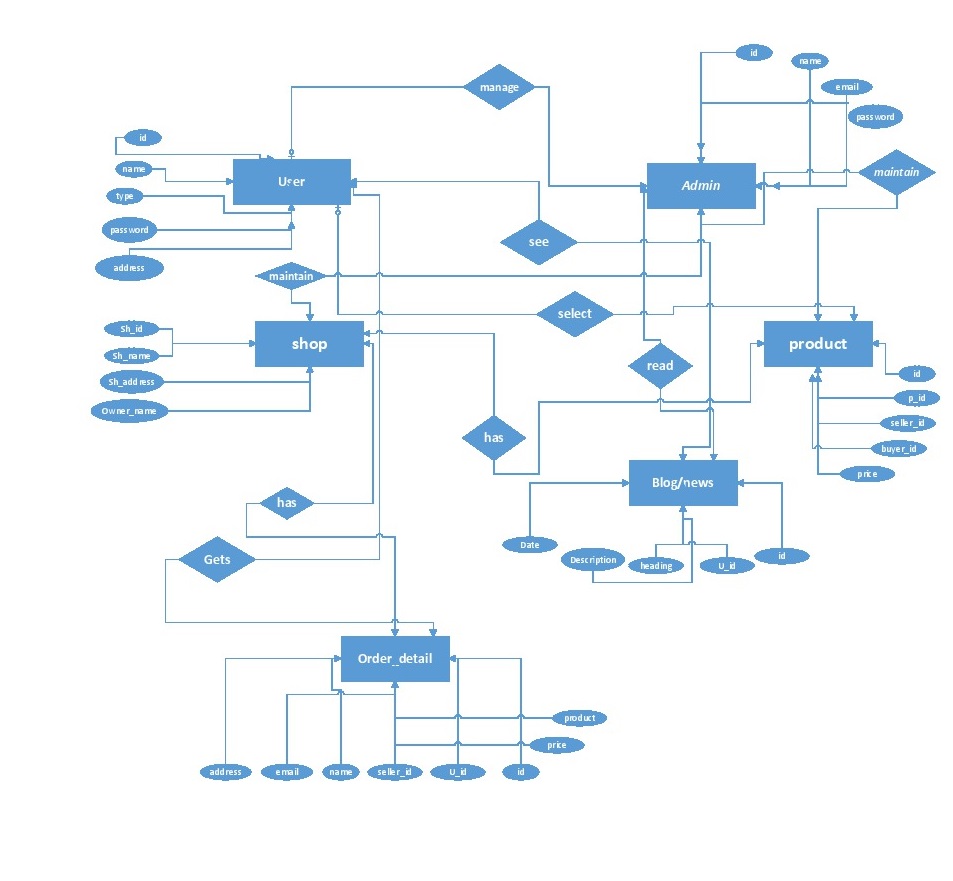
# Chapter 4

# System Design

# System Design

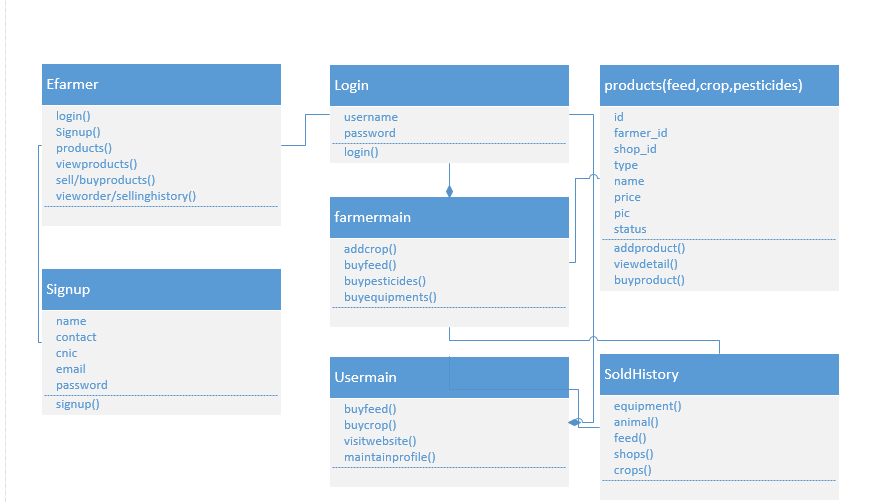
In this chapter, the architecture of the whole project is analyzed. System analysis is the process  
of defining the architecture, components, and data of a system to satisfy specified requirements. Design is a method of studying a system by examining its component parts and their interactions. Before implementation began the system was analyzed and designed. In this section, use cases, requirement analysis, and other part are described in details.

