on the home page and selects “Build Guide” from the menu |
| Success End Condition | User is presented a list of guides or videos for building intel or amd pcs, and links on how to install specific components. |
| Failed End Condition | None of the guides are displayed or non of the links work |
| Primary, Secondary Actors | User |
| Trigger | User selects “Build Guide” from the home page menu |
| Description | 1. User selects “Build Guide” from the home page menu 2. A list of guides is presented to the user 3. The user selects a guide and reads or watches them |

# **Functional & Non-Functional Requirements**

## Functional Requirements

* Login system
  + Allow users to register and login to their pre-existing account
  + Store saved parts into the account’s database
  + Feature would allow users to share their computer part list with other users through a link to their profile or a newly created shared link
* Database storage (Firebase)
  + Databases storing account information and customer part list
* Search engine system for hardware parts
  + User is able to filter parts by price, brand, new parts, most viewed
* Price history graphs
  + Display a graph to the user with the price of a certain part over a time frame
* Build suggestion
  + Users are able to set a price range for their build and the system will recommend computer builds
  + Users are also able to specify if they are using the PC for gaming or work
* Part ordering system
  + Users are able to sort by name, date released, most viewed, most recommended, etc

## Quality Requirements

* User friendly
  + Ensure the application functions are easily accessible
  + Efficient code - ensuring the program runs efficiently and effectively
  + Ensure the price tracking functions is up to date
* Application is secure
  + Ensure customer’s account information is encrypted
  + Ensure that the database is secure against SQL injection, brute-force, and privilege escalation
* Maintain the CIA triad
  + Make sure that data is private and well maintained
  + Ensures data has not been tampered with
  + Ensure the application is attack proof from denial-of-service-attack
  + Ensure that the application is available at all times

## Constraints

* Ensure that the project has enough funding for servers, databases, etc
* Mobile friendly (iOS, Android)
* Ensure the application is fully functional
* UI is done on Adobe XD
* Application is written in Dart using Flutter
* Cloud Database (FireBase) is used to store customer information

## Development Process

* Complete documentation for the application
* Set up Google Firebase / SQLite to store customer information
* Become familiar with Flutter and Adobe XD
* Develop the UI through Adobe XD
* Develop the mobile application through Flutter using Dart
* Bug testing - ensuring the application does not crash when accessing certain app functions
* Deploy the application
* Application maintenance - making sure the application, parts recommendation, and prices are up to date