الغلاف الخارجى للبحث

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| أولاً: البيانات الخاصة بالطالب | | | | | | | | | |
| **الفرقة الدراسية** | **الثانيه** | | | **التخصص** | | | **General** | | |
| **اسم القسم** | **general** | | | | | | | | |
| **اسم المقرر** | **1 Software engineering** | | | | | | | | |
| **استاذ المقرر** | **DR/Amr sabry ghoneim** | | | | | | | | |
| ثانياً: البيانات الخاصة بالبحث | | | | | | | | | |
| **عنوان البحث** | **Product Review Analysis For Genuine Rating** | | | | | | | | |
| **طبيعة المشاركة** | **بحث فردى** | | | | | **بحث جماعى** | | | |
| **ارسال البحث** | **بواسطة البريد** [**الالكترونيnahlaahmed.na1@gmail.com**](mailto:الالكترونيnahlaahmed.na1@gmail.com) | | | | | | | | |
| **اسماء الطلاب المشاركين فى البحث**  **(يكتب الاسم رباعيا)** | **م** | **الاسم رباعى** | | | | **رقم الجلوس** | | | **الرقم القومى** |
| **1** | **نورا يار عباس علي** | | | | **2709** | | | **30009250104644** |
| **2** | **مارتتينا مسعد جرجس ابسخرون** | | | | **2537** | | | **30008280100183** |
| **3** | **مهرائيل الفي رشدي** | | | | **2687** | | | **29901200103781** |
| **4** | **منه الله عبد المجيد محمد** | | | | **2683** | | | **29911270100343** |
| **5** |  | | | |  | | |  |
| **تاريخ الإرسال** | **7 / 6 / 2020** | | | | | | | | |
| ثالثاً: البيانات الخاصة بالكونترول | | | | | | | | | |
| **النتيجة** | | | **ناجح** | | **راسب** | | | | |
| **أعضاء لجنة تقييم البحث** | **الاسماء** | | | | | | | **التوقيع** | |
| **1** |  | | | | | |  | |
| **2** |  | | | | | |  | |
| **3** |  | | | | | |  | |

|  |  |
| --- | --- |
| **فى حالة عدم قبول البحث يرجى ذكر الأسباب** | * **..............................................................................................................................................** * **..............................................................................................................................................** * **..............................................................................................................................................** * **.............................................................................................................................................** |

**Product Review Analysis of Genuine Rating**

**Table of Contents**

**Part1:**

**1. Introduction**

1.1. Purpose

1.2. Intended Audience

1.3. Project Scope

1.4. System Perspective

1.5. Glossary and Abbreviations

1.6. List of the System Stakeholders

1.7. References

**2. Interface Requirements**

2.1. User Interfaces

2.2. Software Interfaces

2.3. Communications Interfaces

2.4. Hardware Interfaces

**3. SRP definition**

**4. Nonfunctional Requirements**

4.1. The General Categories of Non-Functional Requirements

4.2. Non-Functional Requirements Specification, the Fit-Criteria and the impact on architecture of the system

**5. Functional Requirements**

5**.**1. User Requirements Specification.

5.2. System Requirements Specification.

5.3. Requirements’ Priorities

**6. Domain Requirements Specification.**

**7. Design & Implementation Constraints.**

**8. System Evolution:**

8.1. Anticipated changes.

8.2. How should any anticipated changes in the future affect the system design?

**9. Requirements discovery approaches**

**10. Requirements validation techniques**

**Part2:**

**1. Structural & Behavioral diagrams:**

1.1. Class Diagram ver.3

1.2. Design Smell.

1.3. Class Structuring Criteria.

1.4. Forks/Cascades (advantages/dis advantages).

1.5. Package Diagram.

1.6. Object Diagrams.

1.7. Database Specification.

1.8. State-Machine Diagram.

1.9. Deployment Diagram.

**Part 4:**

**1. Complexity &testing:**

1.1. Quality Factors.

1.2. LOC/CCM.

1.3. Complexity Metrics.

1.4. White Box Testing.

1.5. Black Box Testing.

1.6. Heuristic Evaluation.

* **Team roles**

**Part 1:**

1.Introduction:

* 1. **Purpose:**

This “Software Requirements Specification (SRS)” documents key specifications, describes a prototype in terms of functional and nonfunctional requirements for product review analysis for a genuine rating.

The information documented helps the intended audience to design and develop the product. There will be a need for future updates of this document as we are planning to launch a prototype version for testing then start officially the beta version then the final version.

* 1. **Intended Audience:**

This project is a prototype for product review analysis for a genuine rating.

And it is restricted within the college premises. This has been implemented under the guidance of college professors. This project helps the users to get product of best choice and best price.

* 1. **Project scope:**

The scope of the project is to help the users to get product of best choice and best price.

The system provides the users registration form to login, then these registered users are provided with a platform of multiple products added by the admin that helps them: to rate a particular or multiple product and give review about the products if they want.

The system also takes these reviews from other various users based on their opinion (the users also allowed to see the rating and the other’s feedback)

And that helps users to compare between products of the same kind an choose the best.

* 1. **system Perspective:**

The main aim of product review analysis website is to allow users to collect all comments, reviews, rates about a certain product offered by our website according to these collected data users can compare between any two selected items, users are also able to provide review and rate about the products and a feedback about the website. The system gives a list of the best recommended products according to their rate and price. It will be the only system on the market that provides only rating and reviews about the product and recommend the best products. There are other kinds of software for buying and rating products like: amazon, souq.com, jumia and other staff. Even though they cover a wide range of interests, none of them are specifically designed for rating and review products.

* 1. **Glossary and Abbreviations**

**There is any abbreviations in our project so I’m going to define general terms.**

**SRS**: Software Requirements Specification.

**SRP**: Software Requirement Pattern.

**Functional requirements**: describe how a system must behave, what its

features and functions.

**Nonfunctional requirements**: describe the general characteristics of a

system. They are also known as quality attributes.

**Domain requirements:** Domain requirements are the requirements

which are characteristic of a particular category or domain of projects.

* 1. **List of the System Stakeholders**

**Admin:** The system has only admin how manage the website.

**Users:** The system has many users who benefit from system services.

* 1. **References:**

1.<https://krazytech.com/projects/sample-software-requirements-specificationsrs-report-airline-database>

2.<https://lostclouds.com/2Communicate/project/SRS.pdf?fbclid=IwAR2eDJJfKGk12YGocf8K_u8fUaNYazHoyHmywNx1bDdz9yxR5Ge5SL1SHN8>

3.<https://l.facebook.com/l.php?u=http%3A%2F%2Fwww.cse.aucegypt.edu%2F~rafea%2FSATA%2FReports%2FRequirementsSpecifications-Rafea-3.pdf%3Ffbclid%3DIwAR1_TvpD6WGyql_2gbe-NdxQTVNsvGd_FkG47tAImG8aj_lEoahru-1TDMc&h=AT14gcePFrHufkpd5nj7f9j0Gh5WW2MfFlHhaFCbEirNrlfVhj8EFQY1LPshxrZZPgck7REFK6BV8Soid18D0iW8lxgnEUJBG8cd-W3YAj24arp4qnEyS8okzFhd2JyAqPtkAg>

4.<https://www.altexsoft.com/blog/business/functional-and-non-functional-requirements-specification-and-types/>

5.<https://www.geeksforgeeks.org/software-engineering-classification-of-software-requirements/>

6.<https://jserd.springeropen.com/articles/10.1186/s40411-017-0038-9>

7. <https://dev.to/arquitectosoft1/non-functional-requirements-the-pillars-of-a-solid-software-architecture-2mbg>

10. <https://flylib.com/books/en/4.445.1.154/1/>

11. <http://www.ofnisystems.com/services/validation/user-requirement-specifications/>

12. <https://www.practicalecommerce.com/using-moscow-prioritization-for-ecommerce-projects>

2. Interface Requirements:

**2.1. User Interface**

The interface will visualize the features and functionalities listed in this document for this prototype as the included below not limited to:

* Drop down menu for category selection
* Push buttons for admin to add, edit products
* push buttons for user to compare products, update information
* Visual graphs to show feedbacks and reviews
* Tables to show products, list of users, feedbacks

**2.2. Software Interface**

* MySQL database server with intensive use of memory space.
* PHP server with high performance.
* Windows users’ computers.

**2.3. Communications Interfaces**

Internet connection and a web browser are required in order to make use of several functions and to be executed such as searching, viewing feedback and rating.

**2.4 Hardware Interfaces**

* Windows
* A browser which supports HTML & JavaScript.

3.SRP Definition

**-** A Software Requirements Pattern (SRP) is an artifact that provides guidance on requirements specification, functional, non-functional, so that they can be reused in well-defined contexts and problems.

**-** In software development, there are requirements that are similar in nature or that appear frequently in many softwires, indicating a possible pattern

**-** Usually the non-functional requirements can be reused independently of the domain, i.e., they appear in the same way in different projects. On the contrary, the reuse of functional requirements, in most cases, is only possible for a given software domain. A pattern has a presentation structure that corresponds to formalization and documentation to capture and reuse knowledge.

**-** The use of patterns helps software engineers in reusing successful solutions to develop new projects, eliminating the redundancy of defining a set of requirements for each software project, which in turn reduces the workload and skill required for communication among stakeholders.

4. Nonfunctional Requirements:

**4.1. The General Categories of Non-Functional Requirements**

1.Product Requirement

-Efficiency Requirement

2.External Requirement

- Legislative Requirement

**4.2. Non-Functional Requirements Specification and**

**the Fit-Criteria**

|  |  |  |  |
| --- | --- | --- | --- |
| **Property** | **Category** | **Measure** | **Impact on system architecture** |
| Reliability | Product Requirement | 1.Mean time to failure  2.Probability of unavailability  3.Rate of failure occurrence  4.Availability  “The system should be failure safe” | Implementing a clear and well-deﬁned function in the system. |
| Portability | Product Requirement | 1.Percentage of target dependent statements  2.Number of target systems | Responsive Design able to be used in different environments |
| Usability | Product Requirement | 1.Training time  2.Number of help frames  ‘’The way of using the system should be simple for all users’’ | Simple interface, all users will be able to use it with minimum technical skills  After less than half hour users will be able to use it fluently |
| Performance | Main category: Product Requirement  Subcategory: Efficiency Requirement | 1.Processed transactions/second  2.User/event response time  3.Screen refresh time | Simple interface, any operation can be done in less than 3 steps to increase speed and ease of use |
| Security | Main category: External Requirement  Subcategory: Legislative  Requirement | 1.The ability of the system to protect itself against accidental or deliberate intrusion | 1.Error messages to notify users about any wrong data  2.Encrypted data |

-

ceived as an independent component of the system, which

performs a clear and well-deﬁned function in the system

A module is con-

ceived as an independent component of the system, which

performs a clear and well-deﬁned function in the system

A module is con-

ceived as an independent component of the system, which

performs a clear and well-deﬁned function in the system

A module is con-

ceived as an independent component of the system, which

performs a clear and well-deﬁned function in the system

A module is con-

ceived as an independent component of the system, which

performs a clear and well-deﬁned function in the system

A module is con-

ceived as an independent component of the system, which

performs a clear and well-deﬁned function in the system

A module is con-

ceived as an independent component of the system, which

performs a clear and well-deﬁned function in the syste

5. Functional Requirements:

**5.1 User Requirement Specification**

The user requirement specification (URS) is a document usually used in software engineering that specifies what the user expects the software to be able to do.

1. The system shall allow users to register
2. The system shall allow both users and admin to login
3. Users should be able to update their information ex: password
4. The products must be added by admin
5. Admin should be able to edit products information
6. Admin should be able to delete any product from the system
7. Both admin and users shall be able to see products
8. Both admin and users shall be able to select specific category
9. Admin should have access to users list
10. Both admin and users shall be able to get products rate and review
11. Admin should have access to system feedbacks
12. Admin shall be able to delete any irrelevant feedback
13. Users shall be able to add feedback about the system
14. Users should be able to add review and give a rate to any product
15. System should be able to calculate average rate for each product
16. Users should be able to compare between products
17. Both users and admin shall be able to logout of the system
18. The system shall allow users to search for a certain product
19. The system should allow both users and admin to add any product to their wish list
20. The system shall allow admin to upload files
21. Admin should be able to generate files

**5.2. System Requirements Specification.**

|  |  |
| --- | --- |
| **Function Name** | **Description** |
| Register | 1.The unregistered user will be able to register in the system.  2.The user should fill all required data like username, email, password.  3.The Email used must be registering for the first time. |
| Login | 1.both admin and registered users are allowed to login  2.They should provide the data required for login process such as email and password.  3.Email and password should be correct and user’s email should be registered before. |
| Update Details | 1.User can update their information like password and username.  2.for updating process, users should write the email they used to register.  3.Then, they can add the information they want to update and write the same data as old ones in the other fields. |
| Add products | 1the admin is the one responsible for adding products.  2.The admin should provide all the data required in order to add product to the system such as product name, price.  3.the seller information should also be added to the system while adding the product. |
| Edit Products | 1.Admin can edit any product data.  2.The admin should write the id of the product needed to be edited  3.Then, the updated should be written in their fields and the other data should be written same as the old ones. |
| Delete products | 1.Admin can delete any product from the system.  2.The admin should write product id in order to delete it.  3.the id must be already exist. |
| View products | 1.A table of products will be displayed for both users and admin.  2.The table will contain data about products and product’s seller information.  3.The table will also contain a link to add any product to wish list and other for viewing rate and review for any product. |
| Select category | 1.The products will be categorized to three different categories, men, women and kids.  2.admin and users can select any category from a dropdown menu.  3.The products table will be the same as the table viewed in the home page. |
| View users list | 1.All system user’s information will be preserved.  2.the admin will only have access for user’s information such as username and email.  3.A list will be displayed for the admin containing this information only. |
| View Rate/Review | 1.After some users adding rate and review to the products.  2.all users and admin will be able to see the reviews and rates provided to any selected product.  3.the product id should be written to view the product rates and reviews. |
| View Feedback | 1.After the users adding feedback to the system based on their personal experience the feedback will be saved in the database.  2.The admin and users will have the access to see system feedbacks.  3.The feedbacks will be displayed in a table.  4. admin can access feedbacks by two ways, when admin get a notification that someone has added a feedback admin can press on the notification and a table of feedback will be displayed or admin can press on the feedback button to get all system feedbacks |
| Add feedback | 1.After using the system users will be able to add a feedback about the system to talk about their own experience.  2.The user must write it’s email and the feedback in order to add a feedback.  3.A notification will be sent to the admin to notify him/her that someone has added a new feedback. |
| Delete feedback | 1.the admin will have the access to delete any irrelevant feedbacks.  2.the admin should provide the feedback id in order to delete it. |
| Add Rate/Review | 1.All products in the system can be rated and reviewed by the users.  2.the user should press on rate/review button and fill the form that will be displayed.  3.the user should know the id of the product he/she wants to review it. |
| Calculate average Rate | 1.Each product that have been rated will have an average rate.  2the average rate will be calculated by the system and displayed in products table. |
| Compare | 1.from existing products, users will be able to choose any two products to compare between them.  2.Users should press on compare button and a form will be displayed.  3.users should write the two products ids, the two products information will be displayed. |
| Logout | 1.users and admin will be able to logout of the system by pressing on logout button.  2.they will be returned to the login page. |
| Search | 1.the users will be able to search for any existing product by writing it’s name in a search bar and press om submit.  2.the search bar will be appear after pressing on search button that will take the user to a new page with the search bar.  3.the result will be displayed in a table. |
| Add favorite | 1.the existing products can be added to admin or users wish list.  2.The products table will have a star shape, while pressing on it, it will open a form the user or admin should fill it in order to add the item to their wish list.  3.The information needed to be written is product name, id and product category. |
| Upload files | 1.the admin will be able to upload files of a certain type png or jpg for example.  2.after pressing om upload files button a new page will be open, containing a button to choose the file admin wants to upload and then submit.  3.only files of a certain type will be uploaded. |
| Generate files | 1.the admin will be able to export some data from the website and download it in a form of pdf. |

**5.3. Requirements’ Priorities**

**-** We will use MoSCow scheme to illustrate our project requirements’ priorities:

**-** The capital letters in MoSCoW stand for “must have,” “should have,” “could have,” and “won’t have.”

**- The categories are typically understood as:**

* **Must have**

Requirements labeled as “*must have”* are critical to the current delivery timebox in order for it to be a success. If even one *Must have* requirement is not included, the project delivery should be considered a failure

* **Should have**

Requirements labeled as “*Should have* “are important but not necessary for delivery in the current delivery timebox. While *Should have* requirements can be as important as *Must have*, they are often not as time-critical or there may be another way to satisfy the requirement so that it can be held back until a future delivery timebox.

* **Could have**

Requirements labeled as” *Could have”* are desirable but not necessary and could improve the user experience or customer satisfaction for a little development cost. These will typically be included if time and resources permit.

