**Table of Contents**

[**1. Introduction** 2](#_Toc505603197)

[**2. Analysis** 3](#_Toc505603198)

[**3. System Requirements Specification (SRS)** 5](#_Toc505603199)

[**3.1. Functional Requirement** 5](#_Toc505603200)

[**3.2 Non-Functional Requirement** 8](#_Toc505603201)

[**3.3. Prioritization** 11](#_Toc505603202)

[**3.4. Hardware and Software requirements** 12](#_Toc505603203)

[**4. Use Case Diagram** 12](#_Toc505603204)

[**5. System Architecture** 24](#_Toc505603205)

[**6. Initial Class Diagram** 26](#_Toc505603206)

[**7. Conclusion** 27](#_Toc505603207)

[**8. References** 28](#_Toc505603208)

# **1. Introduction**

The analysis is the organized examination and valuation of the data or information, by separating it into its component parts to find their interrelationships. All the component parts are researched in order to know what they do and how they relate to one another.

The analysis phase is where the project lifecycle begins. Gathering Requirements such as end-user requirements, quality requirements and requirement analysis and setting the overall direction for work that needs to be done are the key activities focused on the analysis phase. It helps to know the overview of what a system will do and find the risk and reward and execute the trading as a business(risk<reward). It helps to gain the knowledge and safeguard our investments and make profits. It helps in problem-solving and decision making. (EQSIS, 2015)

A rich picture is presented below that helps for open discussion and broad, share an understanding of the proposed project for Goshali Khadyanna Pasal.

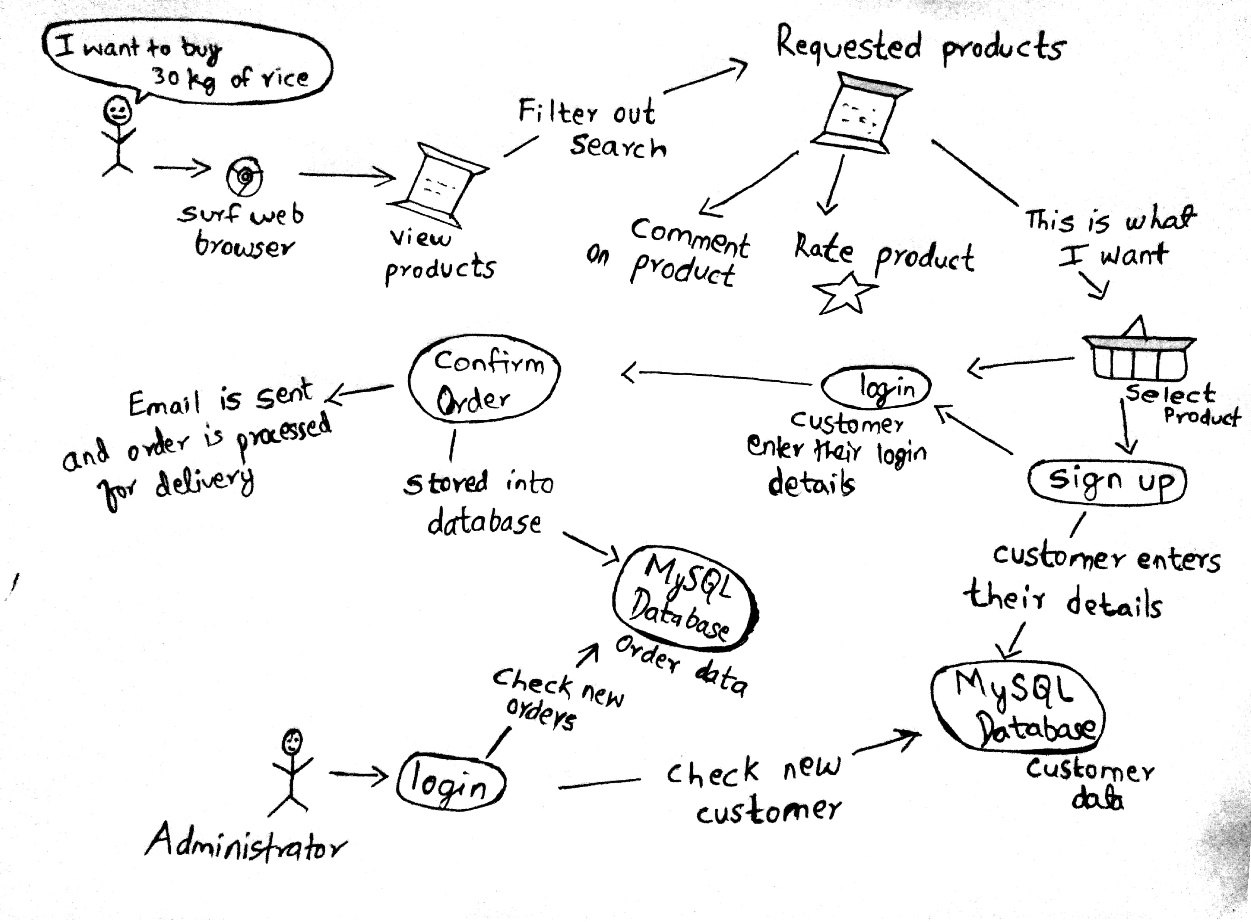


Figure 1: Rich Picture of Goshali Khadyanna Pasal.

