# Chapter 2

## 2. Analysis

## 2.1 Introduction to analysis

To cover the interrelationship between the project nodes and the arcs the details analysis should be taken. Analysis aid to break the project component into units and provides a systematic examination of information.

## Project Analysis

It is the analysis that explorer the project requirement, problem and its solution thus the first and the foremost goal of the analysis is to understand the requirement and to formulate it.

There should be enough information in order to do analysis. The first steps to analysis were taken by gathering information about the project. After the information gathering, the analysis was taken which facilitate the system requirement, which development methodology should be taken, what technical tools are required and how to follow a systematic approach to make the project successful. All those development methodologies, technical tools and some other requirement for the project are discussed below:

## 2.2 Analysis Methodology

Under the analysis methodology, I have chosen to use Object Oriented Analysis.

## Object-Oriented Analysis

The object-oriented analysis does not focus solely on the processes or data of a system but views an information system as a collection of interacting object that works together to accomplish tasks. In a short, it provides various technics such as **object modelling, dynamic modelling** and **functional modelling** that are used in conjunction with each other for object-oriented analysis.

## The three aspects of object-oriented analysis:

**Object Modelling:**

Object modeling develops the static structure of the system. It illustrates the object classes, their relationship and attributes, and operation as a class diagram.

**Dynamic Modelling**

After the static behavior of the system is analyzed it the dynamic modeling that illustrates the behaviourof the system over time and the flow of control and events in Event-Trace Diagrams and State Transition Diagram.

**Functional Modelling:**

The functional modeling is the final component of the analysis part which consist of a set of DFDs that illustrate the internal processes independently from how these processes are performed.

## Application of an object orient analysis:

* Suitable for medium to large scale project
* Organizations including business, bank, airports, e-commerce, etc.

## Advantage of object-oriented analysis:

* Re-usability of analysis, object, design, and programming.
* Improve communication among users, analyst, designers, and programmers.
* More flexible and easier to make updates in response to changing user requirement.
* It can be upgraded from small to the large system at greater ease than in system following structure analysis.

## The disadvantage of object-oriented analysis:

* It has been accused of being too technical and complicated.
* Processes and data flow are often poorly illustrated and described.
* Communication between the object is difficult to identify.
* It is too large to represent all the interface into a single diagram.

## 2.3 Feasibility Study

Feasibility study as the name suggests, it is used to determine the viability of an idea, and ensure a project is socially, financially, technically and legally feasible. It provides a clear justification whether it is worth to invest in a project or not. The feasibility study of Goods Exchange Hinge system in some area of project feasibility is given below:

**Technical Feasibility:**

Technical feasibility focus on the technical resources available for any organization. It also involves some other technical requirement such as software and hardware for a proposed system. My proposed system is for academic solo computing project. So, it is not going to cover a large area that is why all the technical resources fulfill and handle by me. The technical resources such a machine (Laptop), development tool (IDE), manpower which is myself are fulfilled.

**Financial Feasibility:**

This area covers the cost and benefits analysis of the project. The financial feasibility study helps decision taker to determine the positive benefits that the project will provide. As mention earlier in a technical study this is only an academic computing project so, there won’t be any investor. This project going to examines the eligibility of individual students how he/she going to propose a system.

**Legal Feasibility:**

Legal feasibility study investigates the causes and the consequences of the proposed system whether the project violence the government laws, social media rules, and data protection acts. The Goods Exchange Hinge is only going to store some basic information of users. The clear privacy policy is described under the term and condition page.

**Scheduling Feasibility:**

This is the most important steps for any project. It is always worth to estimate proper time and dates for the project. The project will fail if it is not complete on time. For this, proper scheduling for a project was constructed using project libre software which will navigate step by steps to implements the project.

