# 2: Analysis

## 2.1: Introduction

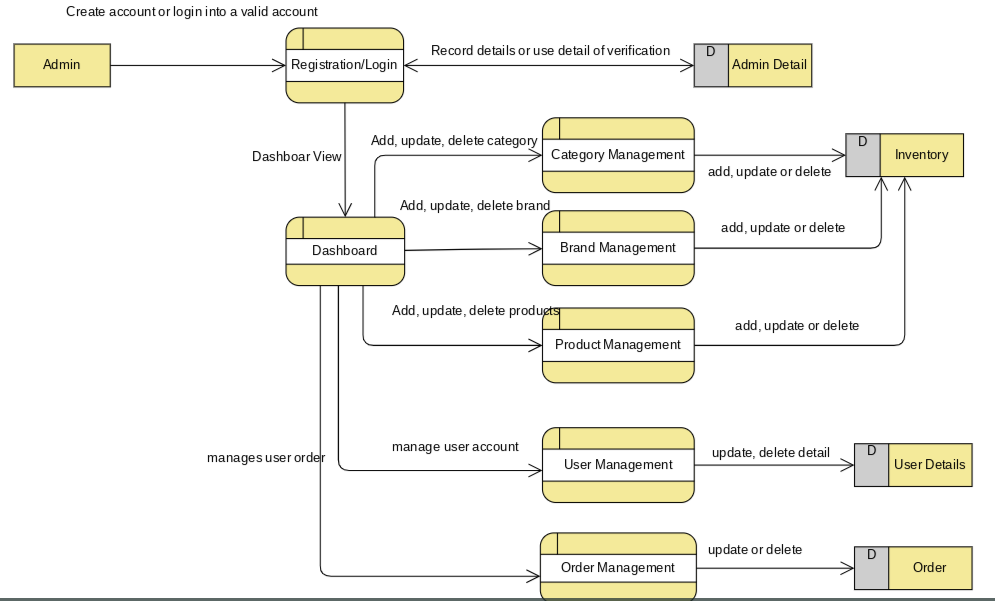
Analysis is a detail study of a complex problem to establish the understanding of the problem which further helps as the basis for problem solving and decision making. Analysis is performed to divide a big complex problem into subproblems to have the better understanding. Analysis helps in finding all the requirements. In software development, analysis helps in gathering user requirements, software functions, etc. which helps in better and systematic development of the software.

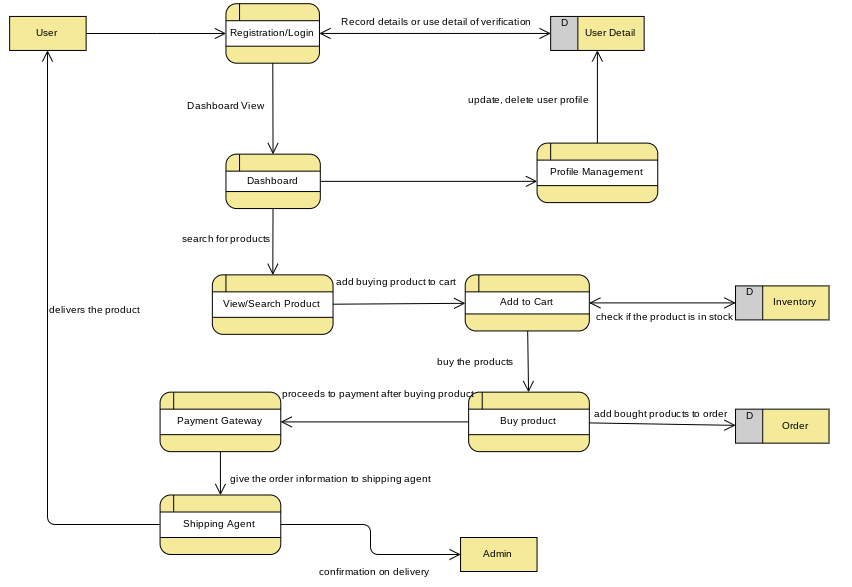
## 2.2: Analysis Methodology

The goal of analysis is to fully understand the systems requirement. For this any appropriate analysis methodology should be used. In any analysis methodology a certain defined step is carried to ensure the maximum understanding of the system. For choosing an analysis methodology we must first know about the system about to be created.