In this report, description of the analysis phase is given in section 2, system requirement specification such as functional and non-function, hardware and software requirement is provided in section 3, use cases diagram and class diagram are presented in section 4 and 6, respectively, and system architecture is described in section 5.

# **2. Analysis**

If a project is done without doing analysis, it might not meet client’s needs and expectation. So, a thorough study and research of all the components that will be used on that project must be done in the early phase, which is called analysis. The analysis is the mental activity that asks how something does what it does and why it is as it is. By doing so, we are aware of the problem and it will be helpful in decision-making. So, the analysis is necessary for any project development.

The analysis is performed by identifying the problem statement. It is a document that explains what customers want. A focus is set up among the analyst to build the understanding of the system. After that, problem domain is identified which is the actual problem that must be solved. Problem domain helps to express the project scope. At the end, a good and accurate understanding of how the system is supposed to work is gained.

There are different types of analysis where each one is used for a specific problem and they are narrative analysis, descriptive analysis, socio-historical analysis, comparative analysis and SWOT analysis. Among these, SWOT analysis is used in this project. It identifies the strengths, weaknesses, opportunities, and threats of the system. Strengths and opportunities act as main assets and weaknesses and threats help to find problems and solve them.

There are different methodologies used in analysis phase such as organization-oriented, combined, object-oriented and so on. In this project, the object-oriented analysis is used. It is a process that groups items that interact with one another, typically by class, data or behavior, to create a model that represents the expected system. The system is decomposed into units which are usually capable to do a specific task. These units are more easily maintained and reused in the future. There are three object-oriented analysis techniques that communicate with each other and they are:

* Object Modelling
  + It describes the static structure of the system through the class diagram.
  + It identifies objects and classes into which the objects can be grouped into.
  + It identifies the relationships between the objects.
  + It identifies object attributes and operations that need to be performed.
* Dynamic Modelling
  + It describes how an individual object responds to events, either internal events caused by other objects, or external events caused by the outside world.
  + It identifies events and states of the objects.
  + It constructs dynamic model diagram such as state transition diagrams.
* Functional Modelling
  + It describes the processes between an object and how the data changes as it moves between operations.
  + It states why the operations are carried out.
  + It constructs data flow diagrams showing functional dependencies.
  + It identifies all the inputs and outputs.

Advantages of using Object-Oriented analysis.

1. It promotes encapsulation and data hiding technique that helps the developer to develop a system that cannot be altered by other parts of the system.
2. Easy management of the system complexity with the help of modularity.
3. Easy to understand, more flexible and easier to make an update in response to changing user requirements.
4. It promotes reusability of the modular parts. So, the system is usually built at a lower cost. (Tutorials Point, 2017)

There are various techniques for gathering information and requirements of the project and they are:

1. One-on-One interviews
2. Group Interviews
3. Questionnaires
4. Surveys
5. Joint Application Development (JAD)
6. Rapid Application Development (RAD)
7. Prototyping
8. Brainstorming

Among these requirements gathering techniques, one-on-one interviews, surveys, questionnaires, and brainstorming is used in this project.

# **3. System Requirements Specification (SRS)**

A System Requirements Specification is a set of documentation that describes the features and behavior of a system. It defines all the intended functionality required by the client and also defines a high-level business process. It also describes what simplifying assumptions have been made and what key performance will need to be met by the system. It is produced because it helps to put a common idea among the development team to translate the business idea into a software product. It eases the development process. Without SRS, it will be very difficult to accomplish the project goals. A product might be produced but won’t meet the clients’ requirements. A system requirements specification ensures that the clients’ and developers are aligned with what needs to be built. (Inflectra, 2016)

A full description of the functional requirements, non-functional requirements, and hardware and software requirements are presented in more detail below.

## **3.1. Functional Requirement**

A functional requirement is a process of identifying and describing the intended functions that should be accomplished by the system. It is a form of document that explains the expected types of inputs and outputs. It can be related to calculations, technical details, data manipulation and processing which defines what a system is supposed to accomplish. (techopedia, 2016)

Below provided tables show the functional requirements of “Goshali Khadyanna Pasal”.

**Rr1:** Customer Registration

**Description:** Any customer can register their accounts in this web-application and become a member of the website.

**Input:** Name, Address, Email and Password

**Output:** New account will be created.

**Rational:** To store the data of valid customers only.

**Dependencies:** N/A

**Rr2:** Login

**Description:** Any customer can login to access this web-application through their email and password.

**Input:** Email and password

**Output:** Successful login

**Rational:** To provide services provided by this web-application.

**Dependencies:** R1

**Rr3:** Add products

**Description:** Admin should be able to add new products.

**Input:** Product name, price, description, and image file.

**Output:** Added successfully

**Rational:** To add new products.

**Dependencies:** R2

**Rr4:** Edit products

**Description:** Admin should be able to update the information of the products.

**Input:** Product information

**Output:** Edited successfully

**Rational:** To edit the information of the products.

**Dependencies:** R3

**Rr5:** Delete products

**Description:** Admin should be able to delete the products.

**Input:** N/A

**Output:** Deleted successfully

**Rational:** To delete the unwanted products.

**Dependencies:** R3

**Rr6:** Display products

**Description:** Registered customer and un-registered customer should be able to view the detail information of the products.