* **Won't have (this time)**

Requirements labelled as “*Won't have”*, have been agreed by stakeholders as the least-critical, lowest-payback items, or not appropriate at that time. As a result, Won't have requirements are not planned into the schedule for the next delivery timebox. Won't have requirements are either dropped or reconsidered for inclusion in a later timebox.

|  |  |  |  |
| --- | --- | --- | --- |
| Must have | Should have | Could have | Won’t have |
| 1-Register  2-login  3-logout  4-view products  5-Add rate/review  5- Add feedback  7-view rate/review  8- view feedback  9-Add products  10-Edite products  11- Update details | 1-select category  2- search  3-Delete feedback  4-Delete products  5- Add favorite  6-view users list  7-comapre  8-calculate average rate | 1-Upload files  2- Generate files | 1-Add to cart |

6. Domain Requirements

* Domain requirements comes from the application domains of the system
* The development team has to be successful enough to create the database and the interface design
* Information about the requirements stored in the database helps the developers to understand the requirements and develop the system according to these requirements
* The new functional requirements and the constraints that reflect the characteristics of the system has to be clear enough to avoid some difficulties as the problem of **understandability and problem of implicitness**

1. **Problem of Understandability: when**requirements are expressed in the language of the application domain nature, but software engineers aren’t often understandable
2. **Problem of Implicitness:** domain specialist understands the areas well but don’t express it Clearly that makes the development team isn’t able to understand and implement the requirements due to the incomplete information

7. Design and implementation constraints

1- All information utilized in the system must be stored in database that's accessible by the website(product information ,user information,….etc.)  
2- Xampp and localhost are going to be used as SQL engine and database  
3- User can use system any time because it is running 24 hours each day  
4- User can access from any other computer that has internet connection  
5- User must register to use this system  
6- User must have their username and password

8. System evolution

**8.1. Anticipated changes**

* Product review analysis is a web application consists of products added by the admin, system takes reviews of various users, based on their personal opinion that helps the user to get his product at its best choice and price.
* The system allows the users to look into a product with different rating and reviews and the average rate of that product, and more information about the price, seller name, seller location
* Users are allowed to compare between two products of same category, they are also allowed to add their favorite product
* **the system is not easy enough to use as new users may need some manual guidance to make some issues more clear**
* **The new system is going to have some simple adjustments and additions to make it more reliable**

1. Add favorite: user will click only on the star and he wouldn't have to fill the form
2. Rating: rating form will be replaced by stars instead of filling the form
3. Review: user will write his review inside the text field without entering product id
4. Products will be provided with different colors and images
5. There will be offers
6. Feedback the system: the user won’t have to enter his email
7. Users will be able to put some products in a shopping cart to choose what he is going to buy
8. Users will be able to make orders
9. Users will be able to buy products online
10. Users can pay online
11. Users are able to contact with managers

**8.2.**

* According to the user needs and hardware anticipated changes:
* as requirements change through changing business circumstances, the software that supports the business must evolve and change therefore the design has to be responsive to accept the approaching changes
* design is going tobe straightforward and effortless that make the users to feel comfy while using it
* users might be ready to use this technique on android and IOS phones and on Ipad

9. Discovery approach

“use case diagram technique” is applied within the discovery approach  
UML use-case diagram helps in studying the system feasibility by identifying the actors (admin ,user) and their environment ,therefore the function which they collaborate in, what's the include and extend action of each

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Functions** | **Admin** | **User** | **Include** | **Extend** |
| Register |  | **-** |  | **-mailExist**  **- password confirmation** |
| Login | **-** | **-** | **View product** | **-invalid** |
| Logout | **-** | **-** |  |  |
| View user | **-** |  |  |  |
| View rate/review | **-** | **-** |  |  |
| View feedback | **-** | **-** |  |  |
| Delete product | **-** |  | **Deleted successfully** | **Id does not exist** |
| Add product | **-** |  | **Added successfully** | **Id Product exist** |
| Edit product | **-** |  | **edited successfully** | **Id does not exist** |
| Delete feedback | **-** |  | **Deleted successfully** | **Id does not exist** |
| View men | **-** | **-** |  |  |
| View women | **-** | **-** |  |  |
| View kids | **-** | **-** |  |  |
| Update details |  | **-** | **Data updated** |  |
| Add Rate/review |  | **-** | **Added successfully** |  |
| Calculate average rate |  |  |  |  |
| Compare |  | **-** |  |  |
| Add feedback |  | **-** | **Added successfully** |  |
| Search |  | **-** |  | **Product not found** |
| Upload files | **-** |  | **Uploaded successfully** | **extension not allowed** |
| Generate files | **-** |  |  |  |
| View product | **-** | **-** |  |  |
| Add favorite |  | **-** | **Added successfully** | **Id product exist** |

* **In tabular description:**
  + Use cases presented in a table of actor actions and system ... its evolution from its initial discovery to the assemblyof its fully detailed.
  + It also helps to decide if there’s any changes we had to form to make the design and the usage more clear and reliable by knowing the normal flow of the functions if they are simple or complicated
* **Sequence diagram**
  + shows the interaction logic between the objects in the system within the time order that the interactions happen
  + example :(when an admin wants to view products system will send a request to the database to show the stored data, the database will send the stored data in response that makes the admin view products) so it clears the interaction between the system and the database and the flow of events in additional details

10. Validation

**“Requirement review technique” is applied to check for the requirement validation. Check list is used to ensure that there’s no source of error that consists of some points:**

* Completeness checks
* Consistency checks
* Validity checks
* Realism checks
* Verifiability
* correctness

After reading SRS document and by checking and testing. the program is correct as it behaves as expected in the SRS documentation

1-all functions are implemented as user’s desire

2-there is no conflict between the requirements

3-implementation constraints are correct

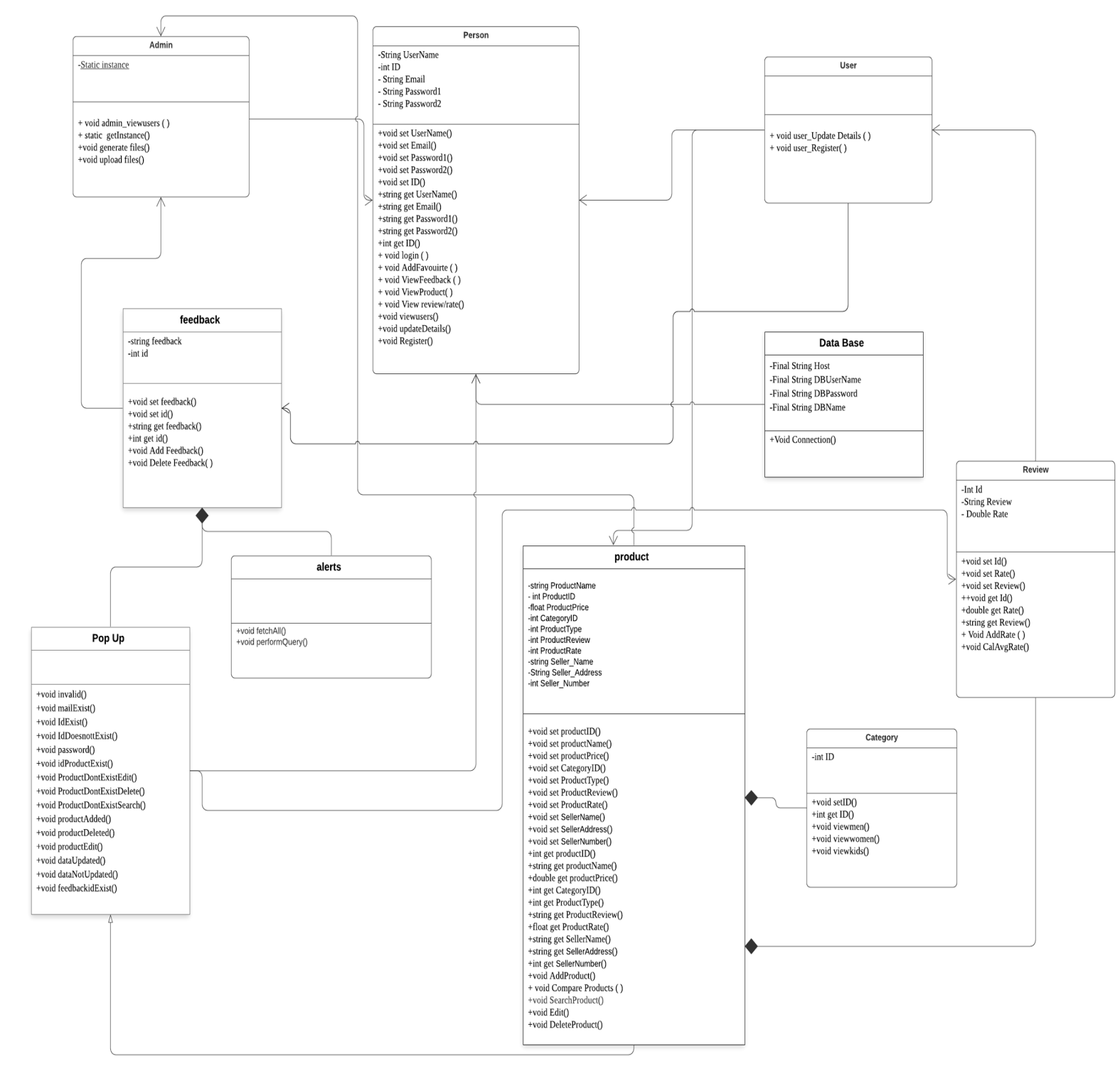
4-all requirements are supported by the existing technology

**But there’s some functions aren’t easy enough to use like (add favorite, review and rate) so there are some changes will happen in the near feature**

**Part 2:**

1. Structural &Behavioral diagrams:

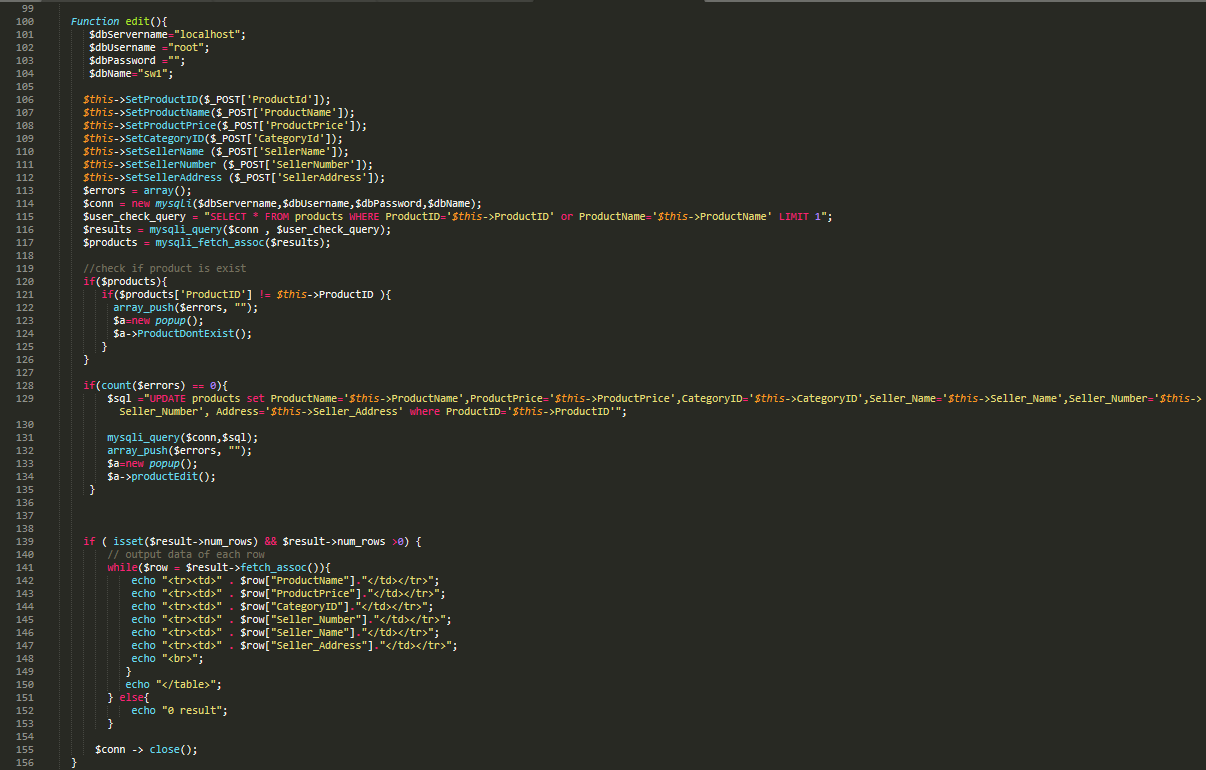
***1.1. Class Diagram ver.3***



***1.2. Design Smell:***

**-** In computer programming, design smell is a structure in the design that indicates violation of fundamental design principles and negatively impact design quality.

**-** The origin of the term “design smell” can be traced to the term “code smell”.



* **Long method in our code.**

The perfect method should be between 4 to 20 lines so we can extract a couple of lines into a new method, decompose conditional.

* **Large class.**

Person class is the largest as we can extend class (or subclass).

***1.3. Class Structuring Criteria:***

|  |  |
| --- | --- |
| **Application Class** | **Classes** |
| Entity object | Database |
| Boundary object | Admin Dashboard, user Dashboard |
| Control object | product, Review, Feedback, Category, popup, Admin, user, person |

1. **Entity object:** 
   * A software object, that encapsulates the business model, including rules, data, relationships, and persistence behavior, for items that are used in our business application.
   * It is used in the database class, that encapsulate the data and provides specific persons to access that data.
2. **Boundary object:**

- Objects that interface with system actors (e.g. a **user** or **external service**).

- Screens and menus are examples of boundaries that interface with users.

1. **Control object:**

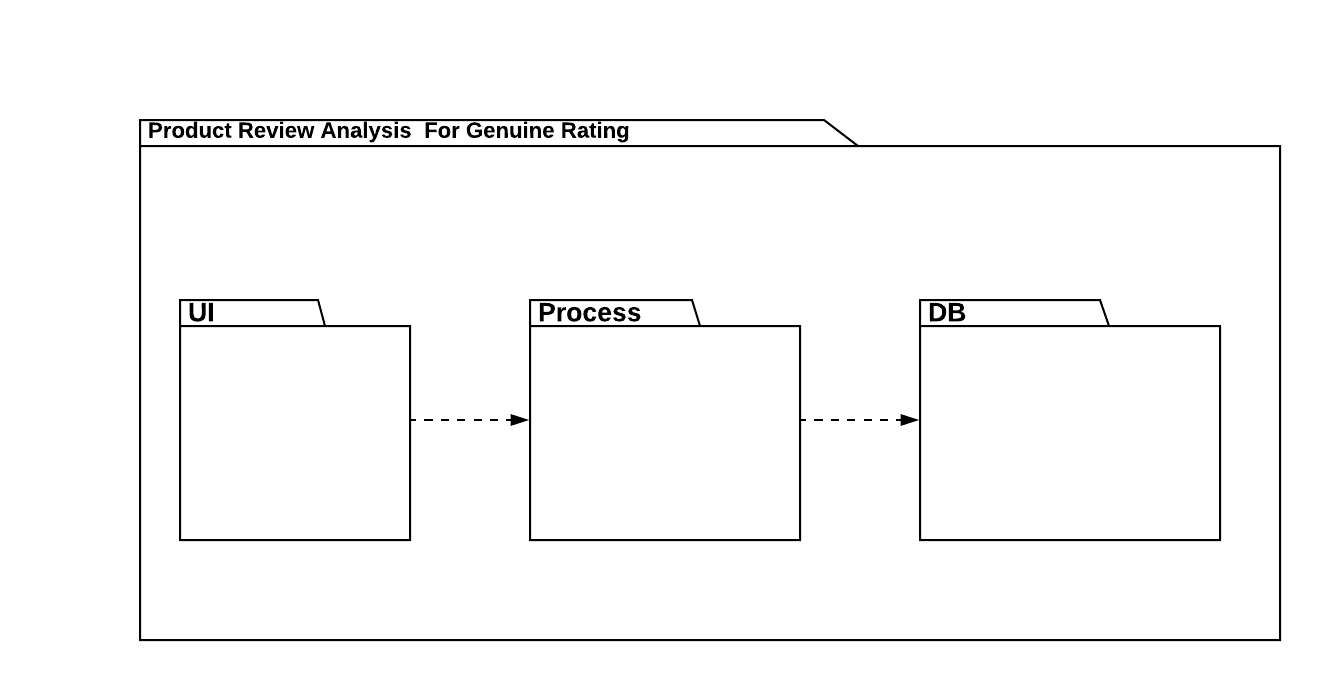
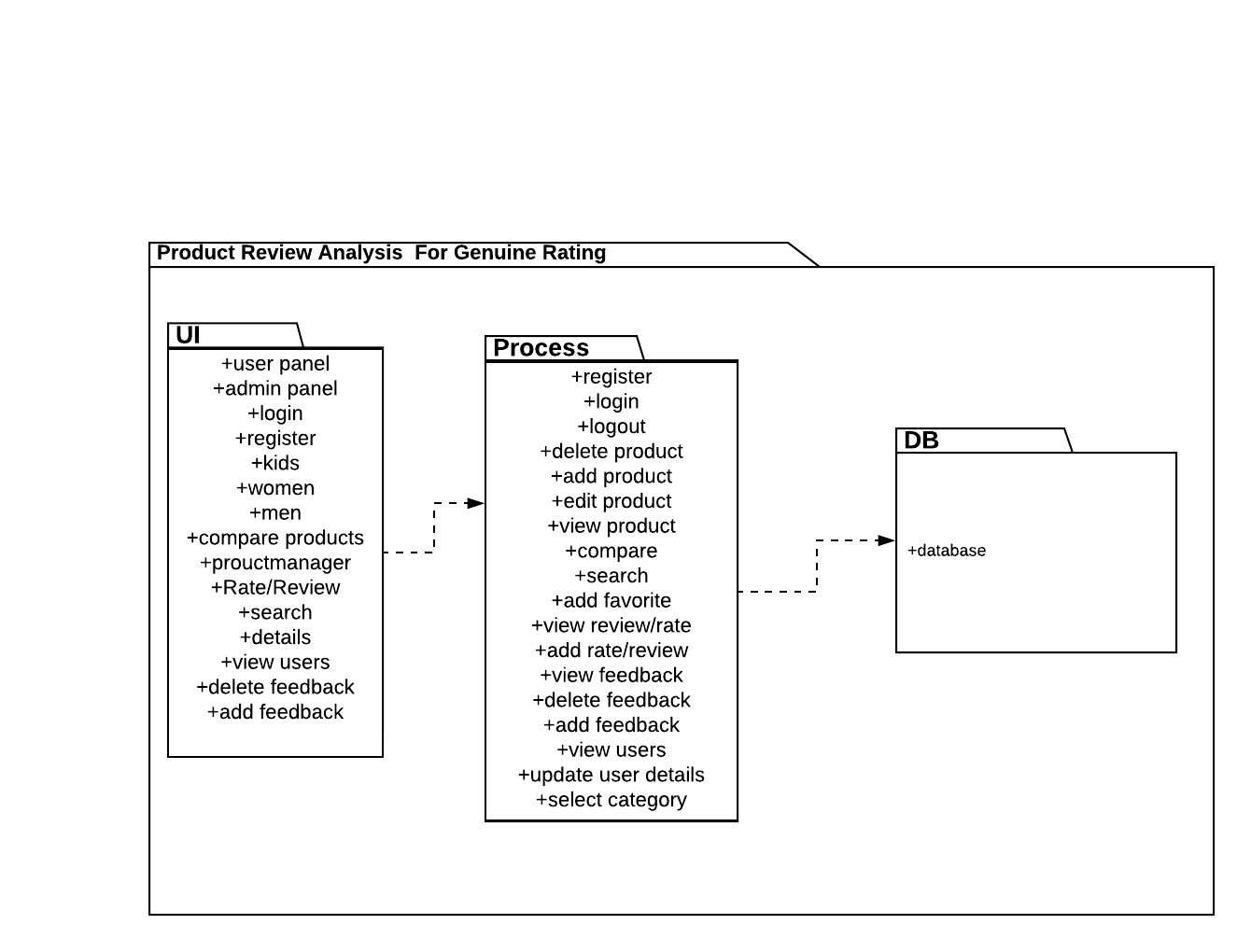
**-**Objects that mediate between boundaries and entities. These serve as the glue between boundary elements and entity elements, implementing the logic required to manage the various elements and their interactions.

