**Functional Requirements**

The system must ask the user to sign up for an account.

The system must ask the user to log in if they already have an account.

The system must allow the user to search for products.

The system must allow the user to start a build list.

The system must recommend to the customer builds that are similar to the user’s list.

The recommended lists must be both less than the total and more than the total cost of the user’s list.

The system must allow users to compare products.

The system must provide the user with ample information for them to make a decision.

The system must allow the user to have multiple build lists.

The system must have a layout such that the user can follow along to it easily.

The user must have an account to use the system

The system must allow the user to choose parts that are compatible with other parts

**Nonfunctional Requirements**

The system shall be able to handle many users at once

The system shall be able to correct mistakes made by the user.

The system shall be easy to use even to the less computer literate.

The system shall not divulge any of the user’s personal information.

The system shall not be down at peak hours.

The system shall be updated and maintained at a time where there is minimal traffic to the site.

The system must be able to work as expected on any operating system and device.

The system shall be able to recover from any errors quickly.

The system shall be secure with its data.

The system shall not cause any harm to the user.

The system shall be easy to navigate.

The system shall accomplish its tasks with as few clicks as possible.

The system shall not be large in size.

The system shall not take up too much of the computer’s resources.

**USER STORIES:**

Alice wants to build her own personal computer. She heard about the web application called Budget Builder. She decides to make an account and try the application. She makes the account and successfully logs in. Alice is greeted with a homepage that asks her to either start a new build, continue a build or browse the products. She decides to browse the product options. By clicking the “browse products” tab, she is presented with a dropdown menu that shows her different categories of PC parts that she can choose from e.g. “memory”, “CPU”. By clicking “Memory” she is then presented with a page that shows a list of all the memory brands, speeds and the capacities of the memory. She can then either “add” the specified memories to a list of her products or simply view how the product looks by clicking the picture of the memory.

Bob works for a company that needs a PC built that will maximize performance and minimize the cost of building the PC. He decides to use the budget builder app to make the pc build in software first. He logs into the app and chooses “New Build”, he is then met with a layout page that specifies all the necessary computer parts with the selections empty, just to be filled in. He clicks on “add” on the motherboard section and he adds a motherboard to the list. He then clicks on “CPU” and then chooses the appropriate CPU that is compatible with his motherboard. Bob then completes his build and the system gives him an accurate total of all the parts’ cost so he can know and would be able to show his company proof of the total of the build option he chooses.

Candice built a PC using the application but she does not like the total cost she received. She wants to reduce the cost of the build that she has. She decides to choose “alter build” in the build section. Her saved build is shown on the screen, she does not want to do over a complete build. There is an option that says “Similar builds” where she can click and the site will take her to a page that shows her a list of builds similar to hers, but the price range is either less than her build or more than her build. The site recommends to her a build that is very similar but the “GPU” is less expensive than the choice she had originally chosen. She decides to choose this build instead and clicks on the “save spec list” option for that specific build.

Darren wants to compare two CPU’s since he is not familiar with the architecture of both. He opens the app and chooses “compare parts”, he then chooses the option to choose which category of components he wants to compare. He selects the two CPUs and clicks compare. This gives him a list of details showing the comparison between the two CPUs, showing details such as pricing, TDP, cores etc.

Loki is unfamiliar with computer components and unsure of the parts he may need to make a build. He navigates to the “Generate Build” section and then chooses what he wants his computer to do. He is then presented with a list of different builds at different price ranges and from there he can select something to suit his price range. If he so chooses, he can select any build that he shows interest towards and get a listing of the parts required with the cost of each part listed.

Paul has a list of parts that he would like to obtain however he is unable to locate any in his general area so he opts to purchase them online. He compiles his build on the builder and is then given a few credible sources that he can then choose from. He can click a link to take him to his desired part, such as “CPU” on the desired website e.g. “Amazon”. After making a purchase he can then return to the builder and get sources for his other parts to complete his build.

Tom is a developer and maintains the website and its components. When an issue arises, he must make every effort to resolve it and because of this he is able to log into the system as an admin. The system will check his credentials and once confirmed he is granted access to the system with admin privileges.

**USE CASES:**

**ID:** UC1

**Name:** Login

**Actors:** Customers (users of website)

**Preconditions:** Must have an existing account.

**Description:** An Actor is able to log into their accounts where they can find saved information and access certain features of the website.