For this project I am using Hard system approach methodology. This methodology is a highly structured and is used to address both qualitative and quantitative problems of the system. It is the approach that follows a set number of logical steps, which is often times iterative, analyze the system and its functions. This methodology uses Structured Systems Analysis and Design Method (SSADM). In this approach a Data Flow Diagram (DFD) should be created which shows the flow of data of the system.

For the software development methodology, I have chosen Iterative process. This will be a good choice as the system now is small but is scalable for future upgrades.





## 2.3: Feasibility Study

Feasibility study is a part of analysis which evaluates the systems potential for success, therefore evaluates five types of study that includes technical feasibility, social feasibility, financial feasibility, legal feasibility. This is usually done to find any problems related to systems development. This study also provides the organization information that could manage or prevent any occurring or potential risks.

* Economic Feasibility: This study typically involves cost/benefit analysis of the system. Furthermore, it helps the organization to determine the cost, benefits and viability of the system before any resources is allocated. Here, cost is not analyzed as it is an academic project.  
  For my project, this feasibility study helps me to determine the positive benefits that the system will provide, and whether the system is scalable as per the increment in users.
* Social Feasibility: This study involves the study of social factors such as political condition, environment around the system, cultural factors, etc.   
  For my project, this feasibility study helps to determine if my there will be any effect in my product due to availability of similar products. It also helps in studying whether the product will be sustainable in the market.
* Technical Feasibility: This study assesses the technical resources available for the completion of the project. It also helps the organization for the evaluation of hardware, software, and other technology requirement of the proposed system.   
  For my project, this feasibility study helps to evaluate if the technical resource is available, is the system going to be compatible with modern emerging technologies, etc.
* Legal Feasibility: This study investigates if any aspects of the system is in conflict with legal requirements like data protection act, zoning laws or social media law.   
  For my project, this feasibility study helps to evaluate if any change in government policy affects the system as there's a big role of trade laws and taxation.

## 2.4: Software Requirement Specification (SRS)

Software Requirement Specification is a detail documentation of the system that is being developed including its functional and non-functional requirements and other relevant cases that helps in making the quality of the system better.

### 2.4.1: Functional Requirement

A functional requirement describes what the system should do. It is the whole function of the system and how it works.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Title | Description | Rational | Dependencies |
| FR1 | Admin Registration/ Login | Administrator should be able to login | To get access to admin dashboard | N/A |
| FR2 | Authentication | The data credentials should be authenticated | To give access to authorized user only | FR1 |
| FR3 | Add Category | Admin should be able to add new category | To add a category for dividing different types of product | FR2 |
| FR4 | Add Brand | Admin should be able to add new brand | To add a brand which provides the product | FR4 |
| FR5 | Add Product | Admin should be able to add new product | To add products in order to sell | FR5 |
| FR6 | Update Category | Admin should be able to update existing category | To edit existing detail of category | FR3 |
| FR7 | Update Brand | Admin should be able to update existing brand | To edit existing detail of brand | FR4 |
| FR8 | Update Product | Admin should be able to update existing product | To edit existing detail of product | FR5 |
| FR9 | Delete Category | Admin should be able to delete existing brand | To delete unused category | FR3 |
| FR10 | Delete Brand | Admin should be able to delete existing brand | To delete obsolete brands | FR4 |
| FR11 | Delete Product | Admin should be able to delete existing product | To delete any product | FR5 |
| FR12 | User Signup | User should be able to register their account | To add their information for buying products |  |
| FR13 | User Login | User should be able to be logged in. | To buy product | FR12 |
| FR14 | View Category, Brand, Product | User should be able to view all the categories, brands and products. | To view all the categories, brand and product | FR3, FR4, FR5 |
| FR15 | Search Product | User should be able to search for products | To search for required product | FR3, FR4, FR5 |
| FR16 | Buy Product | User should be able to buy product | To buy any product | FR3, FR4, FR5, FR13 |
| FR17 | Add Cart | User should have to function to add items to cart and buy in bulk | To add products to cart and buy all at once | FR3, FR4, FR5, FR13 |
| FR18 | View/Update User Information | User should be able to view and update their information | To edit their own information | FR13 |
| FR19 | Delete User | User should be able to delete their account | To delete their information if not used | FR13 |
| FR20 | Cancel bought product | User should be able to cancel any bought products. | To cancel any bought products if not needed anymore | FR3, FR4, FR5, FR13, FR16, FR17 |