## Entity Relationship Diagram



#### Figure 4.1 ER Diagram

## Class Diagram



#### Figure 4.2 Class Diagram

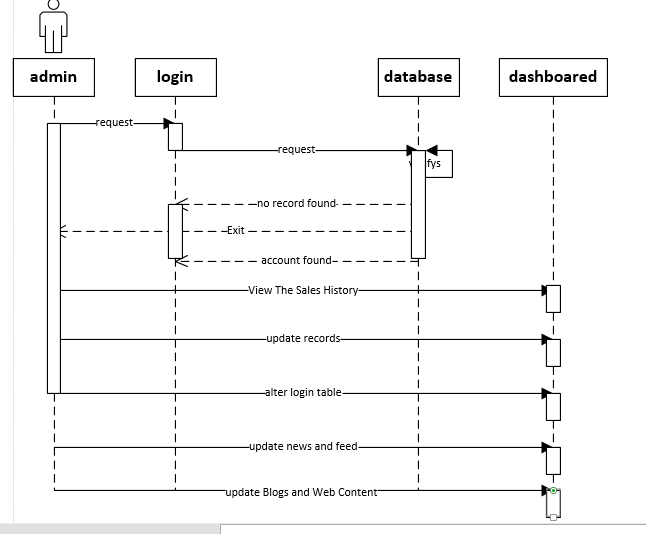
## Sequence Diagram

Sequence diagram uses concept of a Message-Sequence-Chart. It shows interactions of objects in a sequence of time. It shows the classes and objects involved in the scenario and the message sequence between the objects which is desired to carry out the functionality of a given scenario.

Sequence diagrams are usually related with the understanding of use case in the logical View of the system which is under development. “Sequence diagrams are sometimes called event diagrams, timing diagrams, event scenarios”. A parallel vertical line on sequence diagram is called lifeline.

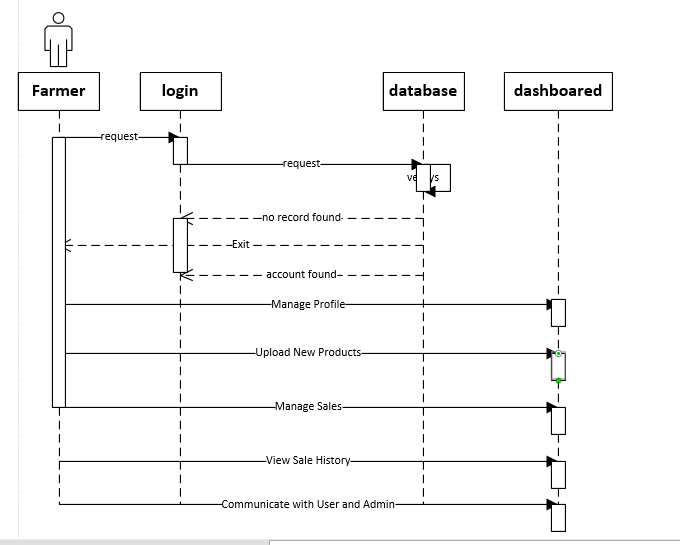
Different objects or processes that live simultaneously, and, on horizontal arrows, the messages exchanged, in the order in which processes occur. This allows some specification of some simple runtime scenarios in a graphical pattern.