***1.4. Forks /Cascades:***

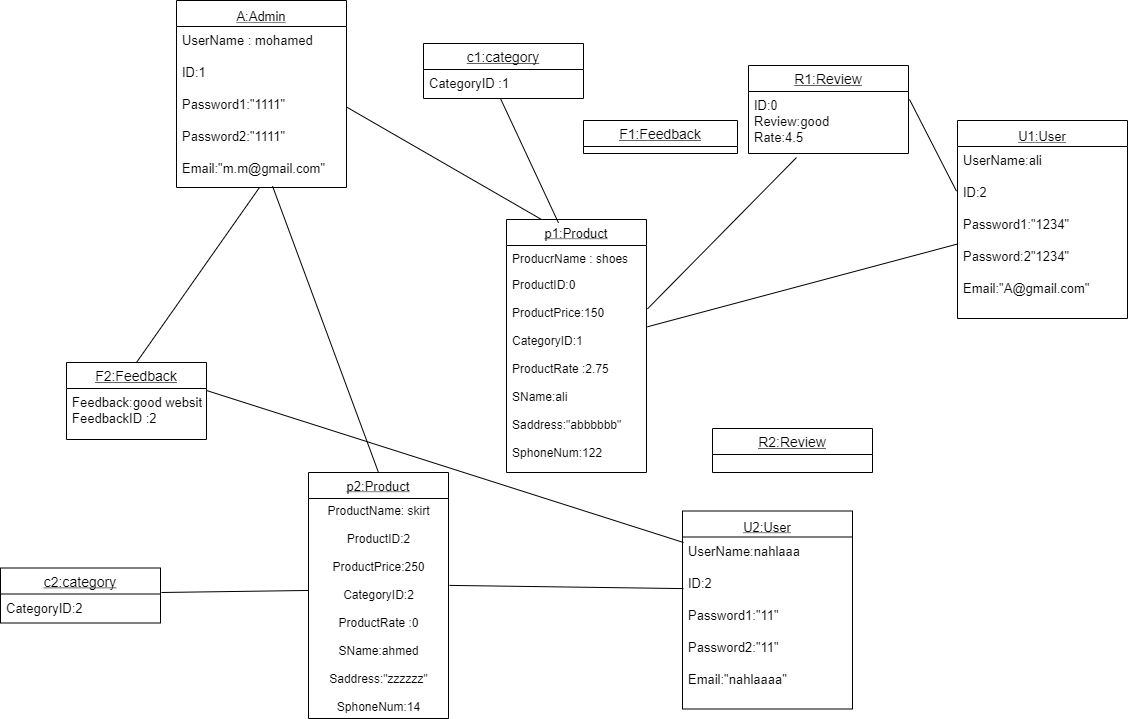
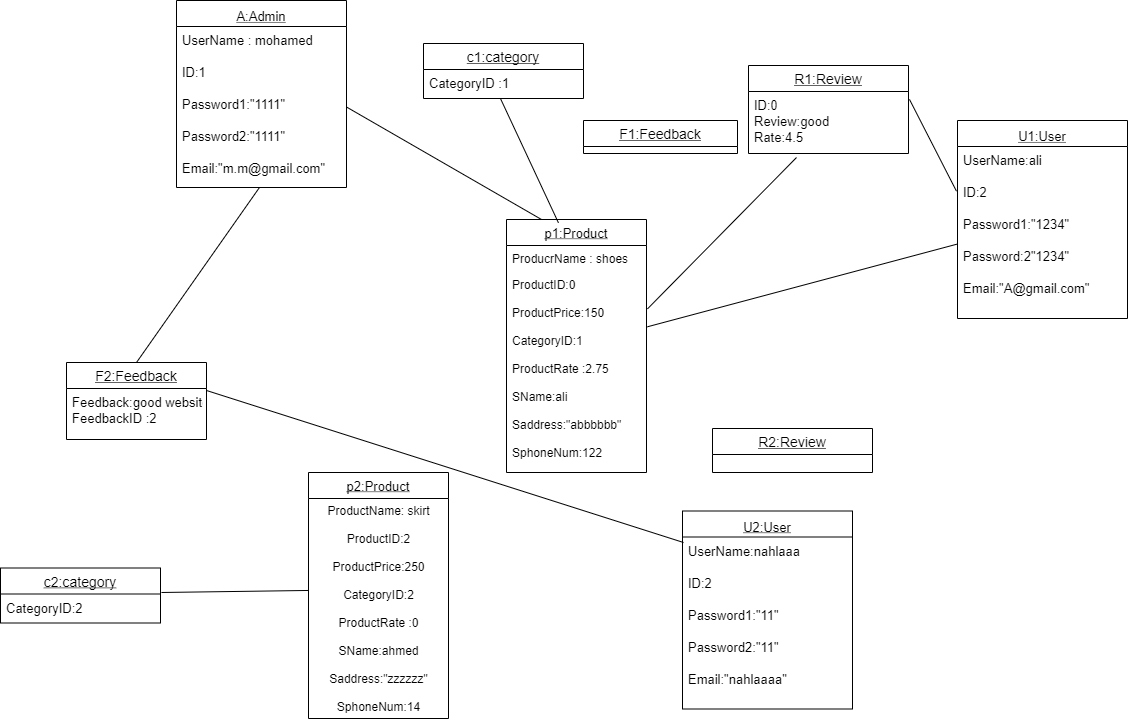
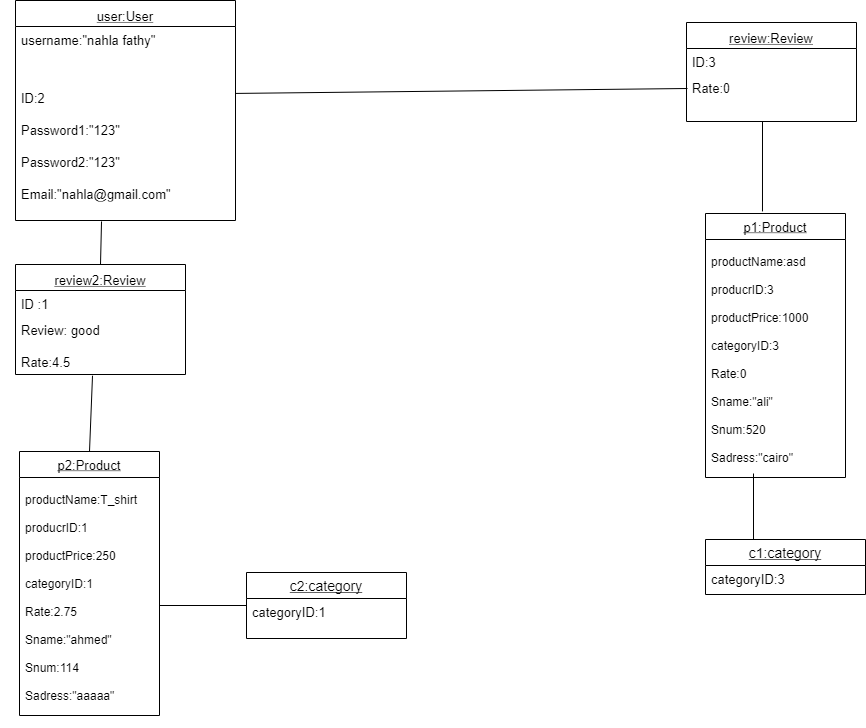
**Cascade is used:**

* + When we need to access and view product in the person (Admin Dashboard) class and we don’t have a **direct association** between Person Class and Product Class, we send message to Admin Class to view product information.
  + Then the Admin Class sends another message to Product Class to view product information (name, id and price and so on.)
  + We used **Cascade** instead of fork; because one of the **advantages** of the cascade is a reasonable position is that Person should need to know only about the existence of Admin, not particular attribute such as Product, its only need to know about admin not Product because there is no direct association or any coupling between each other.
  + But its **disadvantage** is that we need more effort to send message many time instead of send a direct message to a specific class.

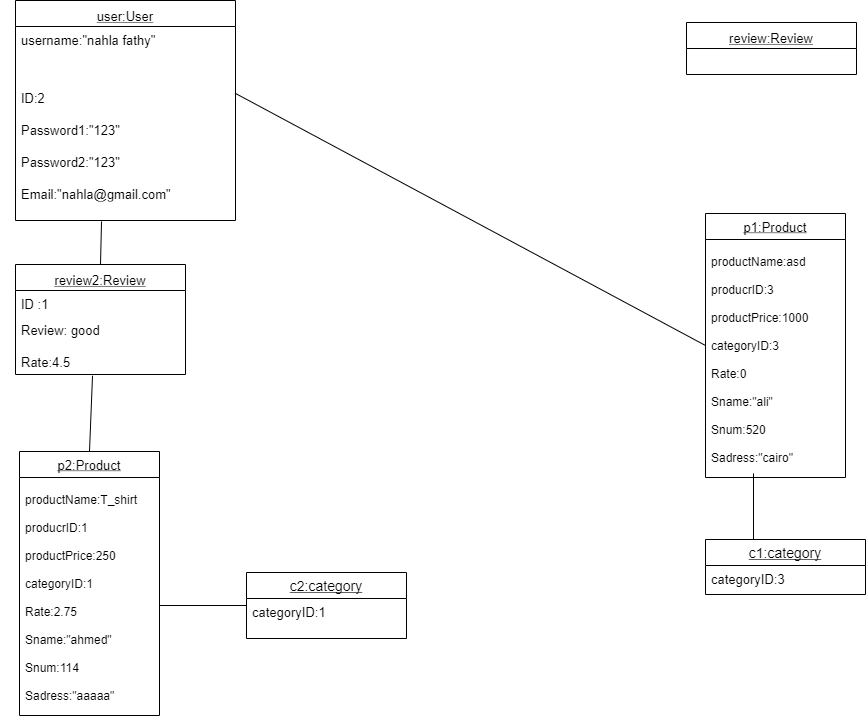
***1.5. Package Diagram:***



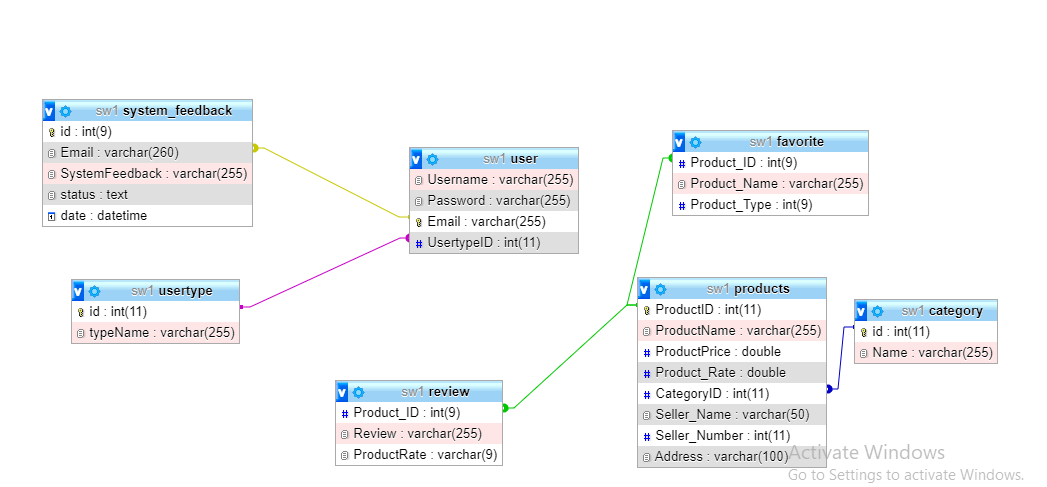
***1.6. Object Diagram:***

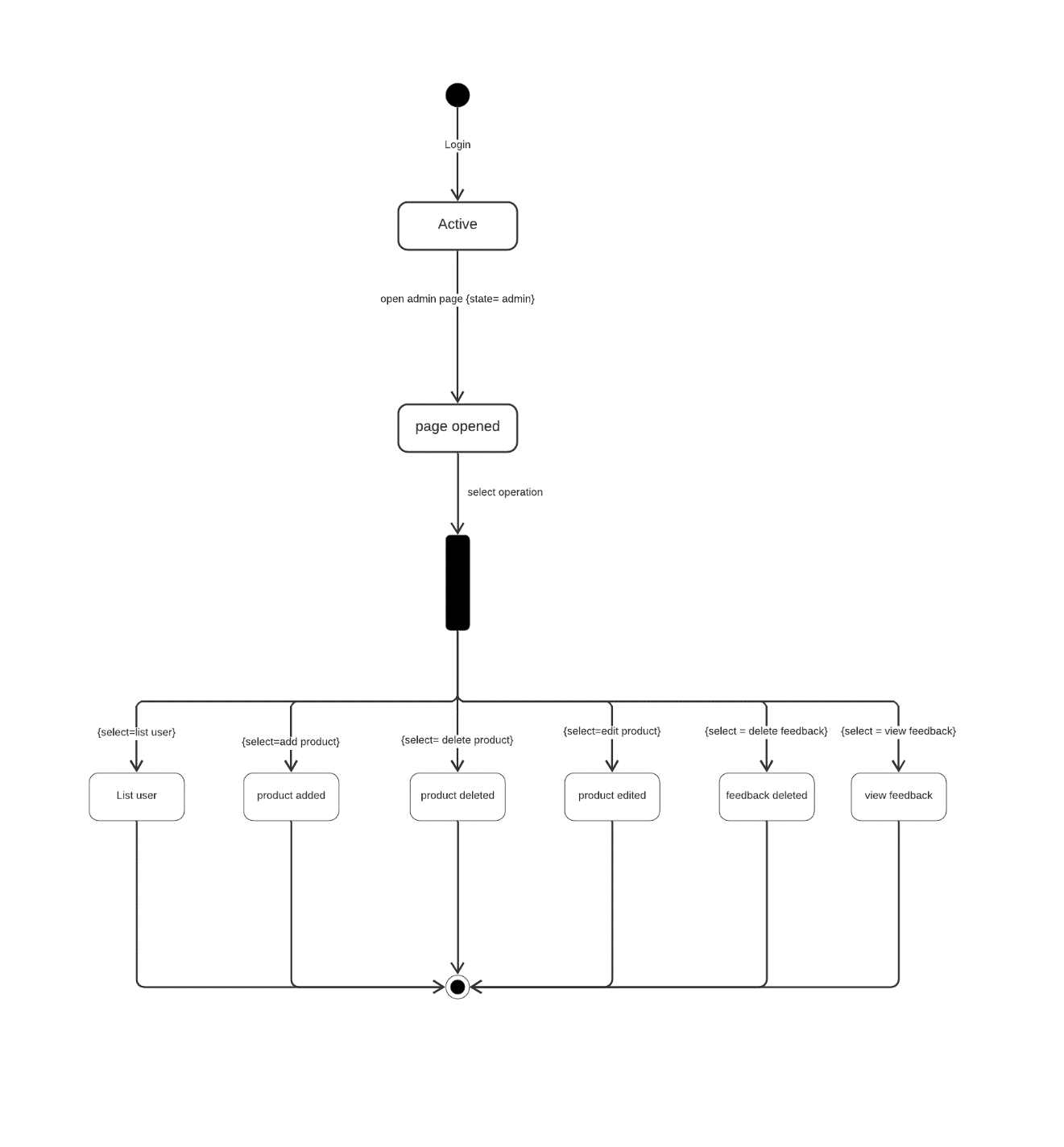
* Pre-Condition:
* Post-Condition:
* Pre-Condition:

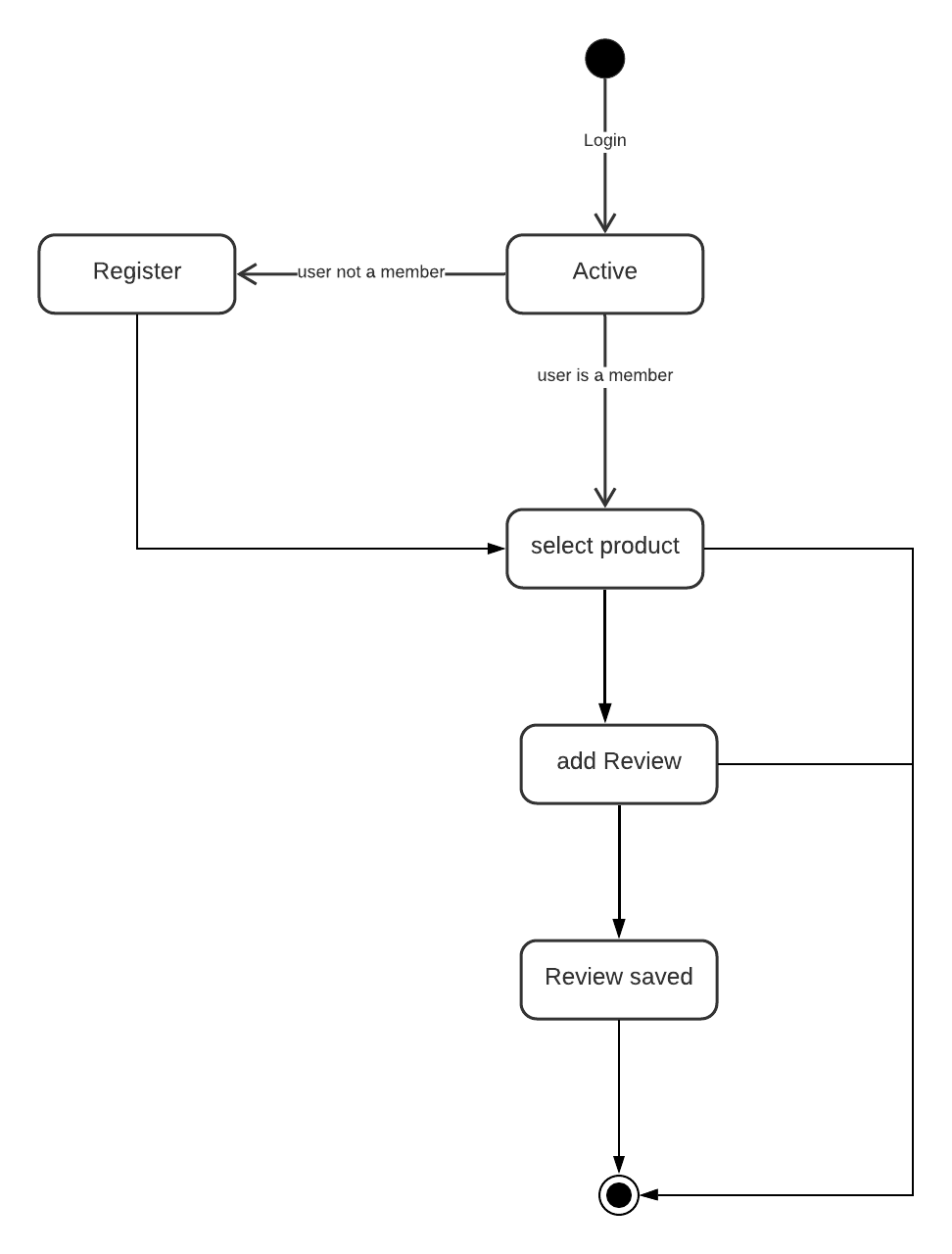
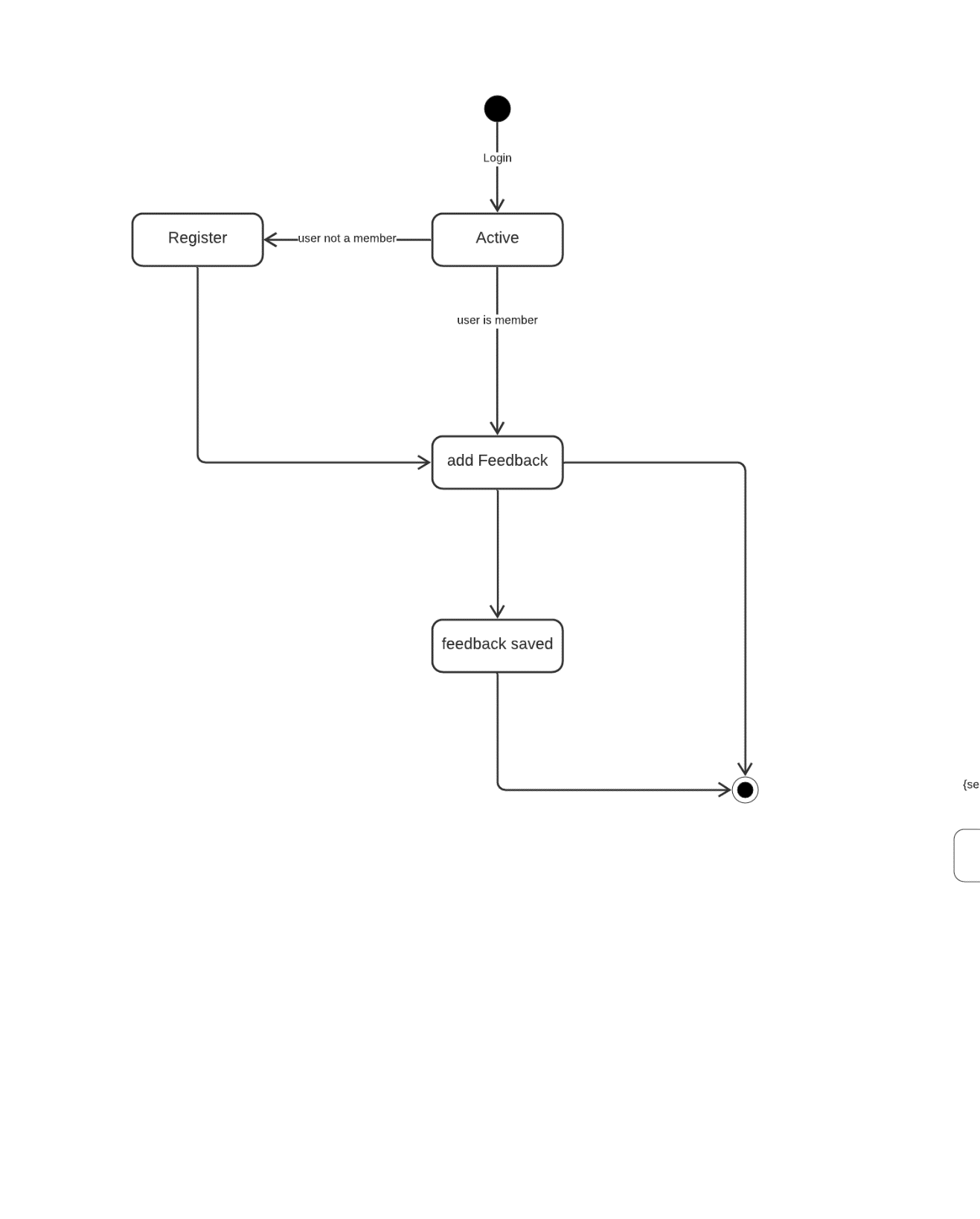
* Post-Condition:



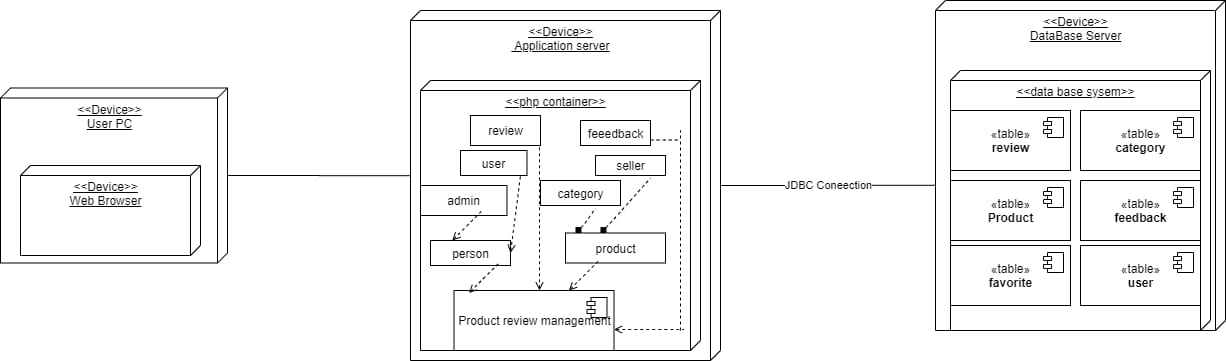
***1.7. Database Specification:***

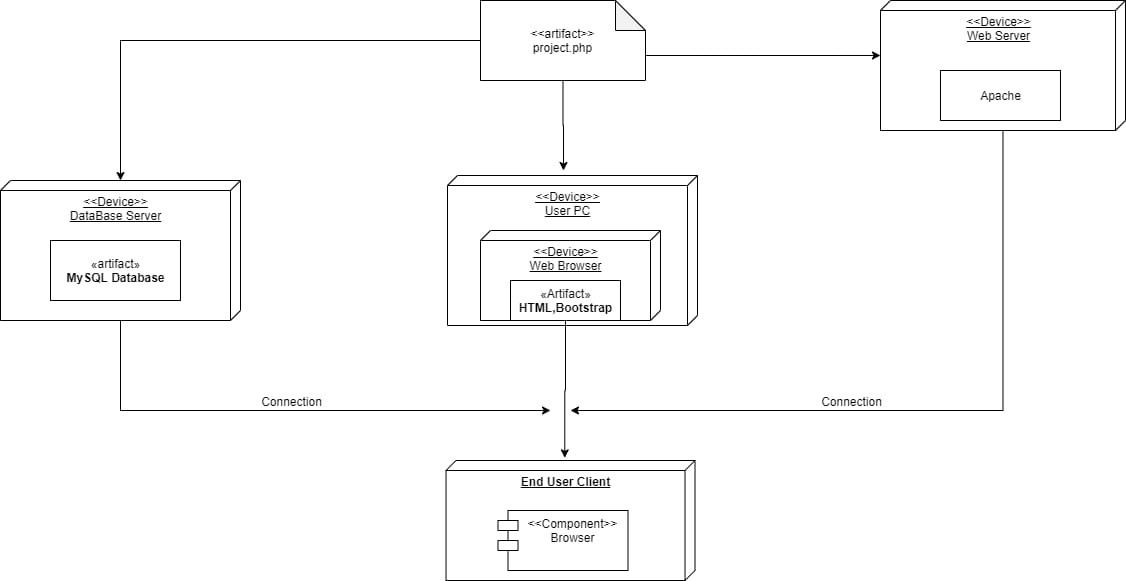
* ERD:

***1.8. State-Machine Diagram:***



***1.9. Deployment Diagram:***





**Part 4:**

1. Complexity & testing:
   1. ***Quality Factors***

**Software Quality Factors that is not independent:**

* **Integrity &efficiency** :
* There is an inverse relationship between efficiency and Integrity
* Growing integrity within a system must affect efficiency because if we increase integrity in our system the system will be more secure. This need to access certain data & an authentication server to check a user's identity, and it means that network traffic needs to be encrypted and decrypted .so efficiency is likely to be minimized
* **Extensibility and Flexibility** :
* Extensibility is the ability of software to add functionality without a harmful system.
* Flexibility is the ability of software to modify and remove implementation without damaging the system.
* Those functionality changes may occur according to changing requirements or a commitment if the development process necessary to happen over time.  
  This is one of the most important properties of quality software; because of changing is an inescapable thing in Software
* **Usability & Portability**:
* Many of the features of the Apple Macintosh that contribute to its reputation of usability are built into its operating system.
* Applications that take advantage of these features are less portable to other systems, such as Windows or Linux.
  1. ***LOC/CCM***

|  |  |  |
| --- | --- | --- |
| **Function name** | **LOC** | **CCM** |
| View user | 4 | 1 |
| Upload files | 19 | 4 |
| Generate files | 10 | 1 |
| fetch all | 5 | 1 |
| Perform query | 9 | 3 |
| View men | 30 | 5 |
| View women | 30 | 5 |
| View kids | 30 | 5 |
| Connection | 10 | 2 |
| Add feedback | 23 | 4 |
| Delete feedback | 22 | 5 |
| Get password | 3 | 1 |
| Login | 22 | 5 |
| Register | 31 | 5 |
| Add favorite | 24 | 4 |
| View user | 18 | 4 |
| View product | 28 | 5 |
| View feedback | 20 | 4 |
| Update details | 15 | 3 |
| View review | 20 | 4 |
| Edit | 48 | 8 |
| Add product | 47 | 4 |
| Delete product | 25 | 4 |
| Search product | 31 | 4 |
| Compare product | 34 | 8 |

***1.3. Complexity Metrics***

1. **WMC (Weighted Methods per Class):**

WMC is the sum of CCM for each method in the class.

1. **Person Class:**
2. CCM login = 5
3. CCM register = 5
4. CCM view product = 4
5. CCM view User = 3
6. CCM view feedback = 3
7. CCM update Details = 2
8. CCM viewRR = 3
9. CCM add favorite = 3

WMC of Person Class = 5 + 5 + 4 + 3 + 3 + 2 + 3 + 3 = 28

1. **Admin Class:**
2. CCM admin view users = 1
3. CCM generate files = 1
4. CCM upload files = 4
5. CCM get Instance = 1

WMC of Admin Class = 1 + 1 + 4 + 1 = 7

1. **User Class:**
2. CCM user update Details = 1
3. CCM user register = 1

WMC of User class = 1 + 1 = 2

1. **Product Class:**
2. CCM edit product = 7
3. CCM Add product = 4
4. CCM delete product = 3
5. CCM search product = 3
6. CCM compare product = 7

WMC of product class = 7 + 4 + 3 + 3 + 7 = 24

1. **Feedback class:**
2. CCM Add feedback = 3
3. CCM delete feedback = 3

WMC of feedback class = 3 + 3 = 6

1. **Review class:**
2. CCM add rate = 1
3. CCM calculate average rate = 1

WMC of review class = 1 + 1 = 2

1. **Category class:**
2. CCM view women = 4
3. CCM view men = 4
4. CCM view kids = 4

WMC of category class = 4 + 4 + 4 = 12

1. **Alter class:**
2. CCM fetchAll = 1
3. CCM perform Query = 2

WMC of alter class = 1 + 2 = 3

1. **Popup class:**

All function CCM = 1, there is 17 functions

WMC of popup class = 17

1. **Database class:**
2. CCM connection = 2

WMC of database class = 2

1. **DIT (Depth of inheritance tree):**
   1. DIT of Person class = 0
   2. DIT of user class = 0
   3. DIT of Admin class = 0
   4. DIT of Product class = 0
   5. DIT of category class = 0
   6. DIT of feedback class = 0
   7. DIT of Review class = 0
   8. DIT of database class = 0
   9. DIT of alter class = 0
   10. DIT of popup = 0
2. **NOC (Number of children):**
   1. NOC of Person class = 0
   2. NOC of user class = 0
   3. NOC of Admin class = 0
   4. NOC of Product class = 0
   5. NOC of category class = 0
   6. NOC of feedback class = 0
   7. NOC of Review class = 0
   8. NOC of database class = 0
   9. NOC of alter class = 0
   10. NOC of popup = 0
3. **CBO (coupling between objects):**
4. CBO of Person class = 1
5. CBO of user class = 1
6. CBO of Admin class = 2
7. CBO of Product class = 1
8. CBO of category class = 0
9. CBO of feedback class = 1
10. CBO of Review class = 1
11. CBO of database class = 0
12. CBO of alter class = 0
13. CBO of popup = 0
14. **RFC (Response of class):**
    1. RFC of Person class = 8 + 1 = 9
    2. RFC of user class = 2 + 1 = 3
    3. RFC of Admin class = 4 + 2 = 6
    4. RFC of Product class = 5 + 1 = 6
    5. RFC of category class = 3 + 0 = 3
    6. RFC of feedback class = 2 + 1 = 3
    7. RFC of Review class = 2 + 1 = 3
    8. RFC of database class = 1 + 0 = 1
    9. RFC of alter class = 2 + 0 = 2
    10. RFC of popup = 17 + 0 = 17
15. **LCOM (Lack of Cohesion of Methods):**

LCOM equation = 1 – (sum(MF)/M\*F)

Where:

* F is the number of Class Instance fields. • M is the number of methods in the class (including static and instance methods, constructors, getters / setters properties, add / remove methods events).

• MF is the number of class methods that access a given instance field.

• Sum(MF) represents the sum of MF over all class instance fields.

* 1. **Person class:**

A particular instance field = email.

M = 17, F=5, sum(MF)= 3 + 5 = 8

LCOM = 1- (8/17\*5) = 0.905

* 1. **User class:**

M = 2, F= 0, sum(MF) = 0 + 0 = 0

LCOM = 1- (0/2\*0) = 1

* 1. **Admin class:**

A particular instance field = instance.

M = 5, F = 1, sum(MF) = 1 + 1 = 2

LCOM = 1 – (2 / 5\*1) = 0.6

* 1. **Product class:**

A particular instance field = product id.

M = 24, F = 10, sum(MF) = 3 + 10 = 13

LCOM = 1 – (13/ 24\*10) = 0.945

* 1. **Category class:**

A particular instance field = id.

M = 5, F = 1, sum(MF) = 3 + 1 = 4

LCOM = 1 – (4 / 5\*1) = 0.2

* 1. **Feedback class:**

A particular instance field = id.

M = 8, F = 3, sum(MF) = 1 + 3 = 4

LCOM = 1 – (4 / 3\*8) = 0.833

* 1. **Review class:**

A particular instance field = id.

M = 8, F = 3, sum(MF) = 1 + 3 = 4

LCOM = 1 – (4 / 3\*8) = 0.833

* 1. **Database class:**

M = 1, F= 0, sum(MF) = 0 + 0 = 0

LCOM = 1 – (0/ 1\*0) = 1

* 1. **Alter class:**

M = 2, F=0, sum(MF) = 0 + 0 = 0

LCOM = 1 – (0 / 0\*2) = 1

* 1. **Popup class:**

M = 17, F = 0, sum(MF) = 0 + 0 = 0

LCOM = 1 – (0 / 17 \* 0) = 1

***1.4. White Box Testing***

**1- Search Product:**

* The function is based on searching for products in database by its name

**Case1:**

* When user insert in the searching box “skirt” the system check if skirt is found or not in database and skirt is found so the system view the information of the product named skirt

**Case2:**

* When user insert in the searching box “cars” the system check if cars is found or not in database and cars are not found so the system display error message saying “Product not exist”

**2-AddFeedback:**

* The function is based on adding feedback about the system with user email only once

**Case1:**

* When user write a feedback for example (the system is good) with email ([Aliahmed@hotmail.com](mailto:Aliahmed@hotmail.com)) and this email is used before in writing feedback so the system will display error message saying “Feedback already exist”

**Case2:**

* When user write a feedback for example (the system is good) with email ([menna@hotmail.com](mailto:menna@hotmail.com)) and this email is not used before in writing feedback so the system will add the feedback to the system

**3- Delete feedback:**

* The function is based on deleting feedback by entering the email used for adding it

**Case1:**

* When user insert the email ([eslam@hotmail.com](mailto:eslam@hotmail.com)) and this email is not used before in writing feedback so the system will display error message saying “Email not found”

**Case2:**

* When user the email ([aliahmed@hotmail.com](mailto:Aliahmed@hotmail.com)) and this email is not used before in writing feedback so the system will delete this email from the database and will display “Feedback deleted”

**4- Delete Product:**

* The function is based on deleting product by entering the product id

**Case1:**

* When user insert the product id “1” and this id exist in the database then the system will delete the product which has the id “1” and the system will display “Product deleted successfully”

**Case2:**

* When user insert the product id “20” and this id is not exist in the database then the system will display error message saying “Product don’t exist”

**5-Login:**

* Function based on log in using email and password

**Case1:**

* When user enter email “aliahmed@hotmail.com” and password “123456” and the email and password are valid in the data base then the system will take the user to the home page

**Case2:**

* When user enter email “Mohamed@gmail.com” and password “918237” and email or password or both are invalid so the system will display error message saying “user name or password are invalid”

**6-Compare Products**

* The function based on comparing 2 products by writing their ids

**Case1:**

* The user enter id1 “1” and id2 “5” and the two ids are exist in the data base then the system will display the information of the two products which have the 2 ids

**Case2:**

* The user enter id1 “12” and id2 “33” and one or both ids are not found in the database the system will display error message saying “one or both ids are not found”

**7-Add favorite**

* The function is based on adding products to favorite list by writing the details of the product in favorite form

**Case1:**

* If the user enter product id “5” and product name “skirt” and product type “women” and the product id is not found in the favorite table then the data that the user enter will be stored in the favorite table and the system will display message saying “product added successfully”

**Case2:**

* If the user enter product id “1” and product name “shirt” and product type “men” and the product id is found in the favorite table then the system will display error message saying “product already exist in the favorite table”

***1.5. Black Box Testing***

* **Using boundary testing**

1. **Register Function :**

Test Cases:

**Username Field 🡪**

|  |  |  |
| --- | --- | --- |
| **Boundary Value Analysis** | | |
| **Invalid**  **(min -1)** | **Valid**  **(min, +min, -max, max)** | **Invalid**  **(max+1)** |
| 2 characters | 3, 4, ...,28,29 characters | 30 characters |

* User name text field can’t accept value less than 3 characters
* Error message displayed to enter a valid value

**Password Field 🡪**

|  |  |  |
| --- | --- | --- |
| **Boundary Value Analysis** | | |
| **Invalid**  **(min -1)** | **Valid**  **(min, +min, -max, max)** | **Invalid**  **(max+1)** |
| 7 characters | 8,9, .. ,19,20 characters | 21 characters |

* Password field can’t accept less than 8 characters
* Error message is displayed to enter valid value
* By entering value ranges from 8 to 20 characters
* The entering value will be stored in the database then message displayed “successfully registered”

1. **Login Function :**

**Must exist in the database:**

Test Cases:

**E-Mail Field 🡪**

|  |  |  |
| --- | --- | --- |
| **Boundary Value Analysis** | | |
| **Invalid**  **(min -1)** | **Valid**  **(min, +min, -max, max)** | **Invalid**  **(max+1)** |
| 6 characters | 7,8, .. ,17,18 characters | 19 characters |

* Email field can’t accept value less than 6 characters
* Error message displayed to enter a valid data

**Password Field 🡪**

|  |  |  |
| --- | --- | --- |
| **Boundary Value Analysis** | | |
| **Invalid**  **(min -1)** | **Valid**  **(min, +min, -max, max)** | **Invalid**  **(max+1)** |
| 7 characters | 8,9, .. ,19,20 characters | 21 characters |

* Password field can’t accept value less than 8 characters
* Error message displayed to enter a valid data
* By entering value ranges from 8 to 20 characters
* The entering value will be stored in the database then message displayed “login successfully”

1. **Add Product Function :**

Test Cases:

**Product Name Field 🡪**

|  |  |  |
| --- | --- | --- |
| **Boundary Value Analysis** | | |
| **Invalid**  **(min -1)** | **Valid**  **(min, +min, -max, max)** | **Invalid**  **(max+1)** |
| 2 characters | 3,4, .. ,28,29 characters | 30 characters |

* Product name field can’t accept value less than 3 characters
* Error message displayed to enter a valid data

**Price Field 🡪**

|  |  |  |
| --- | --- | --- |
| **Boundary Value Analysis** | | |
| **Invalid**  **(min -1)** | **Valid**  **(min, +min, -max, max)** | **Invalid**  **(max+1)** |
| 0 cell | 1 .. 4 cells | 5 cells |

* Price field couldn’t be empty
* Error message displayed to enter a valid data

**Category field 🡪**

\*there are 3 categories:

|  |  |  |
| --- | --- | --- |
| **Boundary Value Analysis** | | |
| **Invalid**  **(min -1)** | **Valid**  **(min, +min, -max, max)** | **Invalid**  **(max+1)** |
| 0 | 1.. 3 | 4 |

* Category field couldn’t be empty
* Error message displayed to enter a valid data

1. **Function Add Rate/Review:**

Test Cases:

**Review Field 🡪**

|  |  |  |
| --- | --- | --- |
| **Boundary Value Analysis** | | |
| **Invalid**  **(min -1)** | **Valid**  **(min, +min, -max, max)** | **Invalid**  **(max+1)** |
| 2 characters | 3,4, .. ,24,25 characters | 150 characters |

* Review field can’t accept value less than 3 characters
* Error message displayed to enter a valid data

1. **Function Add Feedback:**

Test Cases:

**Feedback field 🡪**

|  |  |  |
| --- | --- | --- |
| **Boundary Value Analysis** | | |
| **Invalid**  **(min -1)** | **Valid**  **(min, +min, -max, max)** | **Invalid**  **(max+1)** |
| 2 characters | 3,4, .. ,149 characters | 150 characters |

* Feedback field can’t accept value less than 3 characters
* Error message displayed to enter a valid data
* User’s email must be exist in the database

1. **Function Search Products:**

Test Cases:

**Product name field 🡪**

**Must be exit in the database:**

|  |  |  |
| --- | --- | --- |
| **Boundary Value Analysis** | | |
| **Invalid**  **(min -1)** | **Valid**  **(min, +min, -max, max)** | **Invalid**  **(max+1)** |
| 2 characters | 3,4, .. ,28,29 characters | 30 characters |

* Product field can’t accept value less than 3 characters
* Error message displayed to enter a valid data

1. **Function Add to Favorite:**

Test Cases:

**Product Name Field 🡪**

|  |  |  |
| --- | --- | --- |
| **Boundary Value Analysis** | | |
| **Invalid**  **(min -1)** | **Valid**  **(min, +min, -max, max)** | **Invalid**  **(max+1)** |
| 2 characters | 3,4, .. ,28,29 characters | 30 characters |