### 2.4.2: Non-Functional Requirements

Non-Functional requirement should specify how the system should behave. It is as any requirement that is the quality attribute of any system.

|  |  |  |
| --- | --- | --- |
| ID | Title | Description |
| NFR1 | Efficiency | The system should be efficient for handling high number of requests |
| NFR2 | Security | The system should be secured and admin right should be authorized to admin only |
| NFR3 | Scalability | The system should be able to handle the growing data capacity |
| NFR4 | Data Integrity | The system should assure the stored data is authentic and accurate |
| NFR5 | Availability | The system should be available all the time |
| NFR6 | Reliability | The system should be reliable to perform functions without any failure |
| NFR7 | Maintainability | The system should be easily maintainable with easy error detection and fixes |
| NFR8 | Manageability | The system and its components should be easy to manage |
| NFR9 | Interoperability | This feature helps to facilitate the operation of system with other compatible systems |

## 2.4.3: MoSCoW Prioritization

MoSCoW Prioritization is a prioritization technique for managing the requirements of the system. It prioritizes the requirements in three level of importance: must have, should have, could have and would have.

|  |  |  |
| --- | --- | --- |
| ID | Requirements | MoSCoW |
| FR1 | Admin Registration/ Login | M |
| FR2 | Authentication | M |
| FR3 | Add Category | M |
| FR4 | Add Brand | M |
| FR5 | Add Product | M |
| FR6 | Update Category | S |
| FR7 | Update Brand | S |
| FR8 | Update Product | S |
| FR9 | Delete Category | M |
| FR10 | Delete Brand | M |
| FR11 | Delete Product | M |
| FR12 | User Signup | M |
| FR13 | User Login | M |
| FR14 | View Category, Brand, Product | M |
| FR15 | Search Product | S |
| FR16 | Buy Product | M |
| FR17 | Add Cart | C |
| FR18 | View/Update User Information | S |
| FR19 | Delete User | M |
| FR20 | Cancel bought product | W |
| NFR1 | Efficiency | M |
| NFR2 | Security | M |
| NFR3 | Scalability | S |
| NFR4 | Data Integrity | M |
| NFR5 | Availability | M |
| NFR6 | Reliability | M |
| NFR7 | Maintainability | M |
| NFR8 | Manageabilty | S |
| NFR9 | Interoperability | C |

## 2.4.4: Hardware/Software Specification

**Hardware Specification**

* Processor: any Intel or AMD based CPU with 1.5GHz clock speed
* RAM: 4GB or higher
* Display: SVGA with 1024\*768 pixels +
* Peripherals: Mouse and Keyboard

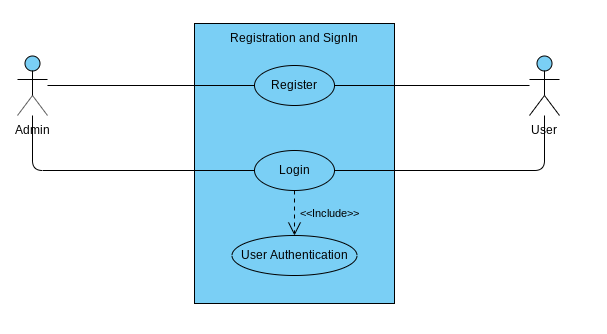
**Software Specification**

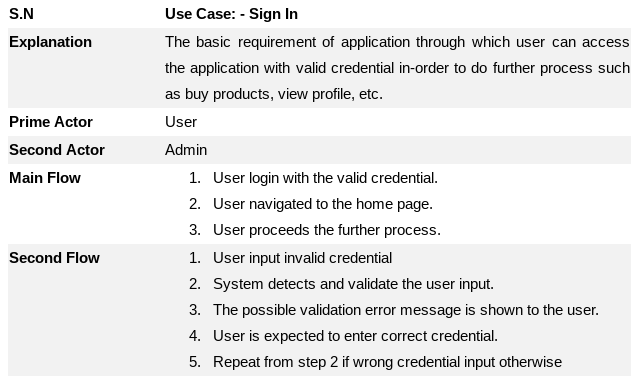
* OS: Windows 7 or higher or any Linux distro or Mac OS
* Browser: Google Chrome v74 or Mozilla Firefox or Microsoft Edge
* Server: XAMPP 7.3.2
* Database: MySQL
* IDE/Editor: Visual Studio Code