**Input:** N/A

**Output:** Product list

**Rational:** To display the products that are for sell.

**Dependencies:** R3

**Rr7:** Search products

**Description:** Registered customer and un-registered customer should able to search the products.

**Input:** Product name

**Output:** Required product details

**Rational:** To find the products that meet customers need.

**Dependencies:** R6

**Rr8:** Order products

**Description:** Only registered customer should be able to order the products.

**Input:** Products name

**Output:** Ordered Products

**Rational:** To order products.

**Dependencies:** R6

**Rr9:** Rate and comment products

**Description:** Only registered customer should be able to rate and comment on products.

**Input:** ratings and comments

**Output:** rated and commented successfully

**Rational:** To rate and comment on the products.

**Dependencies:** R8

**Rr10:** Subscribe products

**Description:** Registered customer should be able to subscribe products.

**Input:** Product name

**Output:** Subscribed product list

**Rational:** To send the subscribed products in a regular interval of time.

**Dependencies:** R6

**Rr11:** Calculate calorie value

**Description:** Each product has a specific calorie value. Total calorie value of the ordered products will be calculated.

**Input:** Products name

**Output:** Ordered Products with total calorie value

**Rational:** To inform customers that they are consuming this much calorie in an order.

**Dependencies:** R8, R10

**Rr12:** Change customer password

**Description:** Registered customer should be able to change their password.

**Input:** Password

**Output:** Password changed successfully

**Rational:** To provide customer data security.

**Dependencies:** R2

**Rr13:** Logout

**Description:** Registered customer and admin should be able to logout from this web-application.

**Input:** N/A

**Output:** Logout successfully

**Rational:** To restrict access to unauthorized person.

**Dependencies:** R2

## **3.2 Non-Functional Requirement**

Non-functional requirements are the quality attributes such as security, reliability, performance, maintainability, scalability, and accuracy. It ensures the usability and effectiveness of the system. It does not deal with the features and capabilities of the system but with the required characteristics of the system. It helps to deliver a value to the customer and much of the time and effort is given in this non-functional requirement. Without these quality attributes, the system may fail to satisfy business, client and market needs. (Scaled Agile, 2017)

A list of non-functional requirements of this project is given on the next page.

**Rq-non6:** Maintainability

**Description:** System must be able to inspect problems and possible problems and be able to restore services even after a failure.

**Rational:** To make the failed system capable to restore into its normal operable state.

**Dependencies:** Q-non5

**Rq-non5:** Recoverability

**Description:** System must have a backup plan so that it can recover in any problematic situation such as damage files.

**Rational:** To maintain availability of the data.

**Dependencies:** N/A

**Rq-non4:** Reliability

**Description:** The result of calculation, measurements, functions must have a quality of being trustworthy. Customers and admin both should be able to rely on this system.

**Rational:** To eliminate the problems

**Dependencies:** Q-non7

**Rq-non3:** Availability

**Description:** System must be always accessible to the customers.

**Rational:** To maintain the system availability to the customers.

**Dependencies:** Q-non4, Q-non5

**Rq-non2:** Scalability

**Description:** The web-application should be able to withstand large number of data.

**Rational:** To make this system work properly even on the increase in size or volume.

**Dependencies:** Q-non4

**Rq-non1:** Performance

**Description:** The web-application should have fast response time and fast data throughput.

**Rational:** To make this system work properly and accurately.

**Dependencies:** Q-non2

**Rq-non10:** Legal

**Description:** System should work in a legal parameter.

**Rational:** To provide legal security to the system.

**Dependencies:** N/A

**Rq-non9:** Usability

**Description:** System should be able to use and learn easily.

**Rational:** To increase the number of customers.

**Dependencies:** R1

**Rq-non8:** Environmental

**Description:** System must be socially accepted.

**Rational:** To encourage more customers to use the website.

**Dependencies:** N/A

**Rq-non7:** Security

**Description:** System must be able to encrypt the message while transferring and retrieving. It must maintain the data integrity.

**Rational:** To secure the customers confidentiality.

**Dependencies:** R1, R2

## **3.3. Prioritization**

Prioritization is the process of evaluation to a collection of functional and non-functional requirements and classifying them according to their importance. We must prioritize the functional and non-functional requirements based on their impact on the overall system. Time and money will be properly utilized by prioritizing the requirements. It helps to differentiate between immediate and long-term requirements of the system.

There are various types of prioritization techniques such as MoSCoW, Needs-based analysis, crowdsourcing, and opportunity scoring. Among these techniques, we are going to use MoSCoW for prioritizing the requirements. It was developed by Dai Clegg in 1994. According to this technique, the requirements are categorized into following groups.