* Product name field can’t accept value less than 3 characters
* Error message displayed to enter a valid data

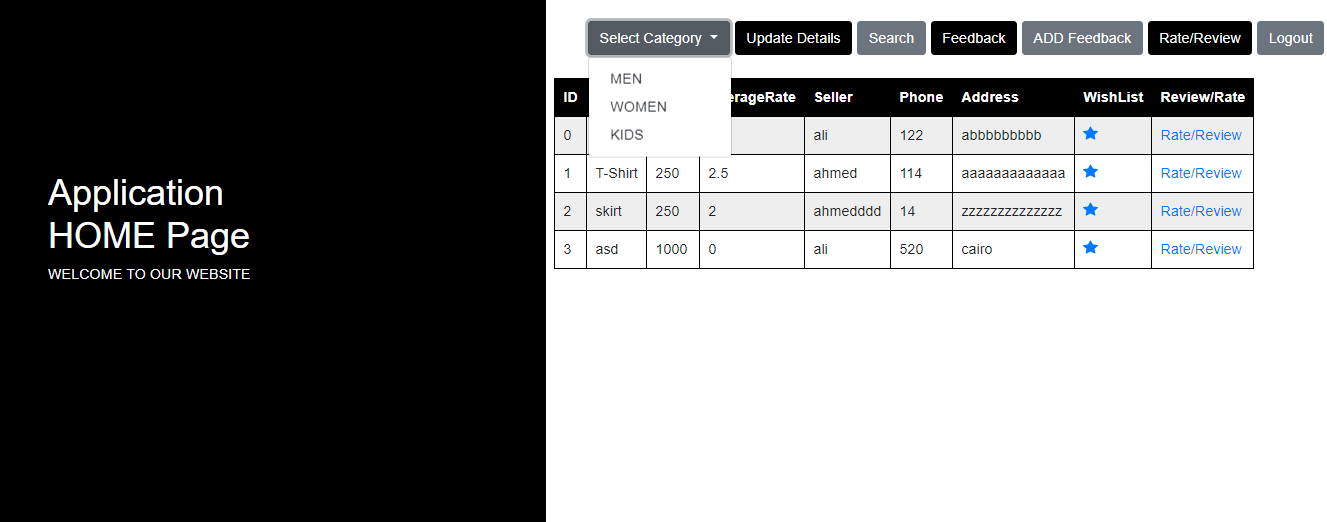
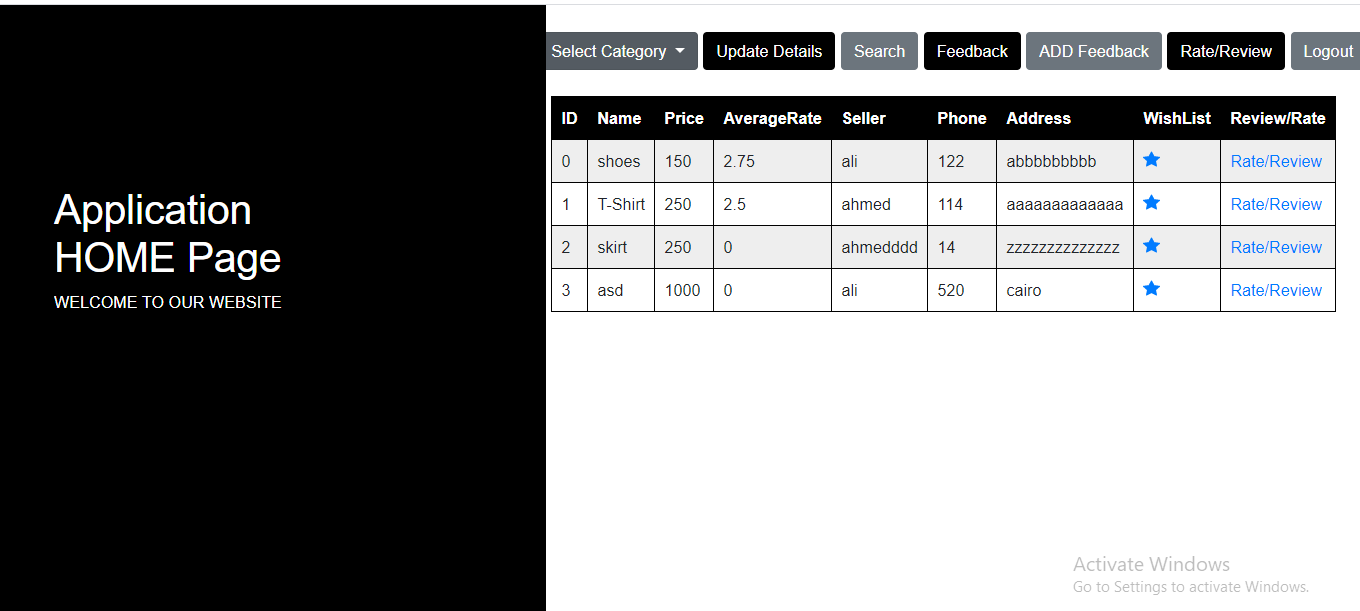
**Product Type Field 🡪**

**\*there are 3 categories**:

|  |  |  |
| --- | --- | --- |
| **Boundary Value Analysis** | | |
| **Invalid**  **(min -1)** | **Valid**  **(min, +min, -max, max)** | **Invalid**  **(max+1)** |
| 0 | 1 .. 3 | 4 |

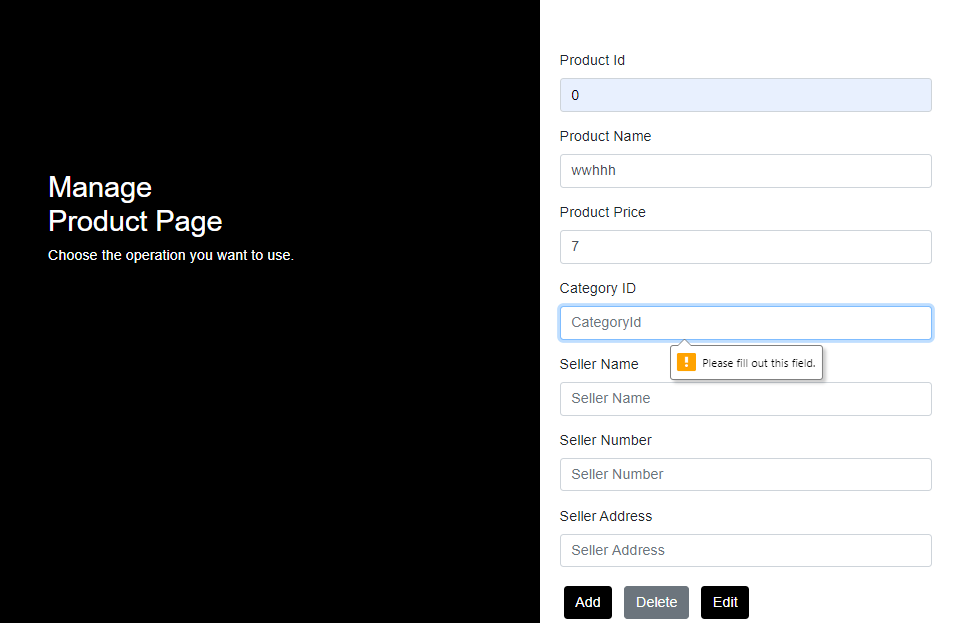
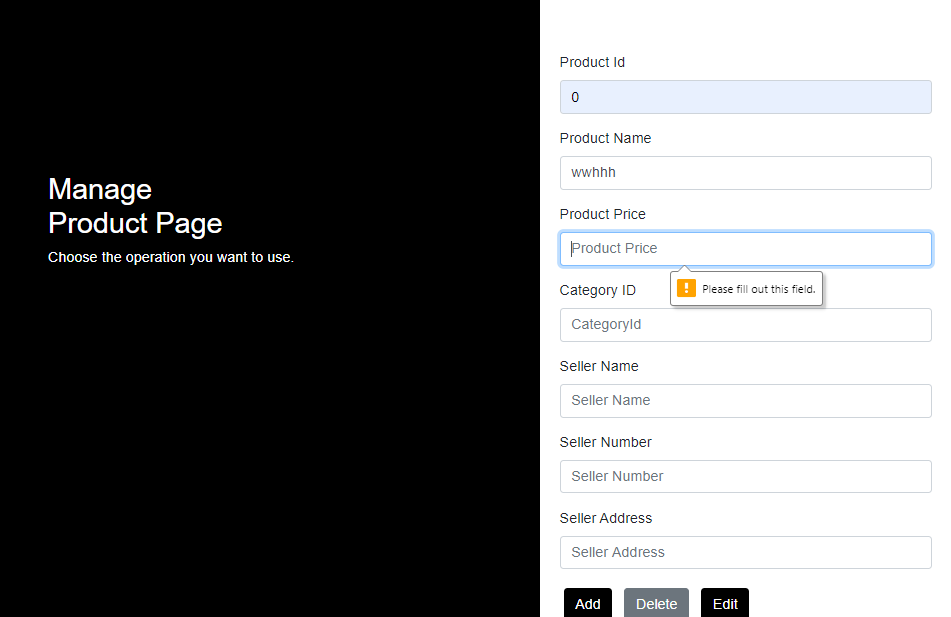
* Product type field couldn’t be empty
* Error message displayed to enter a valid data
  1. ***Heuristic Evaluation***

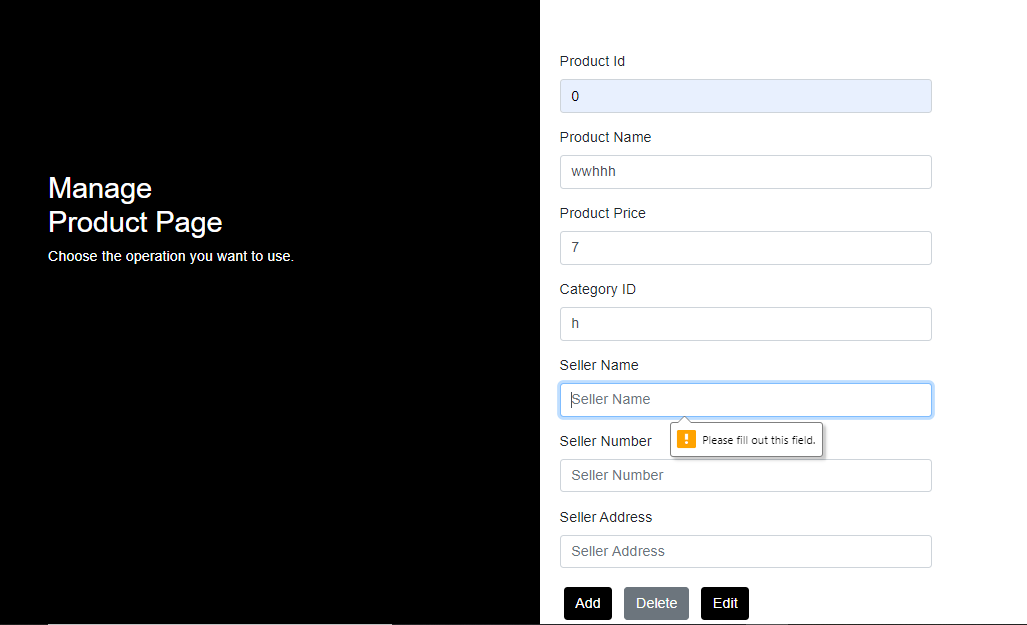
**1-Heuristic Review:**

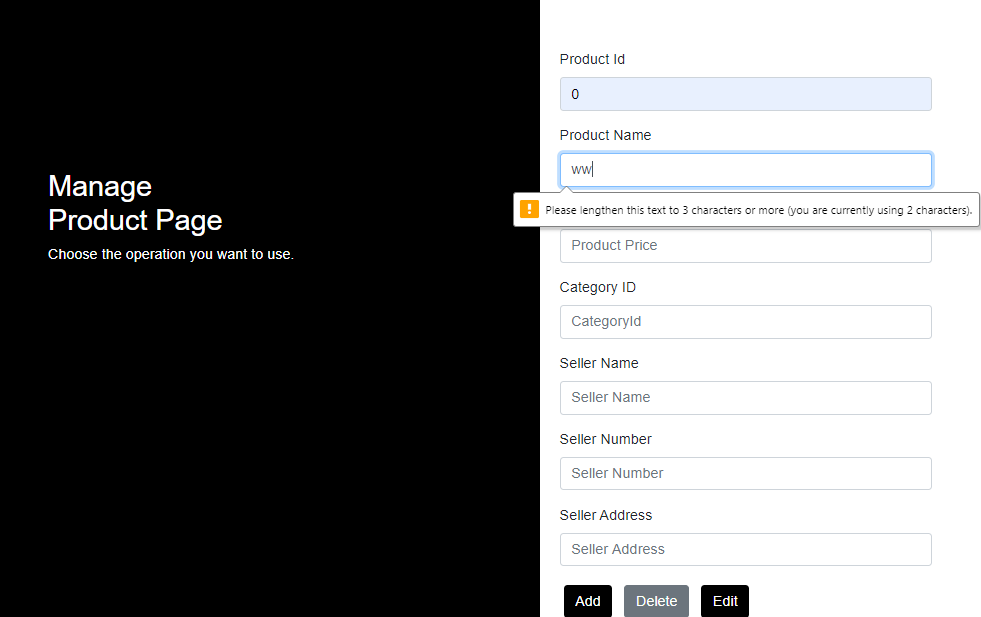
Screen 1: User Panel 

|  |  |  |
| --- | --- | --- |
| **Heuristic** | **Description** | **Potential Solution** |
| **Flexibility** | * + favorite star icon:   it’s clear for users and therefor the beginners they’ll add their favorite icon by clicking on  .   * + review/rate : by clicking on it users will be able to give rate /review to the products   + limited information for products   + user couldn’t buy or contact the seller | * + products should be provided with images ,colors and size   + add chat and contact feature |
| **Consistency standards** | * + favorite star icon: external consistency as people are familiar with this feature while using many sites   + no cart to put the wanted items in   + users cannot contact with sellers | * + add cart icon   + add phone icon |
| **Match between system and real world** | * + system use natural language that anyone can understand(product ,rate ,review) |  |
| **Visibility /user control & freedom** | * + from the displayed menu user are free to decide where to go and know what every tab do |  |

Screen 2:Manage Products

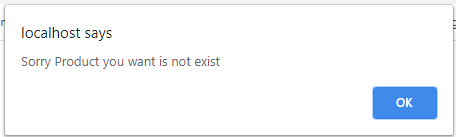
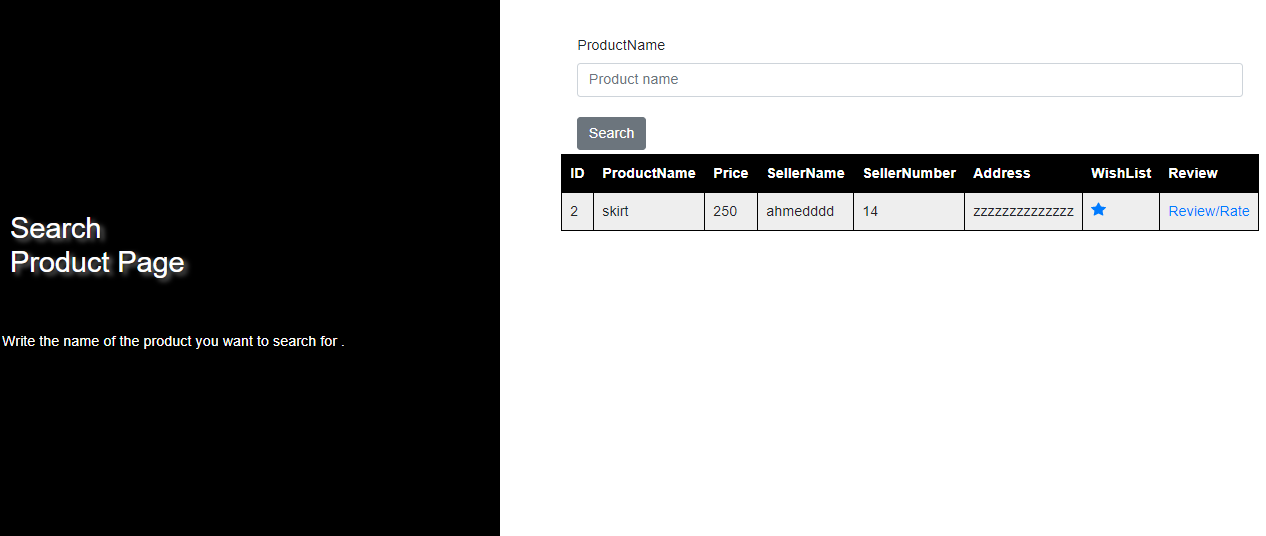


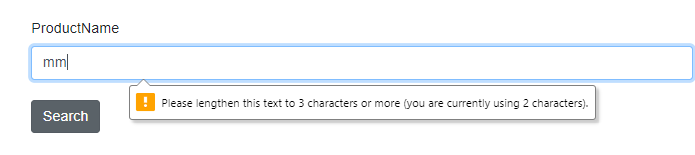




|  |  |  |
| --- | --- | --- |
| **Heuristic** | **Description** | **Potential Solution** |
| **Error prevention** | * + there can be mistakes made while editing and the user want to undo his choice | * + there must be a confirmation message to ensure that the user want to make this action |
| **Help and document / Help Users Recognize, Diagnose, and Recover from Errors** | * + when user forget to fill a field or enters invalid data warning explanation message is displayed |  |