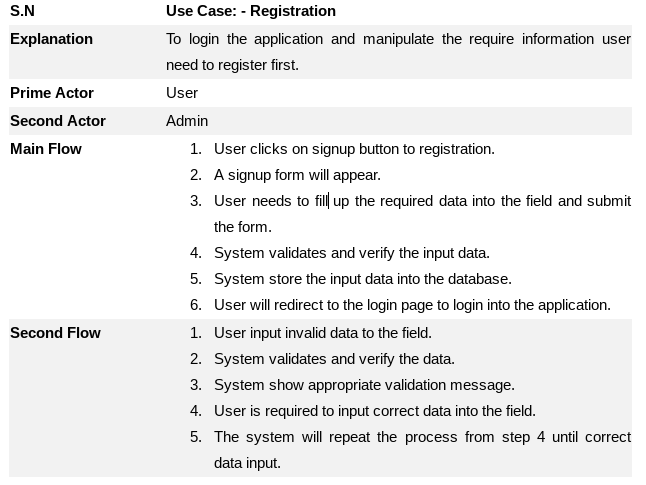
## 2.4.5: Use-case Diagram

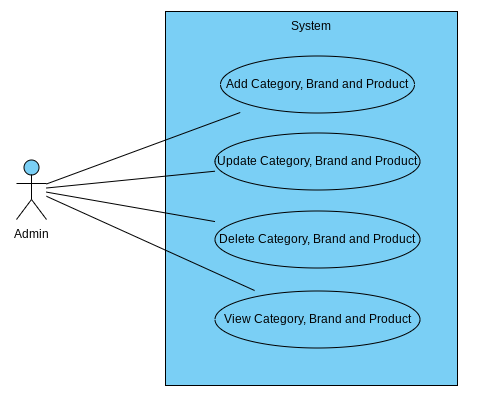
An Use-case diagram is a behavioral diagram that represents the systems interaction with the user (known as actors).

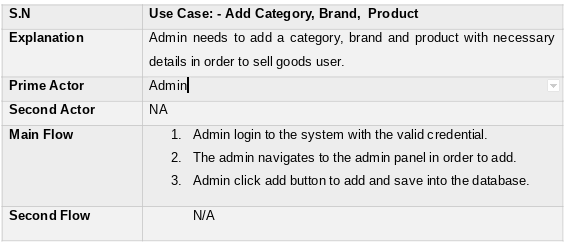
The diagrams below displays the use-case for Pasale.com where two different actors access the system with different rights.