## 2.4 Software Requirement Specification (SRS)

The software requirement specification is documentation which helps to details examined how the system is expected to perform. There is some type of requirement which is given below:

### 2.4.1 Functional Requirement

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S.N. | Title | Description | Reason | Dependencies |
| F1 | User Authentication | User must be registered with a certain role specified and navigate to the application according to their role. | To maintain confidentiality and authenticity so that only verified user can access the application. | NA |
| F2 | Add Product | User should be able to add product with appropriate validation and message. | To add product in order to exchange the product. | F1 |
| F3 | Update Product | User should be able to edit the existing product details. | To edit existing product details in case of incorrect product details entry. | F2 |
| F4 | Product Deletion | User should be able to delete the existing product with an appropriate confirmation message. | To delete product which is exchanged or expired. | F2 |
| F5 | Provide access to all use case | User should be able to use all use case with a fix navigation bar. | To allow the user to access all use case to a fix navigation bar. | F1 |
| F6 | Product Exchange | User should be able to describe how to exchange a product. | To make the user easy to exchange goods. | F1,F2 |
| F7 | Get Notification | User should be able to notify about goods/products. | To notify a user about goods via contact number or via email. | F1,F5 |
| F8 | Add Category | User should be able to add product category. | To add the product into a different category. | F1 |
| F9 | Search product | User should be able to search product without login to see the available product for exchange. | To see the appropriate product for exchange. | NA |
| F10 | Login Count | User login should be count for analysis purpose. | To count user login to analyze how many times a user visits the site. | F1 |
| F11 | View User Profile | User should be able to view their individual profile to confirm their details. | To see the user details and perform CRUD operation if required. | F1 |
| F12 | Product Details | User should be able to view product details for confirmation product details. | To confirm product details are correct and perform CRUD operation if required. | F2 |
| F13 | Product Exchange open for | User should be able to mention which product they want to exchange for. | To provide information which product user want if user exchanges a product. | F1,F2 |
| F14 | Login session termination | User should be able to log out safely. | To terminate the user login session to maintain security. | F1 |
| F15 | Chat Boot | User should be able to communicate with the admin. | To get help if required. | NA |

### 2.4.2 Non-Functional Requirement

|  |  |  |
| --- | --- | --- |
| S.N. | Category | Description |
| NF1 | GUI | Application interface should be user-friendly to attract user and user should be able to navigate where they want. |
| NF2 | Reliability | The application should be reliable to save the user time. The application should be able to provide what user want. |
| NF3 | Performance | To keep user using the application the performance should not be latency and the application should be able to navigate in real-time speed. |
| NF4 | Quality | The application should be able to run smoothly on a modern web browser with minimum requirement. |
| NF5 | Safety | There must be terms and condition described privacy policy and how to use the application. |
| NF6 | Supportability | There must be support available for the user whenever required. For e.g.: -Chat boot for a user to directly communicate with the admin. |
| NF7 | Maintainability | The application should be flexible to use. In case there exist any issue it should be maintained and fixed as fast as possible. |
| NF8 | Usability | It is difficult for the user to learn a new system. So, the application should be easy to use and it should be accessed from a different point of view |
| NF9 | Low perceived workload | User should be able to perform the task in a single attempt to accomplish a particular task. |
| NF10 | Availability | All parts of the application should be available to the user when required. In case of maintainability, user must be notified earlier. |
| NF11 | Scalability | If there is more user visit the sites then it needs to grow up its scale in both hardware and software implication to serve more user. |

### 2.4.3 Moscow Prioritization

|  |  |  |  |
| --- | --- | --- | --- |
| S.N. | Requirement | Moscow | Reason |
| F1 | User Authentication | M | Critical for security so the only valid user should be allowed |
| F2 | Add Product | M | The basic requirement of the application |
| F3 | Update Product | M | The basic requirement of the application |
| F4 | Product Deletion | M | Basic requirement of application |
| F5 | Provide access to all use case | M | The basic requirement of the application |
| F6 | Product Exchange | M | The basic requirement of the application |
| F7 | Get Notification | S | User should be pre-notify in order to exchange goods. |
| F8 | Add Category | S | The basic requirement of the application |
| F9 | Search product | C | Basic requirement of application |
| F10 | Login Count | M | To track user visit in order to improve application usability |
| F11 | View User Profile | M | The basic requirement of the application |
| F12 | Product Details | M | Basic requirement of application |
| F13 | Product Exchange open for | M | User must mention what they want if they exchange goods |
| F14 | Login session termination | M | User session should be terminated after logged out for security reason. |
| F15 | Chat Boot | C | The basic requirement of the application |
| NF1 | GUI | M | The basic requirement of the application |
| NF2 | Reliability | M | The basic requirement of the application |
| NF3 | Performance | M | Basic requirement of application |
| NF4 | Quality | M | Basic requirement of application |
| NF5 | Safety | M | The basic requirement of the application |
| NF6 | Supportability | S | The basic requirement of the application |
| NF7 | Maintainability | M | The basic requirement of the application |
| NF8 | Usability | S | The basic requirement of the application |
| NF9 | Low perceived workload | M | The basic requirement of the application |
| NF10 | Availability | M | The basic requirement of the application |
| NF11 | Scalability | M | The basic requirement of the application |