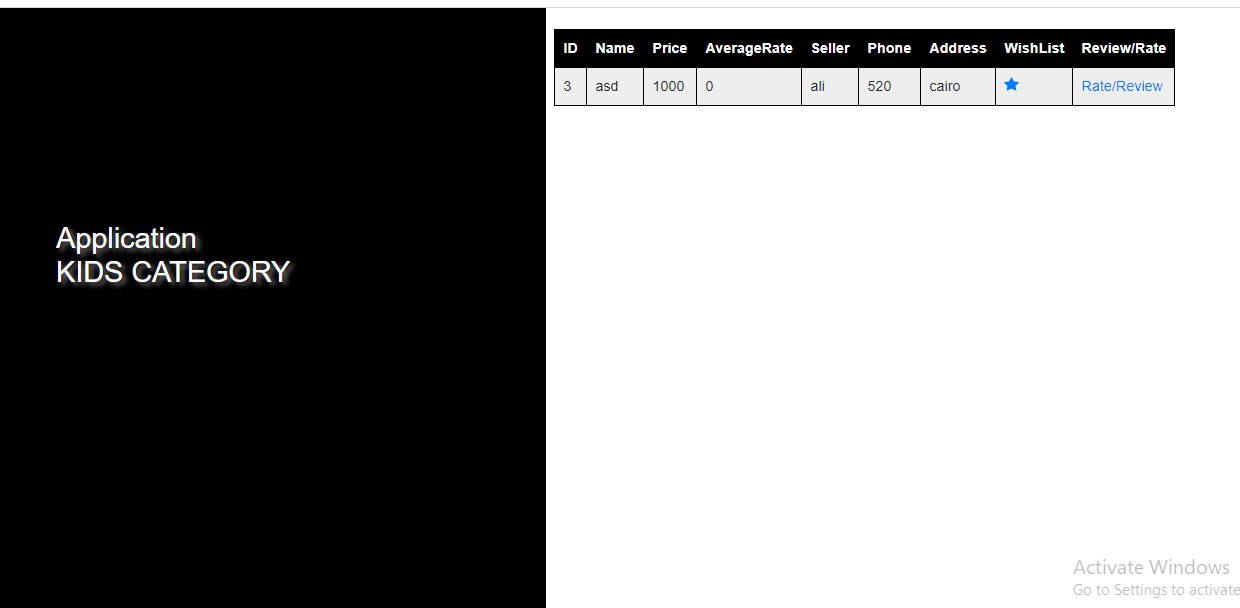
Screen 3: Search





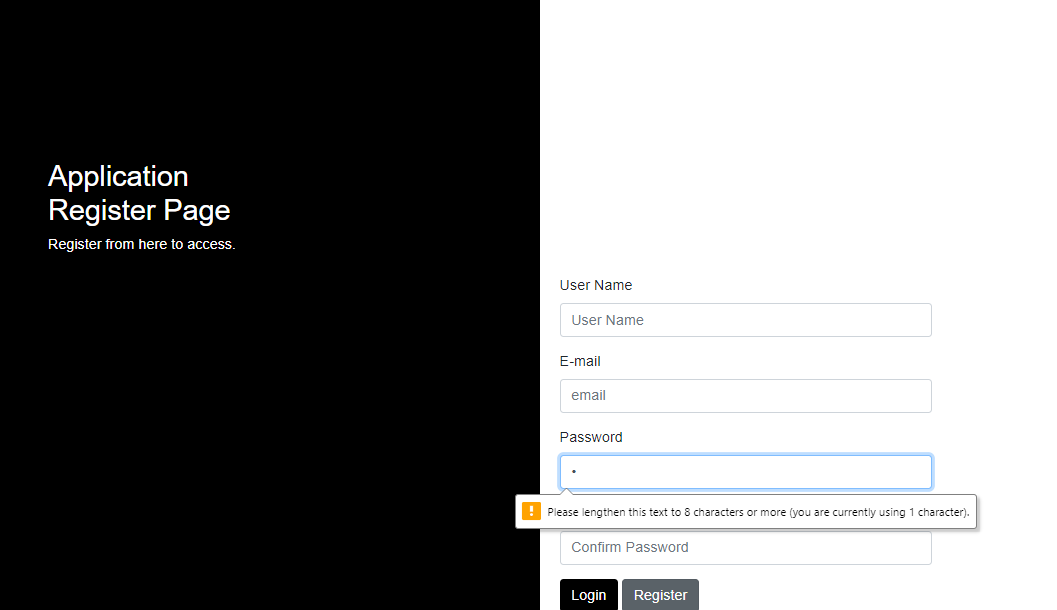
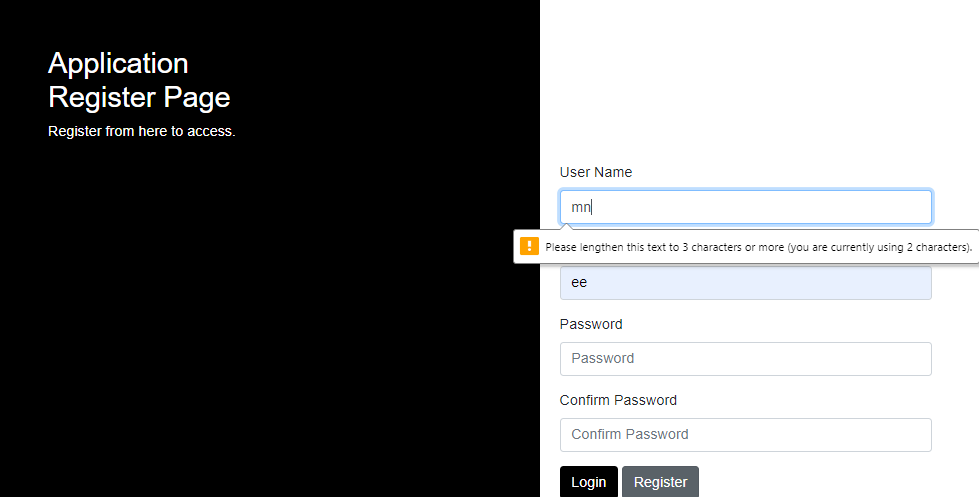
|  |  |  |
| --- | --- | --- |
| **Heuristic** | **Description** | **Potential Solution** |
| **Help Users Recognize, Diagnose, and Recover From Errors** | * + pop up message is displayed while user is searching for unexacting product |  |
| **User control and freedom** | * + User is put in charge to redo all the previous steps until he reach the home page | * + User has to move freely while surfing |
| **Recognition rather than recall** | * + There is no any recommend products while searching | * + Some recommended results(products) must be added to help the user found what he want |
| **Help and document / Help Users Recognize, Diagnose, and Recover from Errors** | * + Warning explanation message is displayed when user enters in valid data |  |

Screen 4: Kids

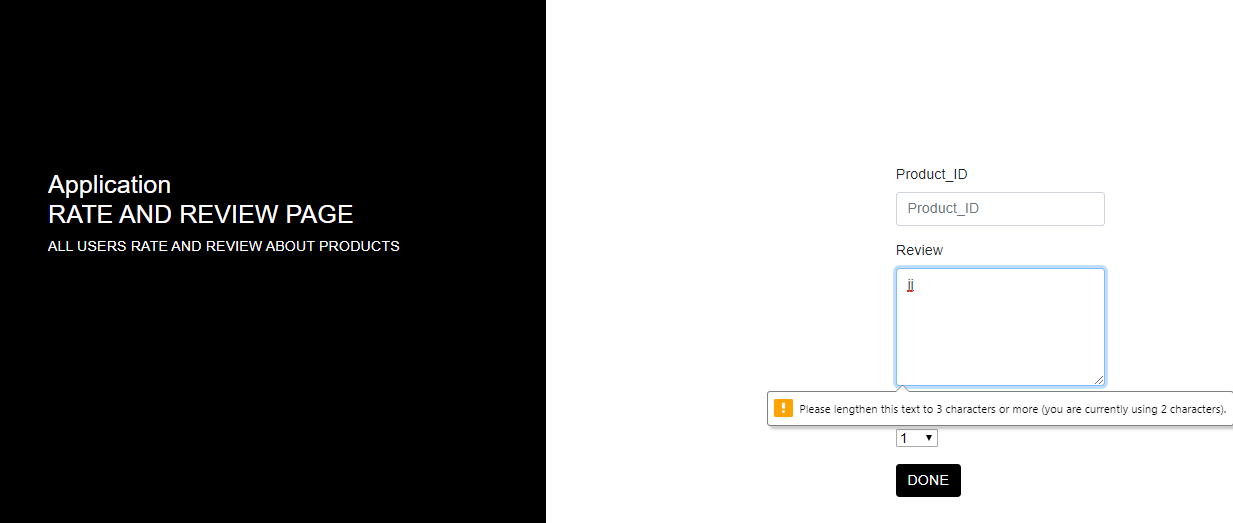


|  |  |  |
| --- | --- | --- |
| **Heuristic** | **Description** | **Potential Solution** |
| **Consistency standards** | External consistency (favorite star icon) |  |
| **Flexibility** | No exit button | * + Add exit button |
| **Visibility** | not enough information for products | * + Products should have images ,colors and size |

screen 5: Register



|  |  |  |
| --- | --- | --- |
| **Heuristic** | **Description** | **Potential Solution** |
| **Help Users Recognize, Diagnose, and Recover From Errors** | * + pop up message is displayed when user registered with used account   + warning message is if the user didn’t fill all the required fields | * + warning message has to displayed |
| **Help and document / Help Users Recognize, Diagnose, and Recover from Errors** | * + warning message is displayed when enters invalid data in password/username field   + no help screen | * + help screen or manual guide should be added to help the beginners how to register an use the system |
| **Flexibility** | * + No exit button | * + Add exit button |

Screen 6: Review /Rate

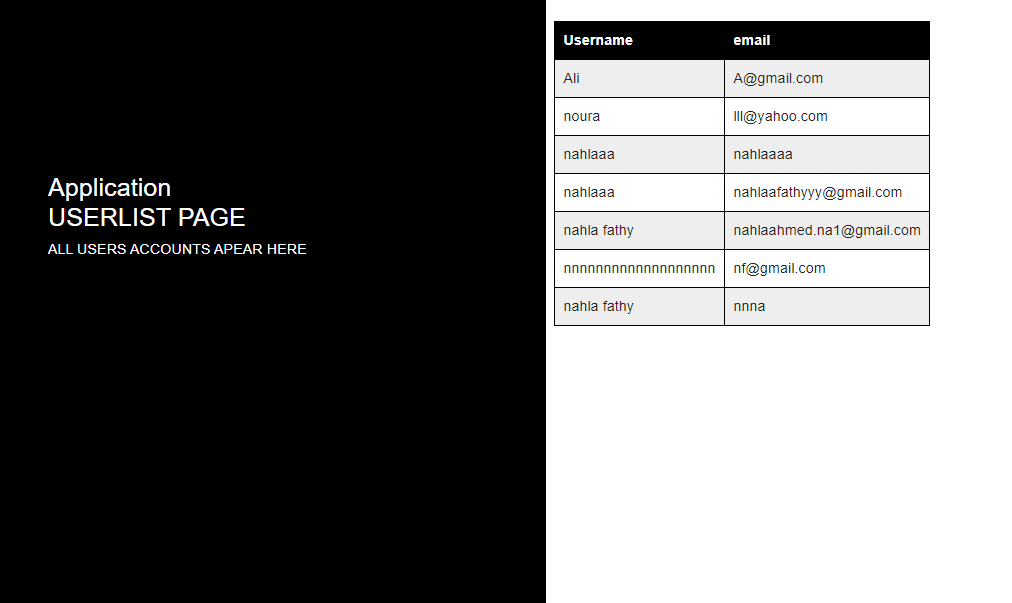


|  |  |  |
| --- | --- | --- |
| **Heuristic** | **Description** | **Potential Solution** |
| **User control &freedom** | * + User has to redo all his previous steps until he reach the home page | * + Default menu has to be added to let user move freely |
| **Flexibility** | * + User might forget product id or enter not existing id | * + User don’t have to insert product id   + Review text field has to be in the same page of the products |
| **Visibility /flexibility** | * + The submitted review is not displayed and user has to return to the home page then click on rate/review | * + Add the submitted review /rate in the same page |
| **Error prevention/user control &freedom** | * + User can’t delete or edit his review | * + Add edit / delete button beside the submitted review |
| **Help and document / Help Users Recognize, Diagnose, and Recover from Errors** | * + Warning explanation message will be displayed when user enters invalid data |  |

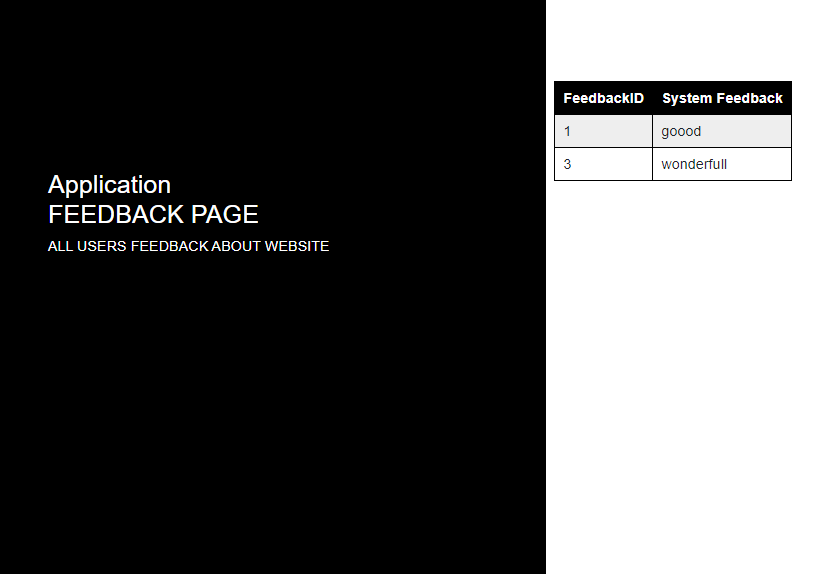
Screen 7: Upload Files



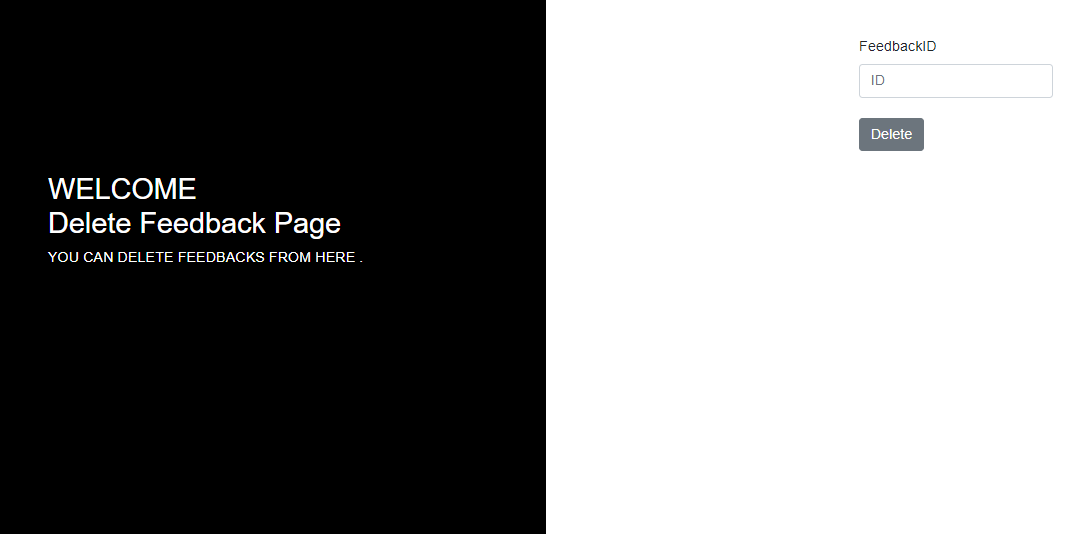
|  |  |  |
| --- | --- | --- |
| **Heuristic** | **Description** | **Potential Solution** |
| **Visibility /Help & Document** | * + Limited information | * + Manual guide should be added |
| **Error Prevention** | * + User cannot undo his action | * + Confirmation message   + Edit button   + Delete button |

Screen 8: View Users 

|  |  |  |
| --- | --- | --- |
| **Heuristic** | **Description** | **Potential Solution** |
| **Visibility /Flexibility** | * + Limited information   + Admin can’t remove users   + No exit button | * + Delete button   + Exit button |

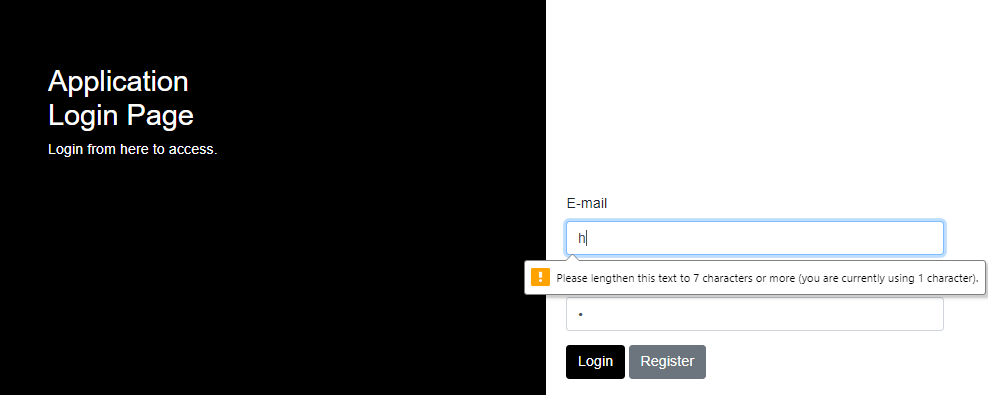
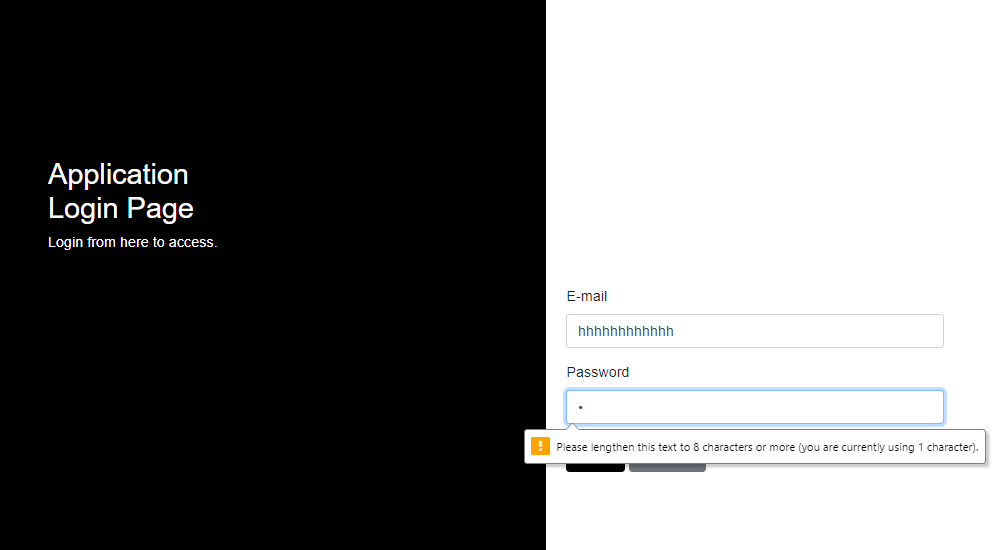
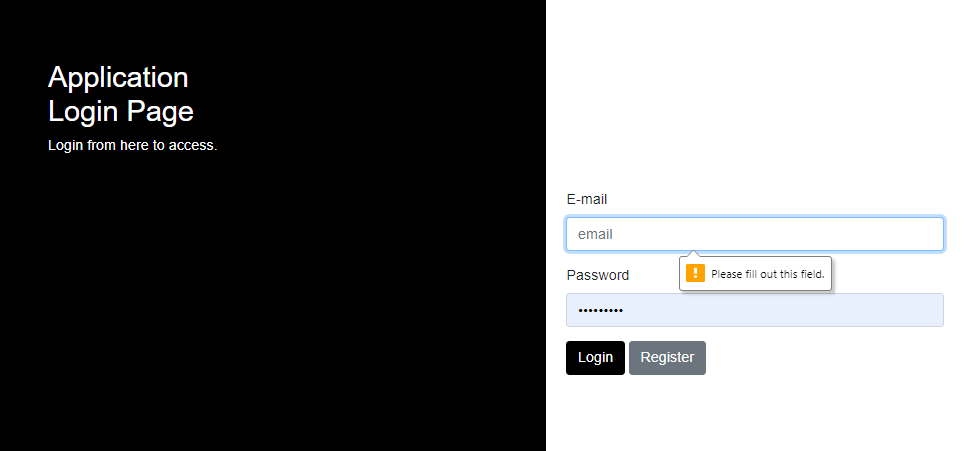
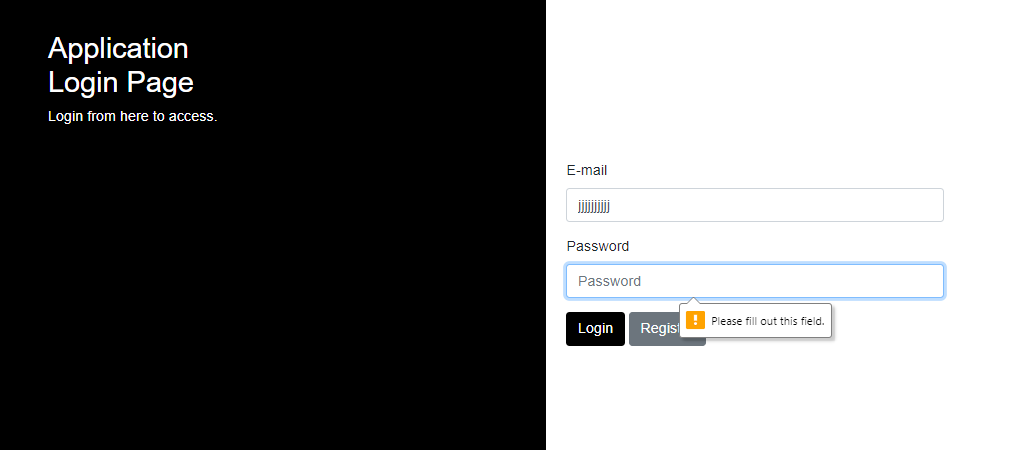
Screen 9: View Feedback

|  |  |  |
| --- | --- | --- |
| **Heuristic** | **Description** | **Potential Solution** |
| **Visibility /Flexibility/User Control &Freedom** | * + Limited information   + No exit button   + User can’t edit, delete his feedback | * + Delete button   + Exit button   + Edit button |