* M – Must have. This point describes those requirements that play an important role in the system. So, it must be fulfilled in the final system.
* S – Should have. A high priority feature that is not crucial to launching but is considered to be important and of a high value to users.
* C – Could have. Requirements that is desirable but not necessary. It can be removed if the time scale of the project is at risk.
* W – Won’t have. The requirement will not be implemented in a current release but may be included in a future stage of development. (Kukhnavets, 2016)

Prioritization of functional and non-functional requirements of this project are given below.

|  |  |  |
| --- | --- | --- |
| **ID** | **Functional Requirement** | **MoSCoW** |
| Rr1 | Customer registration | Must have |
| Rr2 | Login | Must have |
| Rr3 | Add products | Must have |
| Rr4 | Edit products | Should have |
| Rr5 | Delete products | Should have |
| Rr6 | Display products | Must have |
| Rr7 | Search products | Must have |
| Rr8 | Order products | Must have |
| Rr9 | Rate and comment products | Must have |
| Rr10 | Subscribe products | Must have |
| Rr11 | Calculate calorie value | Must have |
| Rr12 | Change customer password | Should have |
| Rr13 | Logout | Must have |

|  |  |  |
| --- | --- | --- |
| **ID** | **Non-Functional Requirement** | **MoSCoW** |
| Rq-non1 | Performance | Must have |
| Rq-non2 | Scalability | Could have |
| Rq-non3 | Availability | Must have |
| Rq-non4 | Reliability | Must have |
| Rq-non5 | Recoverability | Must have |
| Rq-non6 | Maintainability | Must have |
| Rq-non7 | Security | Must have |
| Rq-non8 | Environmental | Should have |
| Rq-non9 | Usability | Must have |
| Rq-non10 | Legal | Must have |

## **3.4. Hardware and Software requirements**

* Hardware requirements: 8 GB RAM, 64-bit processor, hard disk
* Software requirements: NetBeans IDE, xampp, ProjectLibre, word document software, and StarUML.

# **4. Use Case Diagram**

Use case diagram is the pictorial representation of the set of functionalities of the system. It is a Unified Modelling Language (UML) diagram that provides a standard way to visualize the design of the system. It also illustrates the behavior and the dynamic aspects of the system. It helps to organize, identify, and clarify the system requirements by using notations and those notations are mentioned below.

* Boundary – This component defines the system of interest.
* Actors – Individuals who are involved in the system and their roles are defined.
* Use cases – This component defines the precise requirements or functions of the system performed by each actors’ present.
* Relationship – This component put a clear concept of the relationship between the actors and the use cases.

Some use cases tend to have some clearly defined subtasks. In this case, <<extends>> or <<include>> is used to indicate the subtasks.

*Benefits of using Use case diagrams.*

* It breaks requirements down into shorter scenarios that stakeholders can easily understand the problem that needs to be solved.
* It focuses on the users too not only to the system, so the real system needs are defined earlier.
* By making understandable by the stakeholders during the early planning of the project, the problems will be eliminated when the system will deploy.
* It also helps to define other aspects of software development such as planning (cost, complexity, and time estimates), test case definitions, and user documentation.

Use case diagrams of this project are given on the next page.

***Administration System***

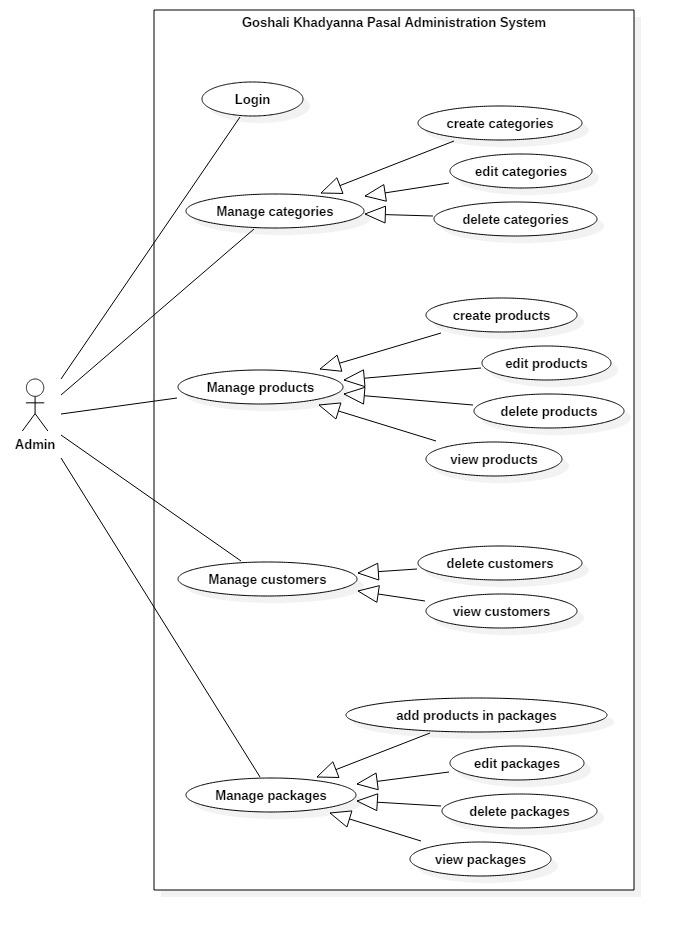


Figure 2: Administration system Use case diagram.

*Add products*

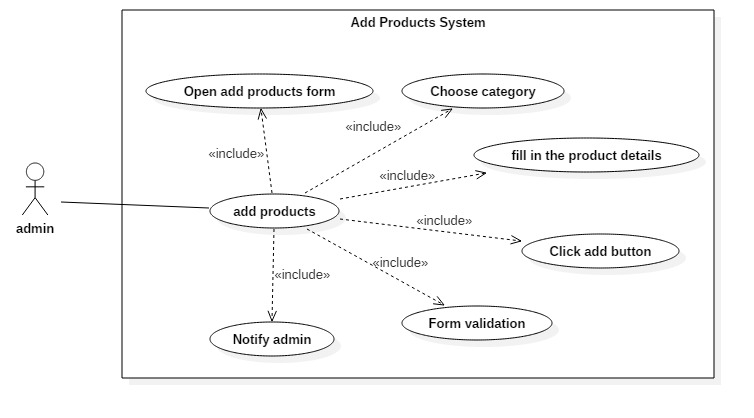


Figure 3: Add product Use case diagram.