### 2.4.4 Hardware Software Specification

|  |  |  |
| --- | --- | --- |
| S.N. | Hardware Specification | Software Specification |
| 1. | CPU @2.20GHz | Browser:-chrome, Microsoft Edge |
| 2. | RAM 4.00 GB | Server:-XAMPP server v3.2.2 |
| 3. | X64-based processor | Database:-MySQL |
| 4. |  | IDE:-Spring Tool Suit (STS) |

## 2.5 Use Case Diagram

A use case diagram is an important tool in managing the system abstraction. It allows representing the broad interactions between the part of the system. Similarly, it is also used to represent the set of functionality that must be supported for each part of actors. The use case diagram is designed in using a visual paradigm tool. The use case diagram for a system is given below:

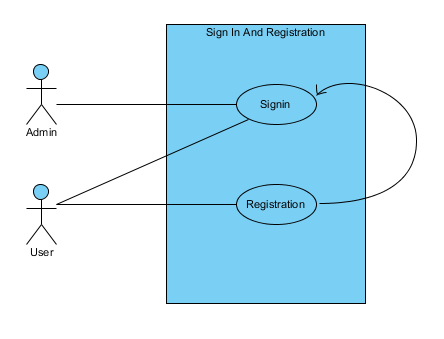


Figure Use case sign in and registration

|  |  |
| --- | --- |
| S.N | Use Case: - Sign In |
| Explanation | The basic requirement of application through which user can access the application with valid credential in-order to do further process such as add products, view profile, etc. |
| Prime Actor | User |
| Second Actor | Admin |
| Main Flow | 1. User login with the valid credential. 2. User navigated to the home page. 3. User proceeds the further process. |
| Second Flow | 1. User input invalid credential 2. System detects and validate the user input. 3. The possible validation error message is shown to the user. 4. User is expected to enter correct credential. 5. Repeat from step 2 if wrong credential input otherwise forwarded to the home page if right credential input. |

|  |  |
| --- | --- |
| S.N | Use Case: - Registration |
| Explanation | To login the application and manipulate the require information user need to register first. |
| Prime Actor | User |
| Second Actor | NA |
| Main Flow | 1. User clicks on signup button to registration. 2. A signup form will appear. 3. User needs to fill up the required data into the field and submit the form. 4. System validates and verify the input data. 5. System store the input data into the database. 6. User will redirect to the login page to login into the application. |
| Second Flow | 1. User input invalid data to the field. 2. System validates and verify the data. 3. System show appropriate validation message. 4. User is required to input correct data into the field. 5. The system will repeat the process from step 4 until correct data input. |

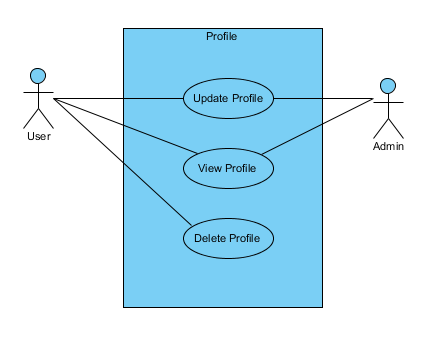


Figure Use case update, delete and view profile

|  |  |
| --- | --- |
| S.N | Use Case: - Update Profile |
| Explanation | User may input invalid or incorrect data so to maintain data integrity update case is required. |
| Prime Actor | User |
| Second Actor | Admin |
| Main Flow | 1. User login with the valid credential. 2. User navigates to the profile page. 3. User click on the update button and the form will open with existing data. 4. User edit the require data into the field and click on the submit button to submit the profile form. 5. System validates and verify the input data. 6. System updates the data into the database. 7. System redirects to the existing page. |
| Second Flow | 1. User input incorrect data into the field. 2. System validates and verify the input data. 3. System shows the necessary validation message. 4. User is required to input correct data. 5. The system will repeat from step 5 until the correct data input. |