**Postconditions:** Actor is logged on.

**ID:** UC2

**Name:** Create New Account

**Actors:** Web surfers

**Preconditions:** Must not have an existing account.

**Description:** A new/potential visitor to the system can choose to create an account to access more features of the system.

**Postconditions:** Visitor now has a n account on the system.

**ID:** UC3

**Name:** Start New Build

**Actors:** Users

**Preconditions:** Must have an existing account.

**Description:** A user is able to start building a new PC system by selecting this function after logging in. Here they can select

**Postconditions:** Persons with access to accounts can now start a new build for their pc using

the system. People with accounts can save their build and resume or review it at a later time.

**ID:** UC4.0

**Name:** Save Build (extended)

**Actors:** Users

**Preconditions:** Must have an existing account.

**Description:** A user is able to save a build that they have compiled or is compiling.

**Postconditions:** After selecting “save build” the user’s account is cleared of all the parts that they previously compiled.

**ID:** UC4.1

**Name:** Edit Build (extended)

**Actors:** Users

**Preconditions:** Must have an existing account, must have an existing build.

**Description:** A user is able to edit a build that they were previously creating. Once they log onto the website, they can continue a build that they were doing at an earlier time once it was saved.

**Postconditions:** After selecting this option an existing user will be presented with the compilation of parts they had previously.

**ID:** UC4.2 (extended)

**Name:** Delete Build

**Actors:** Users

**Preconditions:** Must have an existing account.

**Description:** A user is able to delete a build that they were compiling previously after reviewing it.

**Postconditions:** After selecting “delete build” the user’s account is cleared of all the parts that they previously compiled.

**ID:** UC5

**Name:** Search Item

**Actor:** User

**Description:** The user wants to find a product on the website. They click the search button and type in the product that they desire.

**Precondition:** The user wants to search for an item. the site contains such item.

**Postcondition:** the site is used to search the product

User clicks on the product

User reads the description of the product

User adds the item to their list

Product is found and used appropriately.

**ID:** UC6.0

**Name:** BrowseCategories.

**Actors:** Users, Visitors

**Preconditions:** Actors must be on the website.

**Description:** If Actor is just curious about the components(parts) of a PC (desktop) they can browse through the categories of parts like “Memory” or “CPU”.