*Add products -Scenario description*

1. Admin opens the add products form.
2. Admin chooses the category to which the product will be added.
3. Admin fills the product details in the add products form.
4. Admin click adds button.
5. The system checks whether the supplied important text field in the form is null or not.
6. The system adds products and notifies the admin.

*Create package*

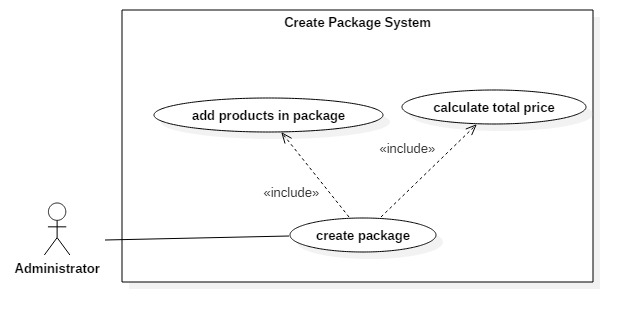


Figure 4: Create package Use case diagram.

*Create package – Scenario description*

1. Admin adds the products inside the package.
2. The total quantity of the products and its price is calculated by the system.

***Customer System***

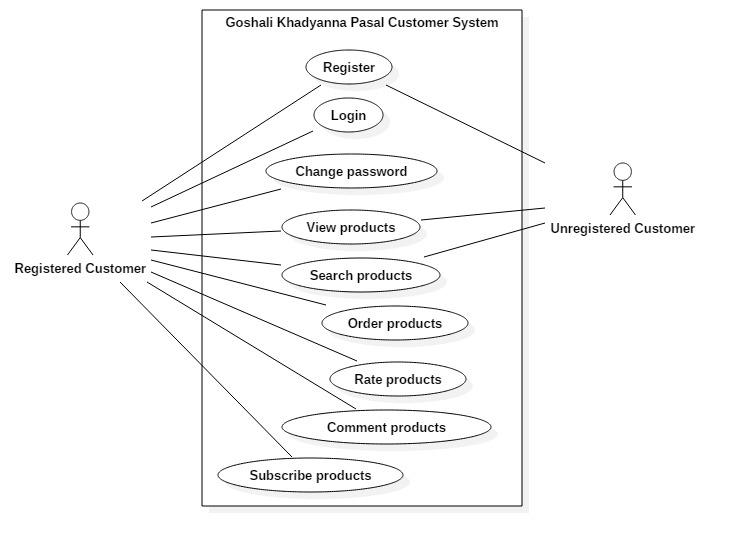


Figure 5: Customer system Use case diagram.

*Registration*

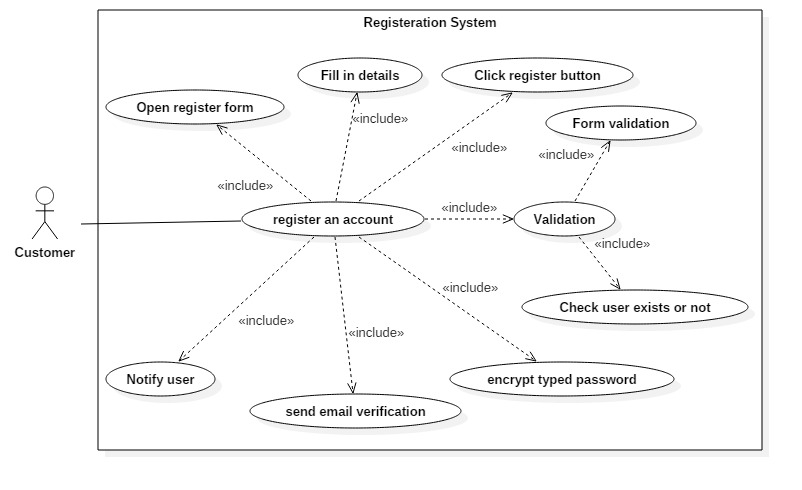


Figure 6: Registration system Use case diagram.

*Registration – Scenario description*

1. Customer opens the registration form.
2. Customer fills their personal information on the registration form.
3. Customer clicks the register button.
4. The system validates the information provided by the customer. One is it validates the empty text field of the form and check whether the provided customer name already exists in the database or not.
5. The system encrypts the password of the customer while storing.
6. The system sends email verification to the customer and notify the customer