|  |  |
| --- | --- |
| S.N | Use Case: - View Profile |
| Explanation | A fundamental requirement of the application, the user has the right to visit and view their individual profile. |
| Prime Actor | User |
| Second Actor | NA |
| Main Flow | 1. User login into the system with the valid credential. 2. User navigates to the profile page and view profile. |
| Second Flow | 1. User input wrong credential. 2. The system validates and verifies the input data. 3. System show appropriate validation message. 4. User is required to input correct data. 5. Repeat from step 1 from the main flow section until valid credential was input. |

|  |  |
| --- | --- |
| S.N | Use Case: - Delete User |
| Explanation | A basic requirement of the application, delete the user if the user is no longer using the application. |
| Prime Actor | Admin |
| Second Actor | NA |
| Main Flow | 1. Admin login into the application with the valid credential. 2. Admin checks the login dates and time with login counts. 3. If user not log in at least once within six months, then admin can delete the user. 4. User will be removed from the database. |
| Second Flow | 1. Admin input wrong credential. 2. The system validates and verifies the input data. 3. System show appropriate validation message. 4. Admin is required to input correct data. 5. Repeat from step 1 from the main flow section until valid credential was input. |

|  |  |
| --- | --- |
| S.N | Use Case: - Delete Profile |
| Explanation | A fundamental requirement of the system, every user has the right to delete their individual profile if the user doesn’t want to use the application. |
| Prime Actor | User |
| Second Actor | NA |
| Main Flow | 1. User login to the system with the valid credential. 2. User navigates to the profile page. 3. User click delete button to delete their profile. |
| Second Flow | 1. User input wrong credential. 2. The system validates and verifies the input data. 3. System show appropriate validation message. 4. User is required to input correct data. 5. Repeat from step 1 from the main flow section until valid credential was input. |

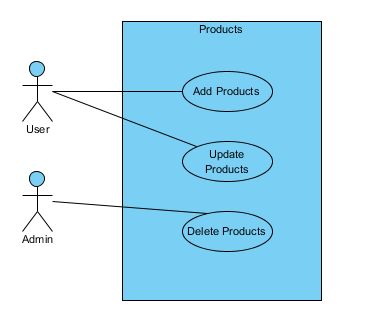


Figure Use Case add, update and delete products

|  |  |
| --- | --- |
| S.N | Use Case: - Add Product |
| Explanation | User needs to add a product with necessary details in order to exchange goods with one another. |
| Prime Actor | User |
| Second Actor | NA |
| Main Flow | 1. User login to the system with the valid credential. 2. The user navigates to the product page in order to add products. 3. User click adds a button to add a product and save into the database. |
| Second Flow | 1. User input wrong credential. 2. The system validates and verifies the input data. 3. System show appropriate validation message. 4. User is required to input correct data. 5. Repeat from step 1 from the main flow section until valid credential was input. |

|  |  |
| --- | --- |
| S.N | Use Case: - Update Product |
| Explanation | This use case needs to maintain data integrity whenever there in incorrect data input. |
| Prime Actor | User |
| Second Actor | NA |
| Main Flow | 1. User login with the valid credential. 2. After login user navigates to the product page where the user clicks on the update button to edit details. 3. The user submits the product edit form. 4. The system validates and verifies the input data. 5. System store the changes into the database. 6. User will redirect to the existing page with data. |
| Second Flow | 1. User input wrong credential. 2. The system validates and verifies the input data. 3. System show appropriate validation message. 4. User is required to input correct data. 5. Repeat from step 1 from the main flow section until valid credential was input. |

|  |  |
| --- | --- |
| S.N | Use Case: - Delete Product |
| Explanation | This use case is required after the successful exchange of a product. |
| Prime Actor | Admin |
| Second Actor | NA |
| Main Flow | 1. Admin login into the application with the valid credential. 2. Admin check if the product has been exchanged or not. 3. If the product is exchanged admin can successfully delete the product from the database. |
| Second Flow | 1. Admin input wrong credential. 2. The system validates and verifies the input data. 3. System show appropriate validation message. 4. Admin is required to input correct data. 5. Repeat from step 1 from the main flow section until valid credential was input. |