Screen 10: Delete Feedback

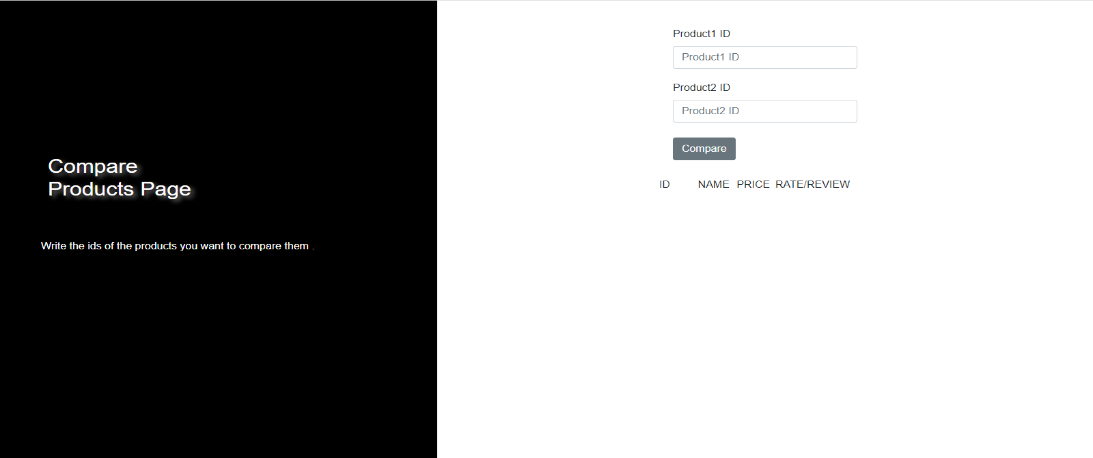
|  |  |  |
| --- | --- | --- |
| **Heuristic** | **Description** | **Potential Solution** |
| **Error prevention/flexibility** | * + User can’t undo his action | * + There must be a confirmation message to ensure that the user want to make this action |
| **Flexibility** | * + No exit button   + User might forget feedback id | * + Add exit button   + Feedback table has to be displayed |

screen11: Login page



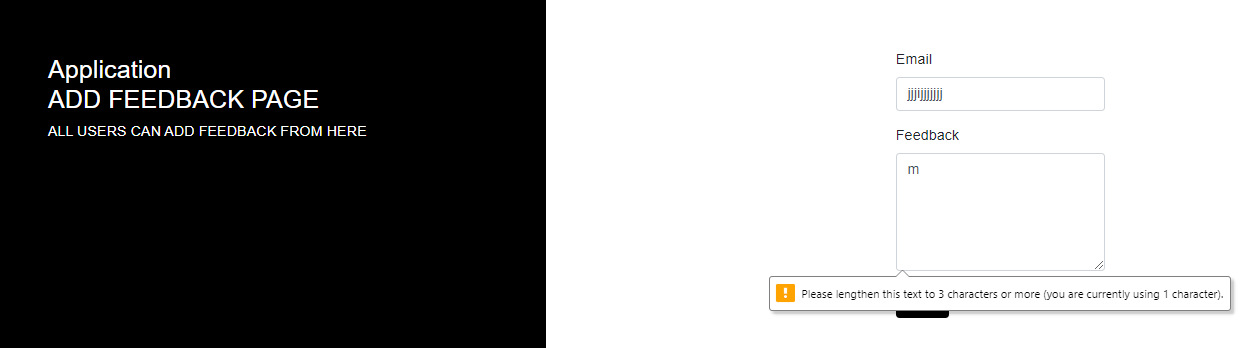
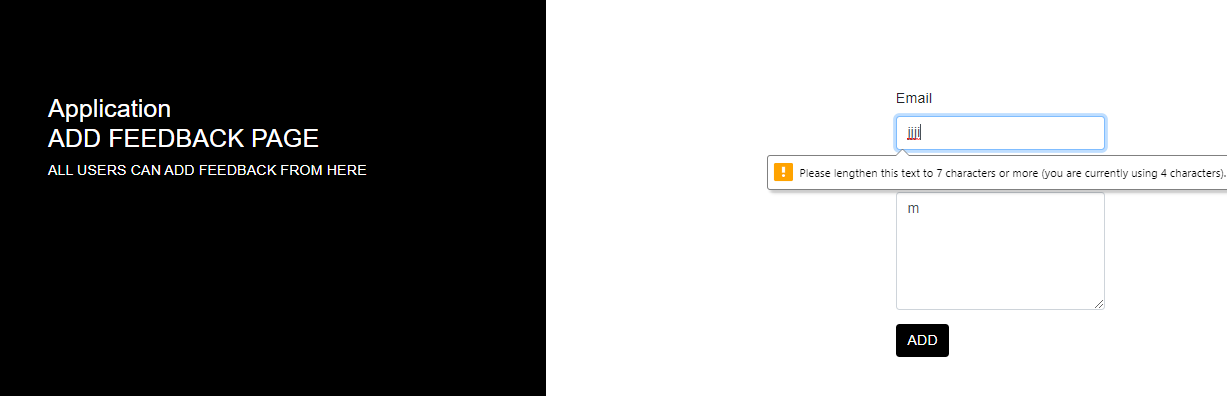
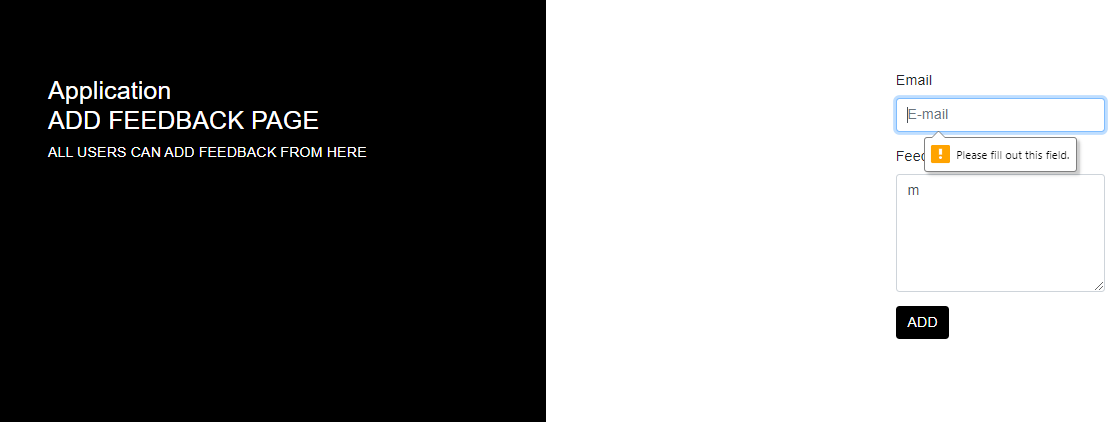
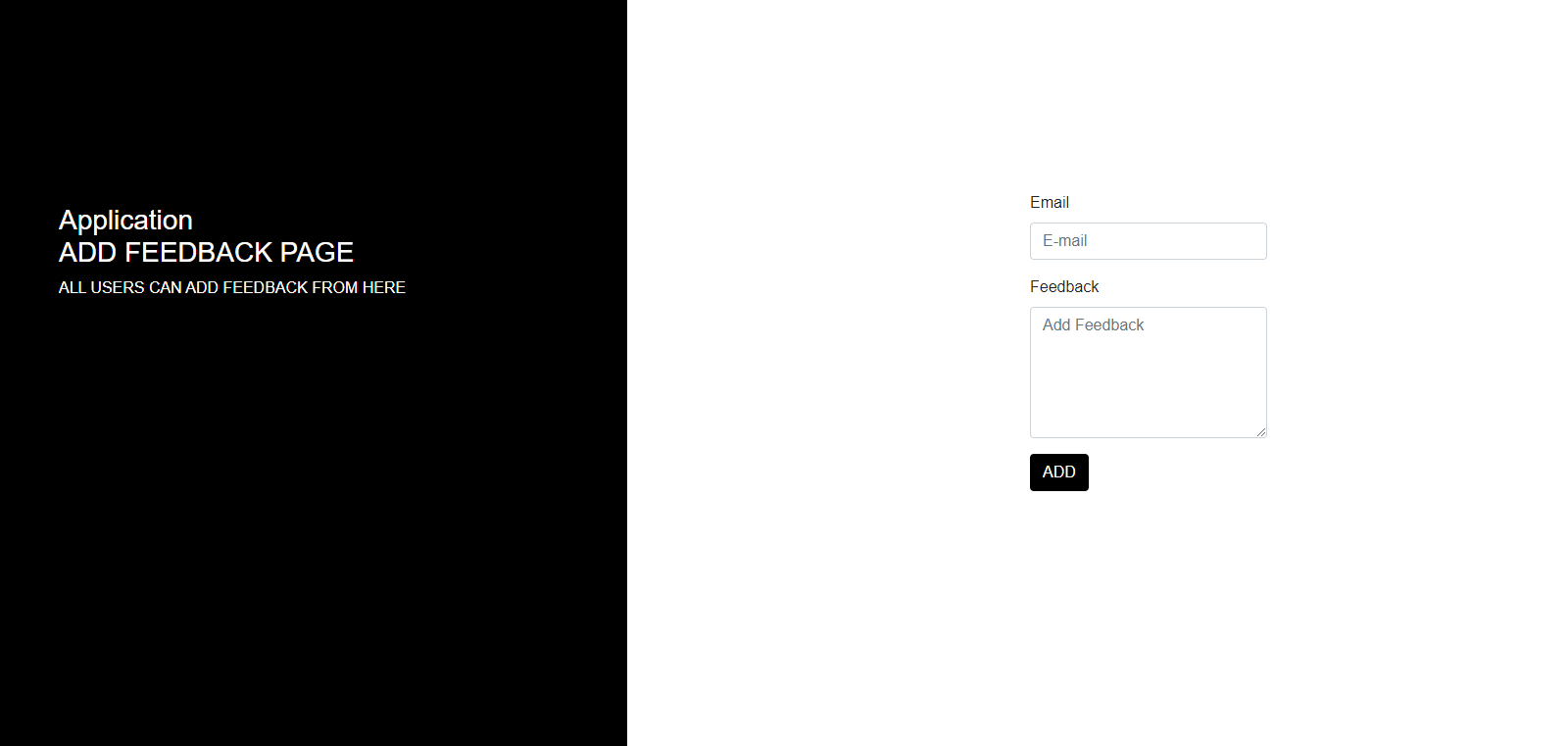
|  |  |  |
| --- | --- | --- |
| **Heuristics** | **Description** | **Potential Solution** |
| **Flexibility** | No exit button | Add exit button. |
| **Error Prevention** | No “forgot password” /”forget username ”button | Add “forgot password” button to modify password  Add “ forget user name” button |
| **Flexibility / Help Users Recognize, Diagnose, and Recover From Errors** | When users forget to fill email/password field /or enters invalid password /email warning explanation message is displayed |  |

Screen12: Compare



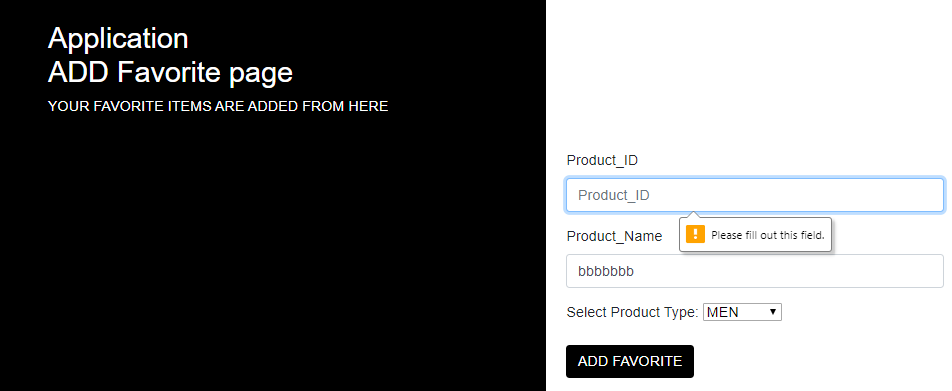
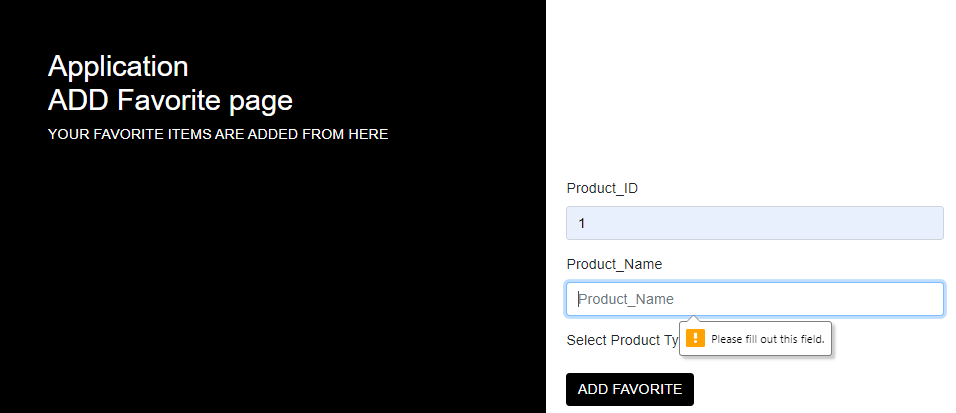
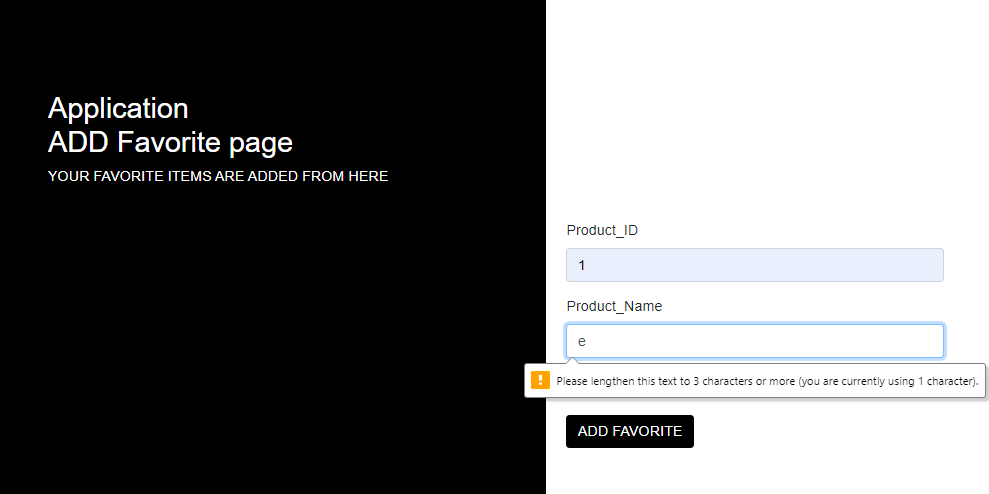
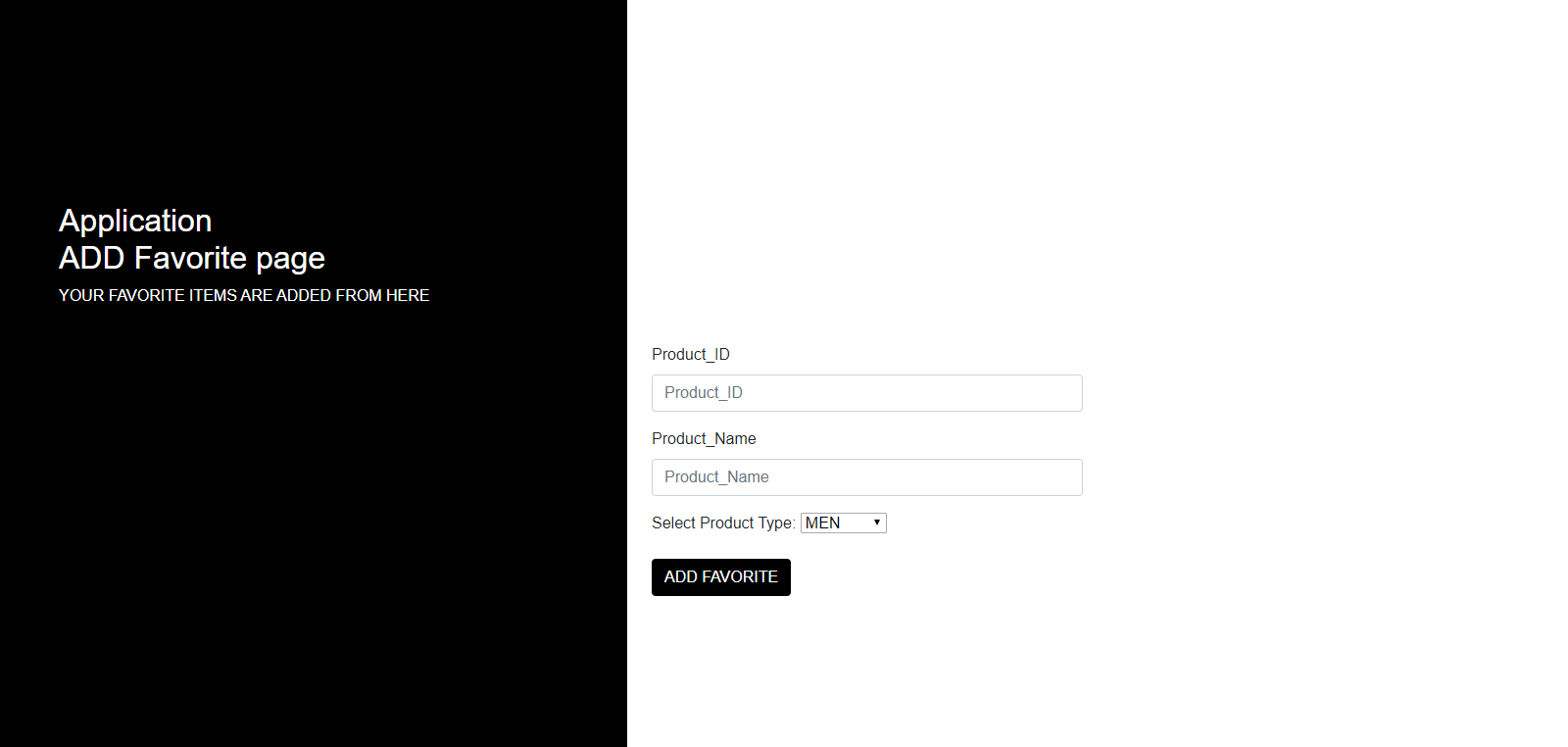
|  |  |  |
| --- | --- | --- |
| **Heuristics** | **Description** | **Potential Solution** |
| **Flexibility** | No table for comparing | Add table for comparing |