*Login*

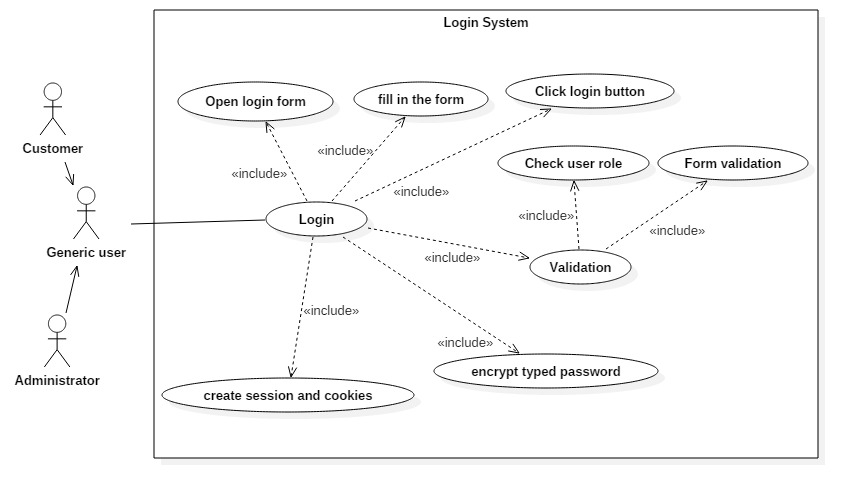


Figure 7: Login system Use case diagram.

*Login – Scenario description*

1. Generic user (Admin and Customer) opens the login form.
2. The generic user fills in the login form details.
3. Generic user clicks the login button.
4. The system validates that includes the null value validation and check the generic user role.
5. The system will encrypt the typed password while transferring from browser to server.
6. The system will create a session and cookies for the customer.

*Change password*

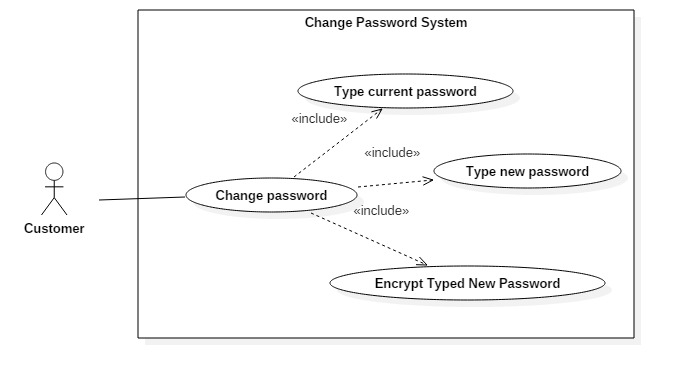


Figure 8: Change password Use case diagram.

*Change password – Scenario description*

1. Customer types the current password.
2. Customer types the new password.
3. The system encrypts the new password while storing and transferring and change the password.

*Search products*

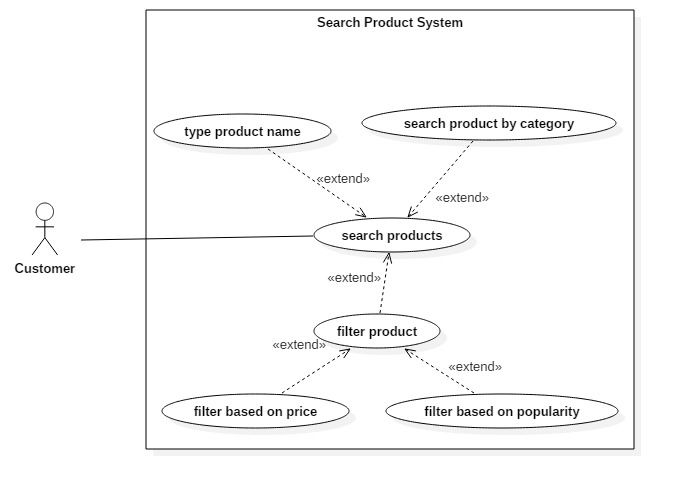


Figure 9: Search product Use case diagram.

*Search product – Scenario description*

There are various ways to search products and they are:

1. Customer types the product name.
2. Customer search product by category.
3. The system filters products based on price and popularity.

*Order product*

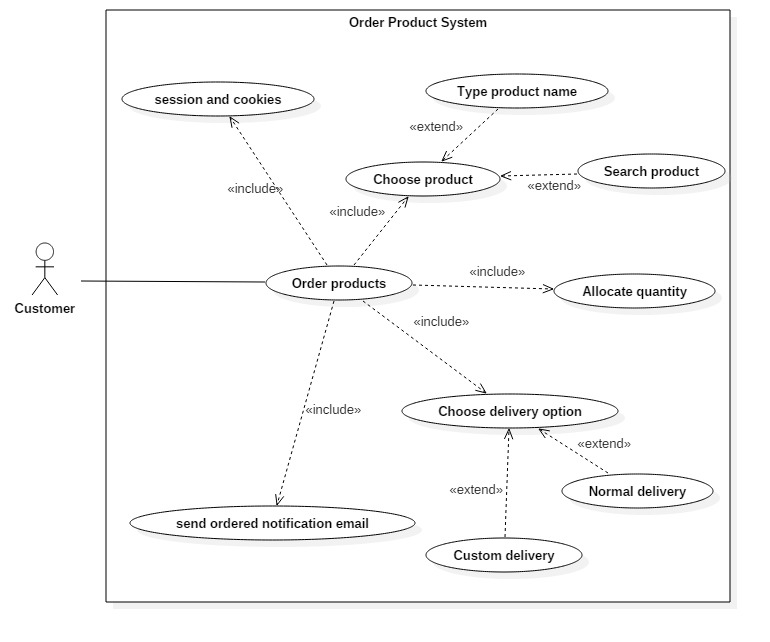


Figure 10: Order product Use case diagram.