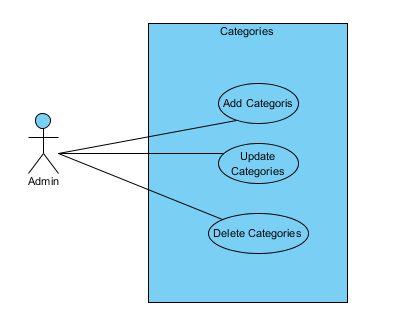


Figure Use Case add, update and delete category

|  |  |
| --- | --- |
| S.N | Use Case: - Add Category |
| Explanation | A product should be added into the specific category for this use case is required. |
| Prime Actor | Admin |
| Second Actor | NA |
| Main Flow | 1. Admin login into the application with the valid credential. 2. Admin navigates to the dashboard. 3. Admin adds necessary product category. |
| Second Flow | 1. Admin input wrong credential. 2. The system validates and verifies the input data. 3. System show appropriate validation message. 4. Admin is required to input correct data. 5. Repeat from step 1 from the main flow section until valid credential was input. |

|  |  |
| --- | --- |
| S.N | Use Case: - Update Category |
| Explanation | There may be a case where input categories were incorrect so to maintain data integrity this use case is vital. |
| Prime Actor | Admin |
| Second Actor | NA |
| Main Flow | 1. Admin login into the application with the valid credential. 2. Admin navigates to the dashboard. 3. Admin update necessary product categories. |
| Second Flow | 1. Admin input wrong credential. 2. The system validates and verifies the input data. 3. System show appropriate validation message. 4. Admin is required to input correct data. 5. Repeat from step 1 from the main flow section until valid credential was input. |

|  |  |
| --- | --- |
| S.N | Use Case: - Delete Category |
| Explanation | There may be cases arises where the specific category is no longer required for this reason a delete use case is needed. |
| Prime Actor | Admin |
| Second Actor | NA |
| Main Flow | 1. Admin login into the application with the valid credential. 2. Admin navigates to the dashboard. 3. Admin deletes necessary product categories. |
| Second Flow | 1. Admin input wrong credential. 2. The system validates and verifies the input data. 3. System show appropriate validation message. 4. Admin is required to input correct data. 5. Repeat from step 1 from the main flow section until valid credential was input. |

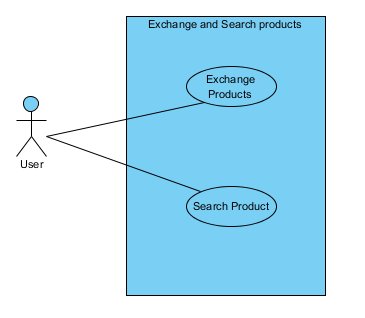


Figure Use Case exchange and search products

|  |  |
| --- | --- |
| S.N | Use Case: - Search Product |
| Explanation | It is crucial to look for a product one by one. So, to get a specific product instant with the best possible matches for the given input from a large pool of data. |
| Prime Actor | User |
| Second Actor | NA |
| Main Flow | 1. There is no need to login into the application in order to search for a product. 2. User needs to visit the application. |
| Second Flow | 1. User must have to visit the application/website. |

|  |  |
| --- | --- |
| S.N | Use Case: - Exchange Product |
| Explanation | The main theme of the system is exchanged a product. |
| Prime Actor | User |
| Second Actor | NA |
| Main Flow | 1. User login with the valid credential. 2. After login user navigates to the product page to see the product details. 3. User contact with the product owner to exchange a product with each other. |
| Second Flow | 1. User input wrong credential. 2. The system validates and verifies the input data. 3. System show appropriate validation message. 4. User is required to input correct data. 5. Repeat from step 1 from the main flow section until valid credential was input. |

## 2.6 Initial Class Diagram

The class diagram is a static view of the application. it is used to represent the different aspect of the system such as to describing and visualizing the system. The main purpose of the class diagram is to model the static view of the system.