Screen13: Add Feedback



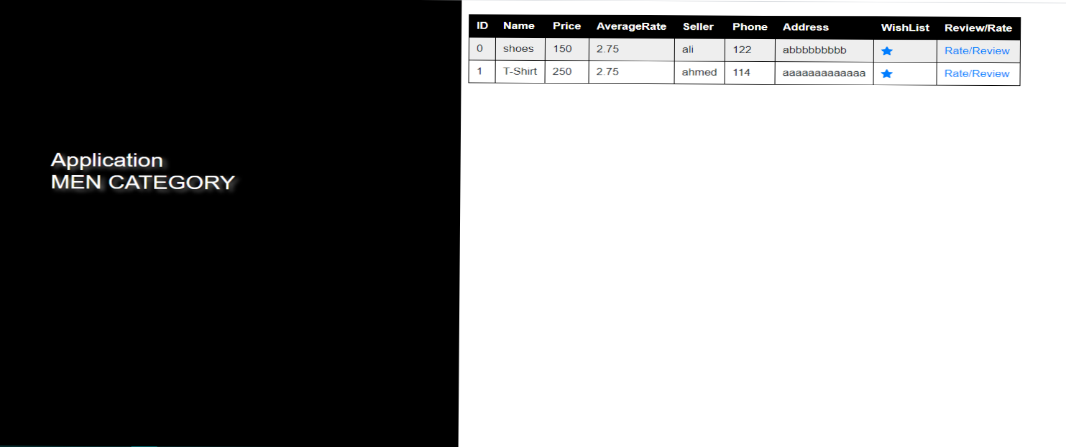
|  |  |  |
| --- | --- | --- |
| **Heuristics** | **Description** | **Potential Solution** |
| **User control &freedom** | User has to redo all his previous steps until he reach the home page | Default menu has to be added to let user move freely |
| **Error prevention/flexibility** | * + User can’t undo his action | * + There must be a confirmation message to ensure that the user want to make this action |

Screen14: Favorite



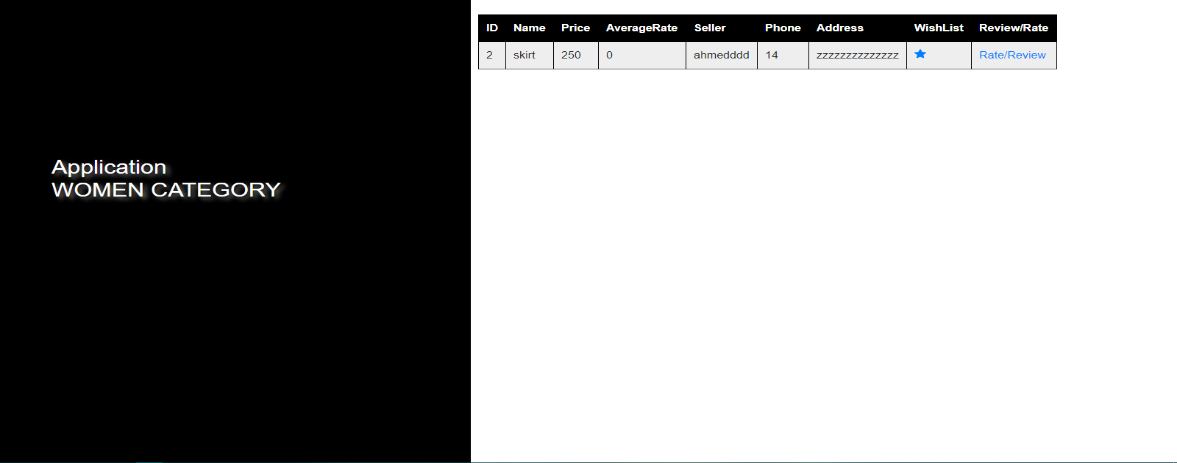
|  |  |  |
| --- | --- | --- |
| **Heuristics** | **Description** | **Potential Solution** |
| **Flexibility / Help Users Recognize, Diagnose, and Recover From Errors** | When users forget to fill product id/product name field /or enters invalid value warning explanation message is displayed |  |
| **Error prevention/flexibility** | * + User can’t undo his action | * + There must be a confirmation message to ensure that the user want to make this action |

Screen 15: men



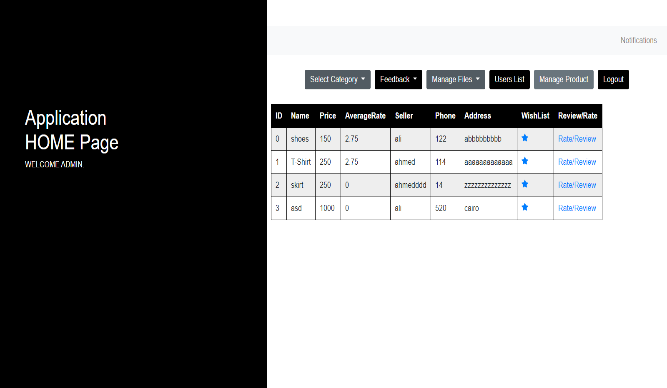
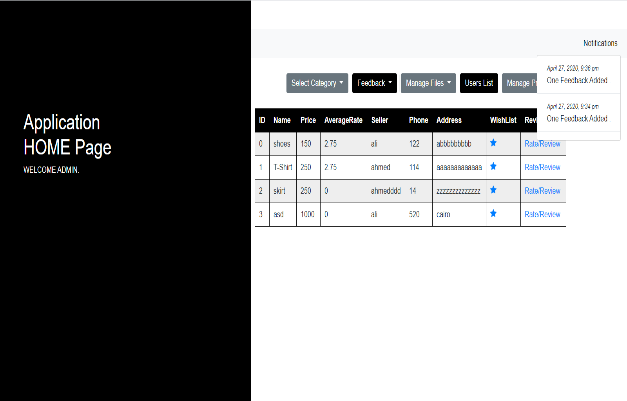
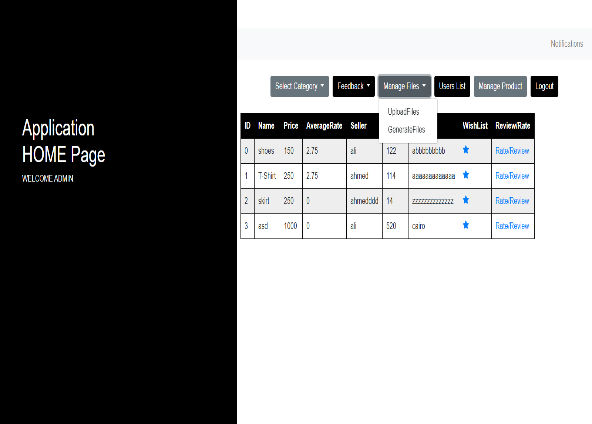
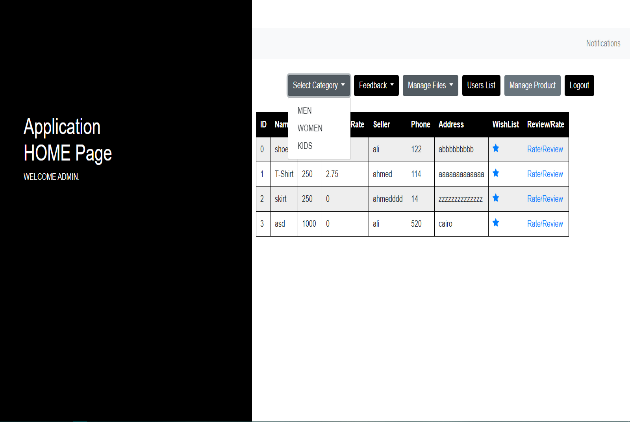
|  |  |  |
| --- | --- | --- |
| **Heuristics** | **Description** | **Potential Solution** |
| **Help And Documentation** | not enough information to make users use it easily | Add “help” button |
| **Flexibility** | No “exit” button | Add “exit” button. |

Screen 16: women



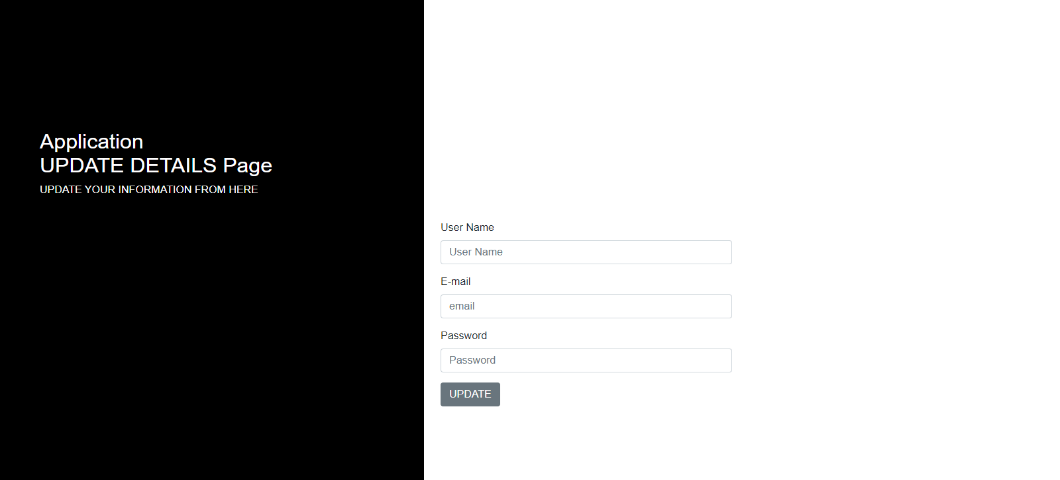
|  |  |  |
| --- | --- | --- |
| **Heuristics** | **Description** | **Potential Solution** |
| **Help and documentation** | not enough information to make users use it easily | Add “help” button |
| **Flexibility** | No “exit” button | Add “exit” button. |

Screen 17: Admin Panel



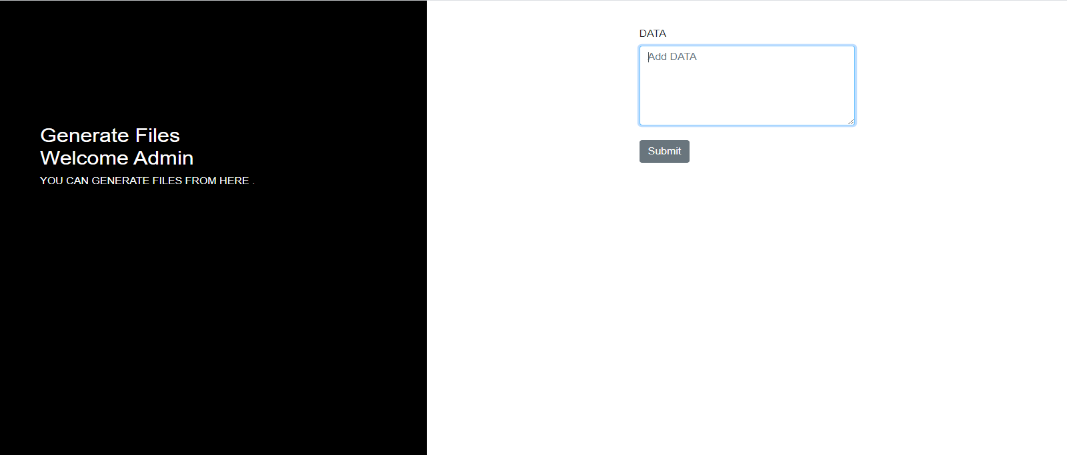
|  |  |  |
| --- | --- | --- |
| **Heuristics** | **Description** | **Potential Solution** |
| **Minimalist** | Too much in one page | Reduce or separate functions to another page |
| **Visibility** | Notification is not clear | Add detail to notification button |

Screen18: Update Details



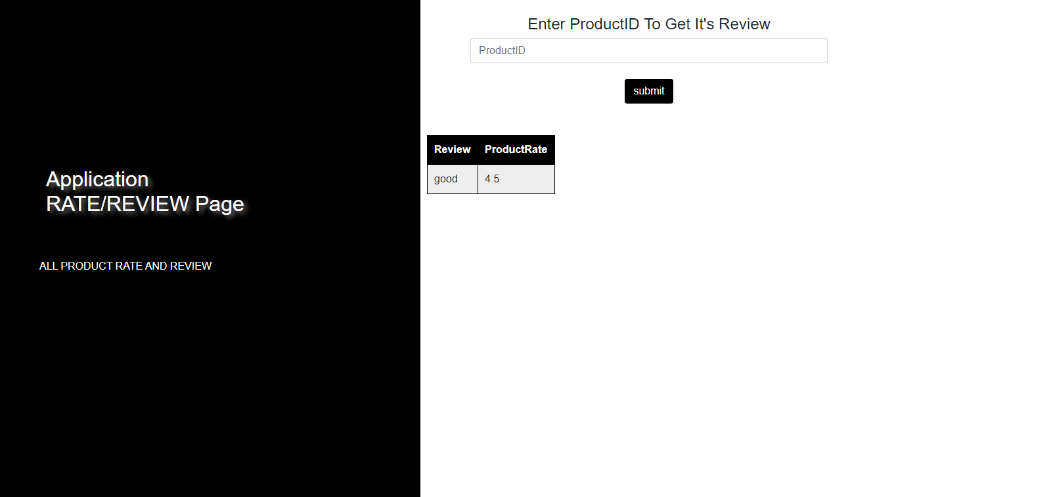
|  |  |  |
| --- | --- | --- |
| **Heuristics** | **Description** | **Potential Solution** |
| **Flexibility** | No exit button | Add exit button. |
| **Visibility/help users recognize and recover from errors** | There is no confirmation message when users want to undo his action | Confirmation message displayed before submitting |

Screen 19: generate file



|  |  |  |
| --- | --- | --- |
| **Heuristics** | **Description** | **Potential Solution** |
| **Visibility/help & Documentation** | Limited information | Manual guide should be added |

Screen 20: View Rate/Review



|  |  |  |
| --- | --- | --- |
| **Heuristics** | **Description** | **Potential Solution** |
| **Visibility/help & Documentation** | Limited information | Manual guide should be added |

**2- Evaluation Report**

|  |  |  |
| --- | --- | --- |
| **Screen** | **Heuristic** | **Description** |
| **1:User Panel** | **Flexibility** | * + Users should know more information about products like images ,colors ,and available size   + What if users want to buy products or contact with sellers? Cart and chat feature should added |
| **2:Manage Products** | **Error Prevention** | * + What if I made a mistake while filling the form? There must be a confirmation message to undo my action |
| **3:Search** | **User control &freedom** | * + Why users had to redo all the previous steps to reach the home page?   + Tabs should be added that make users move freely |
| **3:Search** | **Recognition rather than recall** | * + When users search for non-exist products some recommendations should be displayed that helps the users to find what they are looking for |
| **4:Kids** | **Flexibility** | * + No close button for users |
| **4:Kids** | **Visibility** | * + Not enough product information for users to know size ,image , colors available |
| **5:register** | **Help users recognition, diagnose and recover from errors** | * + When users forget to fill a required field warning message should be displayed to register correctly |
| **5:register** | **Help &document** | * + Manual guide or help screen should be added to help the beginners how to interact with the system |
| **5:register** | **Flexibility** | * + No close button for users |
| **6:rate /review** | **Flexibility** | * + What if user forget product id? There must be a table for products in the same page |
| **6:rate /review** | **Flexibility/visibility** | * + Users’ submitted review/rate must be displayed to make sure from his submitted data |
| **6:rate /review** | **User control &freedom** | * + Why users had to redo all his previous steps to return to the home page or do any other action? Tabs should be added that users can move freely |
| **6:rate /review** | **Error prevention/ User control &freedom** | * + Why user is in charge to add only a rate/review and he can’t undo his action?   + Confirmation message should be displayed to ensure his action   + What if user wants to edit or delete his review? Delete /edit button should be displayed |
| **7:upload file** | **Visibility/help & document** | * + Users don’t know what they had to do so help screen should be added |
| **7:upload file** | **Error prevention** | * + Confirmation message should be displayed before submitting |
| **8:view users** | **Visibility /Flexibility** | * + Limited information   + Admin can’t remove users , delete button should be added   + Add exit button |
| **9:view feedback** | **Visibility /Flexibility/User Control &Freedom** | * + Add exit button   + User can’t edit, delete his feedback   + Add Edit/delete buttons |
| **10:delete feedback** | **Error prevention/flexibility** | * + User can’t undo his action   + There must be a confirmation message to ensure that the user want to make this action |
| **10:delete feedback** | **Flexibility** | * + Add exit button   + User might forget feedback id   + Feedback table has to be displayed |
| **11: Login page** | **Flexibility** | No exit from this screen |
| **11: Login page** | **Error prevention** | * Forgot password is good but no solution for forgetting username |
| **12: compare** | **Flexibility** | * Table to compare products |
| **13:Add Feedback** | **Error prevention/flexibility** | * + There must be a confirmation message to ensure that the user want to make this action |
| **14:favorite** | **Error prevention/flexibility** | * + There must be a confirmation message to ensure that the user want to make this action |
| **15: men** | **Help and documentation** | * not enough information to make users use it easily |
| **15: men** | **flexibility** | No “exit” button |
| **16: women** | **Help and documentation** | * not enough information to make users use it easily |
| **16: women** | **Flexibility** | * No “exit” button |
| **17: admin** | **Minimalist** | * Too much for one page |
| **17: admin** | **Visibility** | * Add detail to notification button |
| **18: update details** | **Flexibility** | * Add exit button |
| **18: update details** | **Visibility/help users recognize and recover from errors** | * What should user do if he wants to undo his action? * Confirmation message displayed before submitting |
| **19: generate file** | **Visibility/help & Documentation** | * Manual guide should be added to help users |
| **20: View Rate/Review** | **Visibility/help & Documentation** | * Manual guide should be added to help users |

* **Team roles:**

|  |  |
| --- | --- |
| **Name** | **Role** |
| **Nahla Fathy** | 1.Activity diagram  2.Deployment Diagram  3.Archeticture diagram  4.class diagram  5.SRS (from 1 to 4)  6.Documenting sectionA  7.Database specification  8.Design pattern  9.Use case description  10. part 2 structing criteria  11. Customer Dashboard frontend  12.Implementing (login, add favorite, compare, calculate average rate, upload files, generate files) with their frontend  13.popup and Database classes |
| **Noura Yasser** | 1.Activity Diagram  2.SRS (from 5 to 9)  3.Documenting sectionB  4.Package diagram  5.part 4 no (20) (from 1 to 10)  6.customer Dashboard front end  7.class diagram  8.Usecase description  9.implemeting (view product, view user, view rate/review) with their front end |
| **Ali gad** | 1.Activity diagram  2.sequence diagram  3.system sequence diagram  4. class diagram  5.usecase diagram  6.white box testing  7.implementing (edit, delete, add, search products) with their front end |
| **Mohrail Alfy** | 1.state machine diagram  2.class diagram  3.usecase diagram  4.package diagram grouping relevant use cases  5.Database specification  6.Admin dashboard frontend  7.implementing (register, update details) with their front end and popup class  8.question (17) part 4 |
| **Martina Mosaad** | 1.Archeticture diagram  2.Archeticture Pattern  3. questions (15,16) part4  4.implementing (add, delete feedback) with their front end  5.Design Pattern  6.Writting about interaction design principle |
| **Heba Abdelaziz** | 1.communication diagram  2.class diagram  3.Data base specification  4.Design Pattern  5.Anaylsis pattern  6.questions (n, o) part 2  7.black box testing  8.implementing (add rate/review) with their front end + notification and popup classes |
| **Menna Abdelmajeed** | 1.Object diagram  2.class diagram  3.DataBase specification  4.11(c) list of user interfaces based on activity diagram  5.questions (I, m) part 2  6.part 4 question (20) from (11 to 20)  7.implementing (view feedback category class(view men, view women, view kids) with their frontend  8.Admin dashboard frontend |