*Order products – Scenario description*

1. First, a session or cookies is needed to set up.
2. Customer chooses products by typing the product name or by searching the product.
3. Customer allocates the quantity.
4. Customer chooses the delivery option where one is custom delivery and another is a normal delivery.
5. The system sends the ordered notification email to the customer.

*Product feedback*

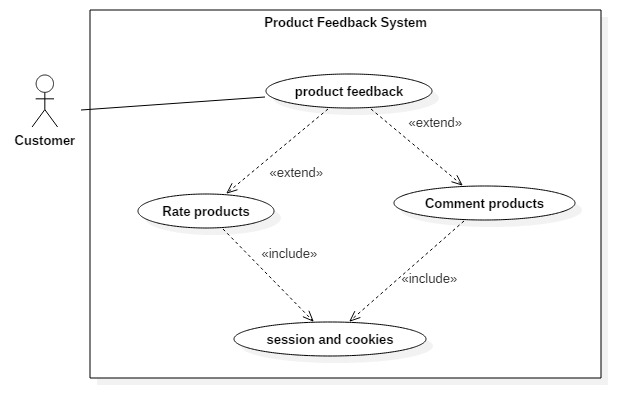


Figure 11: Product feedback Use case diagram.

*Product feedback – Scenario description*

1. Product feedback can be given by the customer in two process where one is rating and another is commenting.
2. Session and cookies of the customer are required to both, rate and comment products.

*Subscribe product*

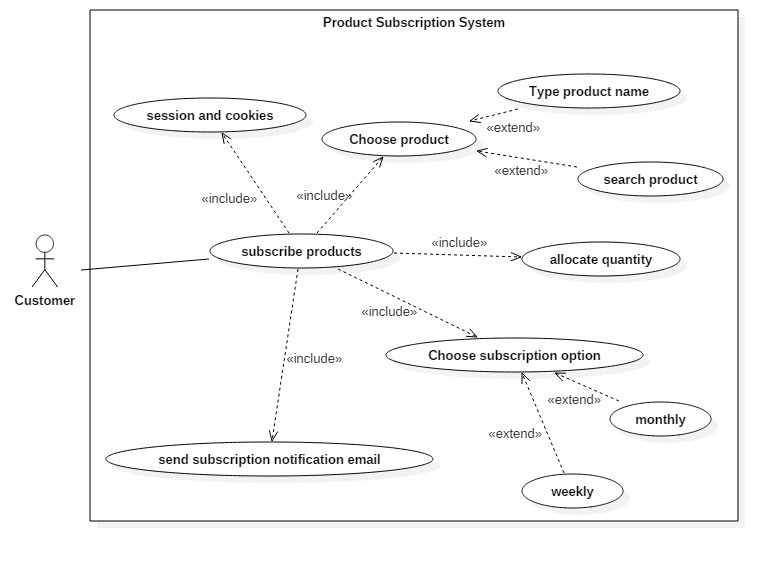


Figure 12: Subscribe product Use case diagram.

*Subscribe product – Scenario description*

1. Session and cookies is required to subscribe products.
2. Customer chooses the product by typing the product name or by searching the product.
3. Customer allocates the quantity of the products.
4. Customer chooses the subscription option where one in weekly subscription and another is a monthly subscription.
5. The system sends subscription notification email to the customer.

# **5. System Architecture**

System Architecture is the process of establishing a concept that describes the structure and behavior of the proposed system. It acts as the backbone for the proposed system. It provides a preview of the proposed system. System Architecture is also considered to be a blueprint for the proposed system. Blueprints help developers ensure that what they are planning to develop will work and provide the required purpose. Blueprints are used to understand the proposed system before developing it.

We need to build architecture before developing a system and its reasons are given as follows.

*Communication among stakeholders*

* Stakeholders will be able to understand the proposed system by viewing its architecture. It helps in negotiating, consensus, and communicating the system among the stakeholders.
* The architecture includes the description of all the elements and how the elements communicate with each other to carry out the required behavior which will help new project members to understand the system.

*Formulates the earliest set of design decisions*

* By developing the architecture of the system in the earliest phase of the system development, we can predict the system qualities. Early analyzing produces the need quality of the system and by analyzing the architecture of the system we can determine whether a system will be able to show its quality attributes.
* The architecture provides a template for how elements will interact with each other. So, it provides template-based development.
* It allows for accurate time and cost estimation. The requirement of the system has been reviewed and validated which means there will be more knowledgeable about the proposed system. Thus, it will be easy while estimating.

*Supports Reusable and Transferable*

* Reuse at architectural level is very beneficial. We can reuse the architecture of a previous system in a similar new system. Some common requirements of the system at architectural level is predetermined. It is also very easy in building the reused architecture. Thus, it can be transferred to another system with similar requirements. (Electrical and Computer Engineering, 2016)

In this proposed system, MVC architecture will be used. MVC stands for Model, View, and Controller.

*Model:* It illustrates the shape of the data. It also represents the business entity of the application. It is responsible for managing the data. It responds to the request from the view and also responds to instructions from the controller to update itself.

*View*: It is a user interface where users are able to view the data from the model and also modify the data. It is produced through the use of HTML.