**Postconditions:** Actors will most times than not find parts to their liking in a selected category.

**ID:** UC6.1

**Name:** Compare Items.

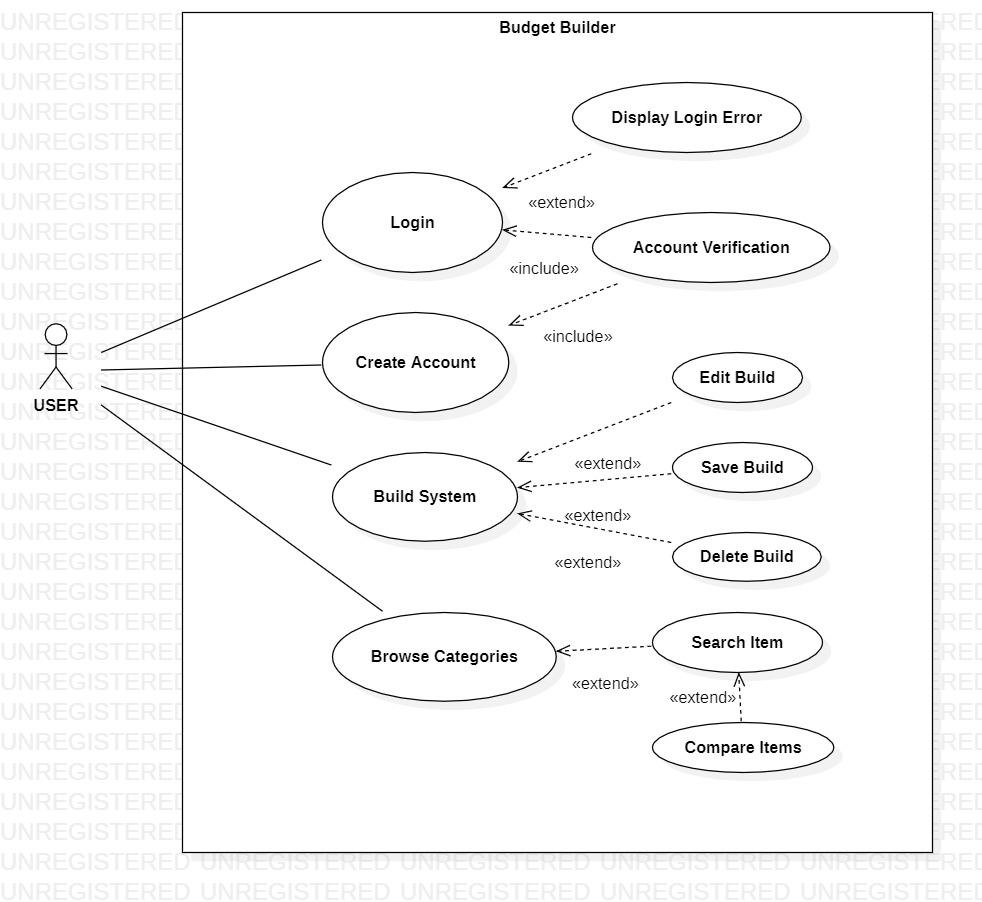
**Actors:** Users, Visitors

**Preconditions:** Actors must be on the website.

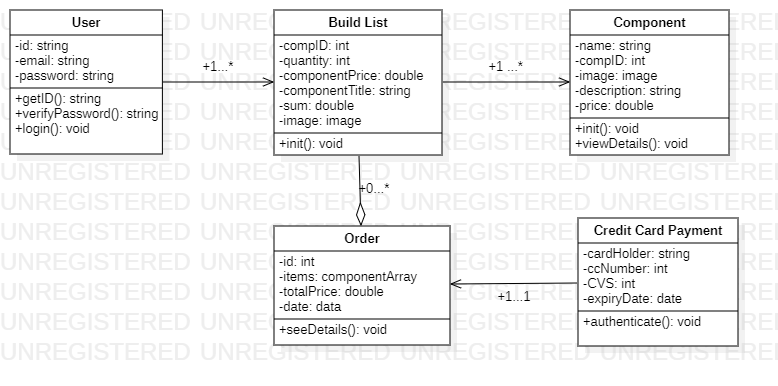
**Description:** If an Actor wants to compare components(parts) of a PC (desktop) they can browse through the categories of parts and compare different brands and specifications.

**Postconditions:** Actors will compare the price and specifications of different parts when making a choice.

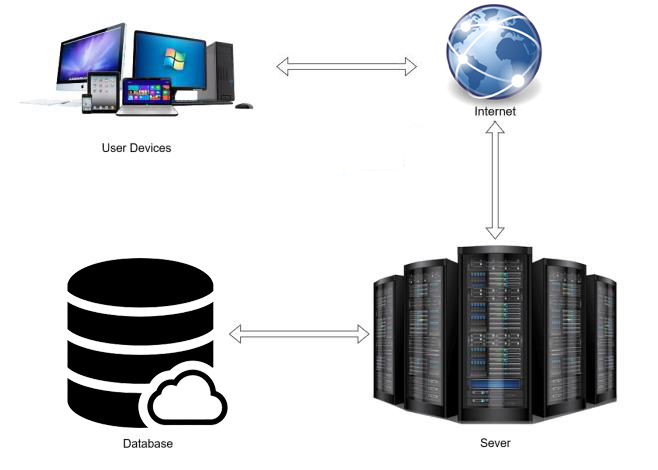
**USE CASE DIAGRAM**



**Class Diagram**



**Logical Design**



**Technologies, Languages, Libraries:**

**• Technologies:**

- Flask Framework

- MongoDB

**• Application (webapp):**

- HTML

- JavaScript

- Stylesheet API for styling

- Hosted using Google Firebase

**• Database:**

- MongoDB

-Google Firebase

• **Libraries:**

**App Navigation**

• **Login:**

- signup

- Build System

**• Signup:**

- login

• **Build System:**

- Component Details

- Add Component

- Remove Component

- Navigation Drawer

• **NavigationDrawer:**

- My Build List

- Components

- Community Builds

- Account

• **Cart:**

- Build Details

- Component Overview

- Checkout

• **Checkout:**

- Order History

- Navigation Drawer

- Cart