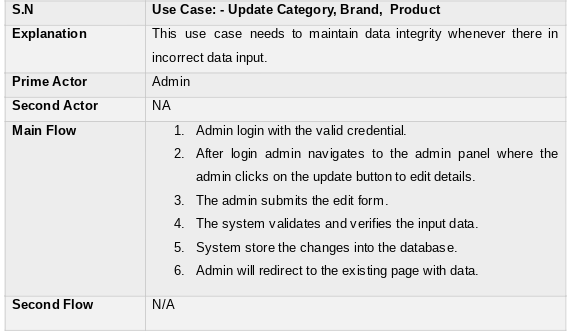


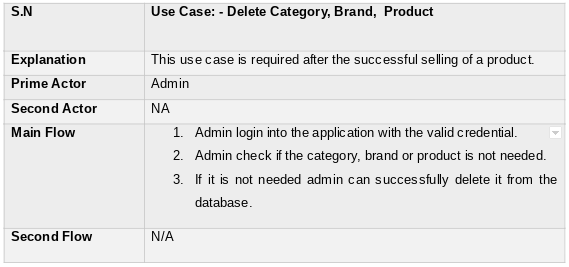


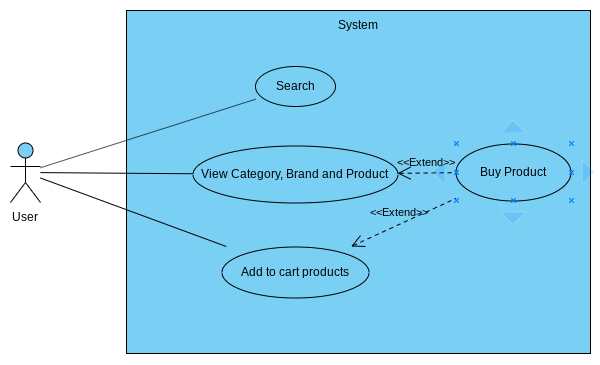


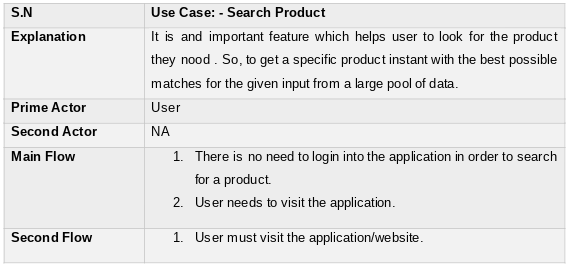


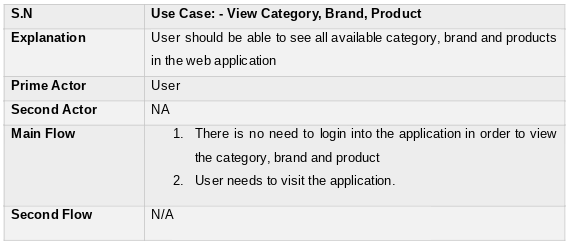


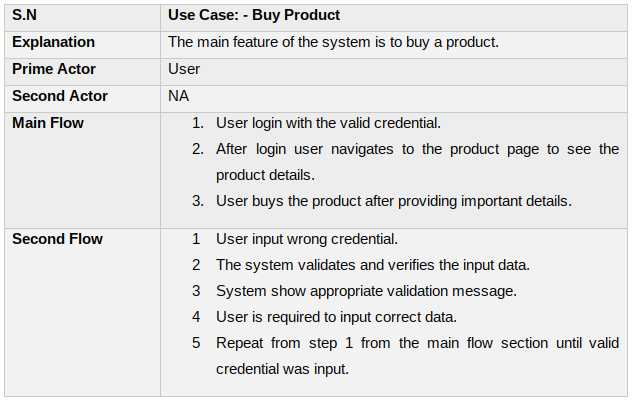


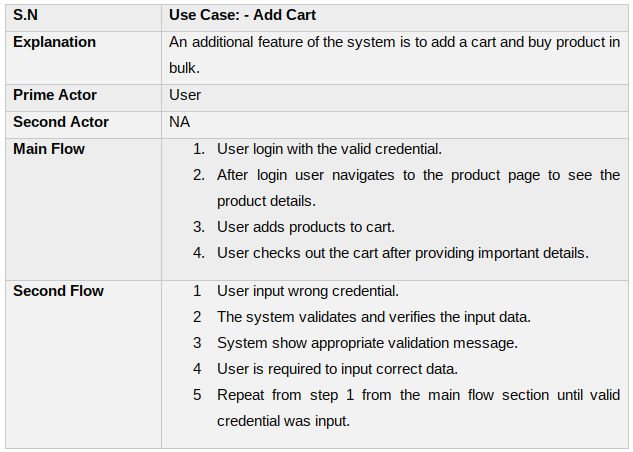


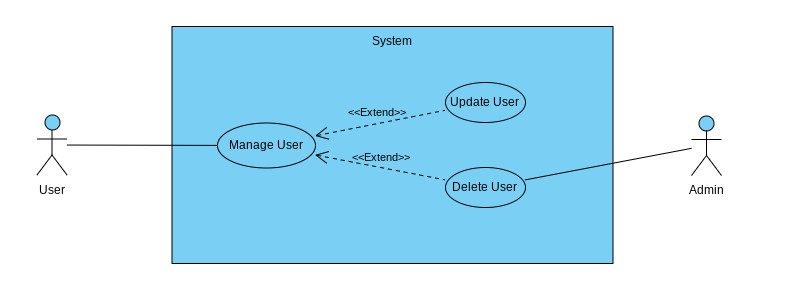


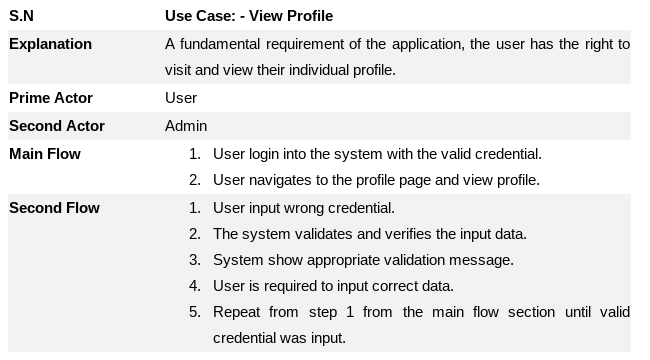


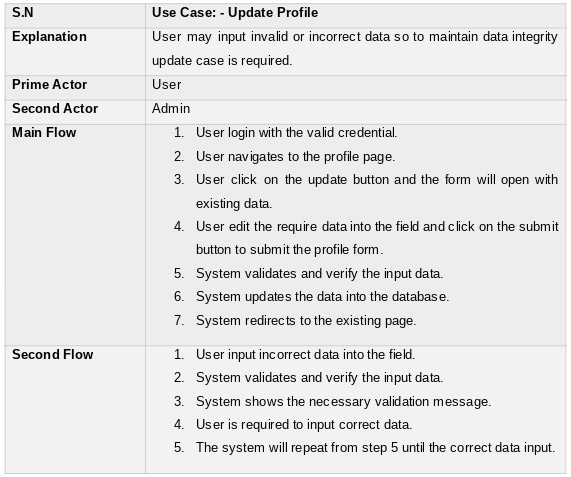


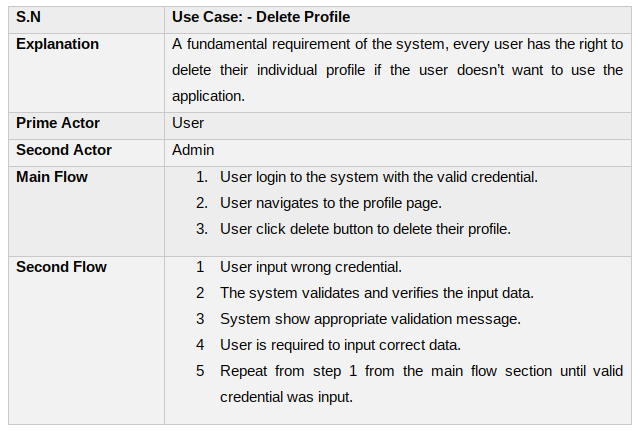












## 2.4.6: Initial Class diagram

### 2.4.6.1: Natural Language Analysis (NLA)

Pasale.com is an online multi-vendor ecommerce platform. Different vendors can sell their products through this platform. All the products are separated in different categories and they also represent brands if they are. Administrator should be able to add all the things from the admin panel. Users should be able to search for the products they need. User should be able to create a cart and buy the products in bulk. User should also be able to update their information if any needed.

The nouns and verbs are separated from above scenario to further help in finding required classes and methods.

**Nouns**: Product, Category, Brand. Cart, Vendor, User, Admin

**Verb**: sell, buy, update, delete, create, insert

After filtering the nouns for repetition and ambiguity, the candidate class and all the potential methods are generated

**Candidate** **Class**

|  |  |  |
| --- | --- | --- |
| Category | Brand | Product |
| Cart | User | Admin |

**Potential** **Methods**

|  |  |  |
| --- | --- | --- |
| Sell products | Buy products | Insert category, brand, product |
| Create Cart | Update category, brand, product | Delete category, brand, product |
| Search |  |  |

### 2.4.6.2: Initial Class Diagram