The class diagram can be obtained by performing Natural Language Analysis (NLA). It helps to obtain a list of possible candidate classes, attributes, and their relationship. By identifying the list of verbs, adjectives, and nouns a possible list of classes, attributes and functions can be obtained respectively.

### The scenario of the application to perform NLA is described below:

Back into the time around 6000 BC introduced by Mesopotamia tribes, bartering was adopted by Phoenicians. Goods were exchanged for food, tea, weapons, and spices. At those times, human skulls were used as well. People use to exchange goods inconveniently. For example, a human skull was exchanged with the sheep which is inconvenient and miss-match. Civilization takes place, things are understood and managed in a proper way.

Keeping the above thing in mind I want to implement this idea in a more appropriate way where peoples can find the list of available goods in several categories. There is a difference in the application rather than swapping goods randomly I want to make items value comparable with one another this can be done in two way either the owner of the item can clearly state which type of products or goods they are open for a swap or pay a little amount of money instead.

The application is going to provide a description of goods, a list of recently added products in the home page, searching facility, a login system with security for both user and admin where both user and admin have a different perspective of viewing the application. Both user and admin can perform CRUD operation such user can add product/goods, edit/update product details and their individual profile details, delete the profile if user decides not to use the website/application. Admin have full access to the application such as admin can have access to the dashboard from where the application can be controlled modify and updates.

##### Natural Language Analysis (NLA):

To identify the possible candidate classes all the relevant nouns have been enlisted below:

#### List of Possible Nouns: -

Time, tribes, bartering, goods, food, tea, weapons, spices, place, things, list, categories, value, item, type, products, state, swap, descriptions, amount, money, home, page, facility, login, system, security, user, admin, perspective, operation, details, profile, dashboard.

#### List of possible Adjectives: -

Human, miss-match, proper, above, appropriate, available, several, comparable, open, little, different, edit, update, individual, full.

#### List of possible Verbs: -

Introduced, exchanged, adopted, used, use, managed, keeping, want, implement, swapping, make, done, pay, provide, added, searching, viewing, add, delete, decide, controlled, modify.

After identifying all the possible list of candiate classes, attributes and functions, all the irrelevance and duplication candiates are removed. Words that are too high level of abstration are remove and the words which are out of project scope are also removed.

**Words that are duplicates: -** *proper, swap,admin, details, profile, use*

**Words that are irrelavents: -** *tribes, tea, bartering, weapons, spices, things, home, perspective, human, appropriate, little*

**Words that are out of project scope: -** *place, page, facility, system, above, available, individual, introduced, adpted, managed,want, implement, make*

Final list of candidate are given below:

|  |  |  |
| --- | --- | --- |
| Classes | Attributes | Methods |
| *System*  *Registration*  *Login*  *Users*  *Goods*  *Categories* | *Name*  *Address*  *Contact*  *Dates*  *Time*  *Count*  *Description*  *Location*  *Swap*  *Value*  *open* | *Add*  *Save*  *Delete*  *Update*  *Edit*  *Search*  *Remove*  *Count* |

## Intial Class Diagram:

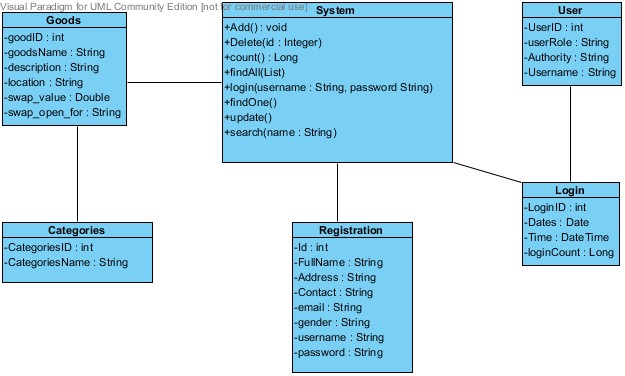


Figure Initial Class Diagram