*Controller*: It handles all the user request. User requests are obtained from the view that invokes an appropriate URL request and it is handled by the controller. And then, the controller will provide the data to the view as a response.

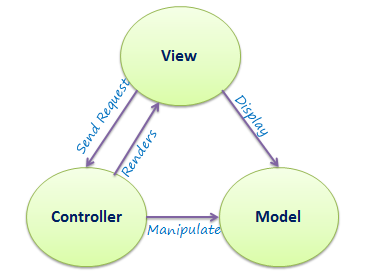


Figure 13: MVC Architecture.

We choose to use MVC for separation of concerns, ease of maintainability, and extensibility of an application.

*Separation of concern*: MVC separates the presentation and data-processing content from an application. Views are responsible for presentation form to the user which does not have to concern about the request coming from the controller. Model is independent of view and controller which holds the business entities that is passed to the view by the help of the controller. The controller is also independent of views and model which is responsible for handling the user requests and pass it on as per the routes defined.

Due to the separation of concern in MVC architecture, it is very easy to maintain all the constituent parts of the application. We can easily find the error and can fix it. Also, it is very flexible. New updates and upgrades can be easily implemented in the application.

*Benefits of using MVC architecture*

* Code complexity is reduced.
* The code is reused.
* Promotes flexibility.
* Decoupled code.

# **6. Initial Class Diagram**

The class diagram represents the static view of the project. It helps in visualizing and describing the characteristics, aspects, and feature of the project. Producing class diagram is also a documentation process that helps in maintenance and future upgrades.

It describes different aspects of the system such as the attributes and operations of classes, their visibility, and relationships. The main advantage of class diagrams is that it helps developers to convert the diagrammatic view into the executable code of the application.

The given class diagram is not a complete class diagram of the proposed system. It is the first draft class diagram. The second draft class diagram will be given on the design phase of the project, however, some of the potential main class and its attributes are mentioned in the figure below.

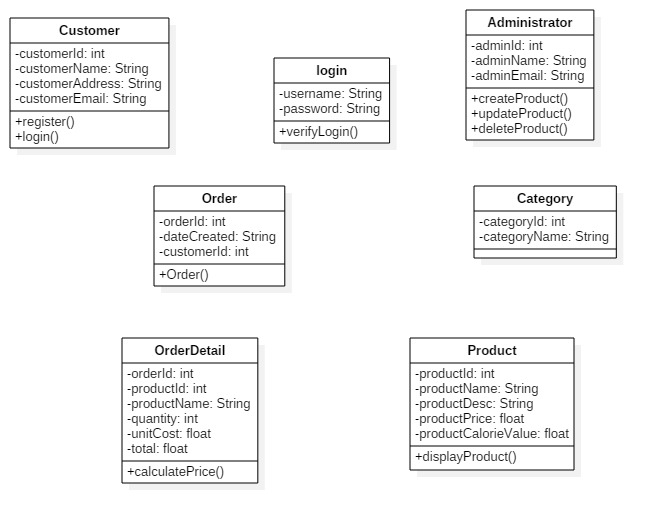


Figure 14: Initial class diagram.

# **7. Conclusion**

Thus, a full analysis specification is presented in this report. Analyzing is the very important task that needs to be performed before developing a system. It helps in identifying the possible error and bugs that might come across after the system is deployed. Problem is discussed and solved in the early phase of the development which is only possible through analysis.

Thus, a rich picture is presented to provide an overview of the proposed system. A full detailed description of analysis and its importance is written. The system requirement specification such as functional and non-functional requirements is identified and it is prioritize using MoSCoW method. The hardware and software requirements for the developing the system is identified. Use cases for administrator and customer are provided in Use case diagram where the main use cases scenario is described. MVC architecture will be used for developing the system and the first draft class diagram of the system is produced.

# **8. References**

Electrical and Computer Engineering, 2016. *Why is Software Architecture Important?.* [Online]   
Available at: http://www.ece.ubc.ca/~matei/EECE417/BASS/ch02lev1sec4.html  
[Accessed 1 February 2017].

EQSIS, 2015. *Why do we need to do an analysis?.* [Online]   
Available at: https://www.eqsis.com/pro/forums/topic/need-analysis/  
[Accessed 18 January 2017].

Inflectra, 2016. *What is a System Requirements Specification(SRS)?.* [Online]   
Available at: https://www.inflectra.com/ideas/topic/requirements-definition.aspx  
[Accessed 20 January 2018].

Kukhnavets, P., 2016. *MoSCow Method: the Most Successful Prioritization Technique For Any Project.* [Online]   
Available at: https://blog.ganttpro.com/en/prioritization-techniques-and-methods-for-projects-with-advantages-of-moscow-model/  
[Accessed 23 January 2018].

Scaled Agile, 2017. *Nonfunctional Requirements.* [Online]   
Available at: http://www.scaledagileframework.com/nonfunctional-requirements/  
[Accessed 22 January 2018].

techopedia, 2016. *Functional Requirement.* [Online]   
Available at: https://www.techopedia.com/definition/19508/functional-requirement  
[Accessed 21 January 2018].

Tutorials Point, 2017. *OOAD - Object Oriented Analysis.* [Online]   
Available at: https://www.tutorialspoint.com/object\_oriented\_analysis\_design/ooad\_object\_oriented\_analysis.htm  
[Accessed 19 January 2018].