### 4.3.1. Admin Sequence Diagram



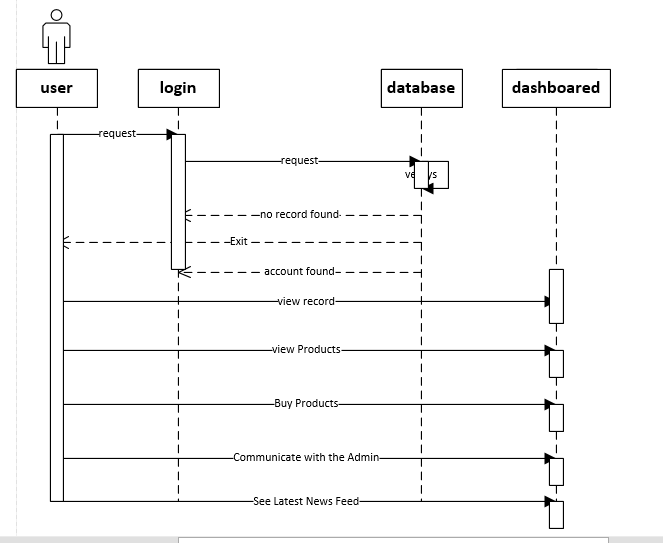
#### Figure 4.3.1 Admin Sequence

### 4.3.2. Farmer Sequence Diagram



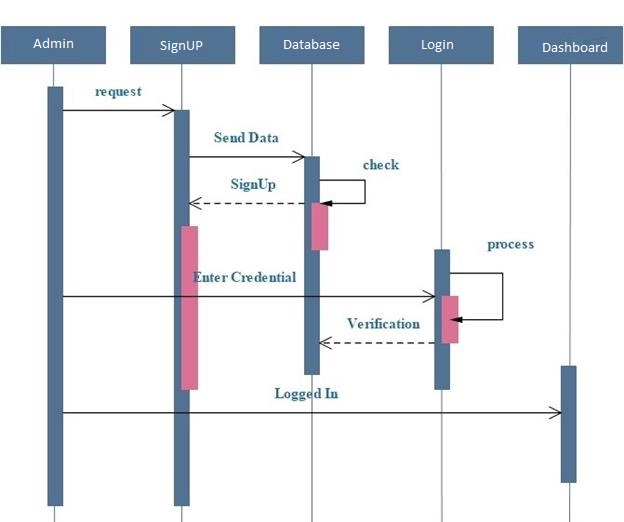
#### Figure 4.3.2 Farmer Sequence

### 4.3.3. User Sequence Diagram



#### Figure 4.3.3 User Sequence

### 4.3.4. Sequence diagram of User/Admin/Farmer Login:

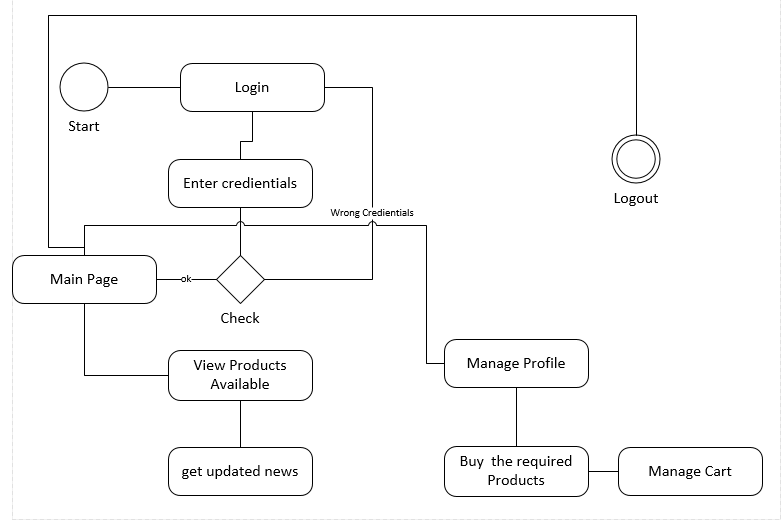


#### Figure 4.3.4 Sequence diagram of Login

## Activity Diagram

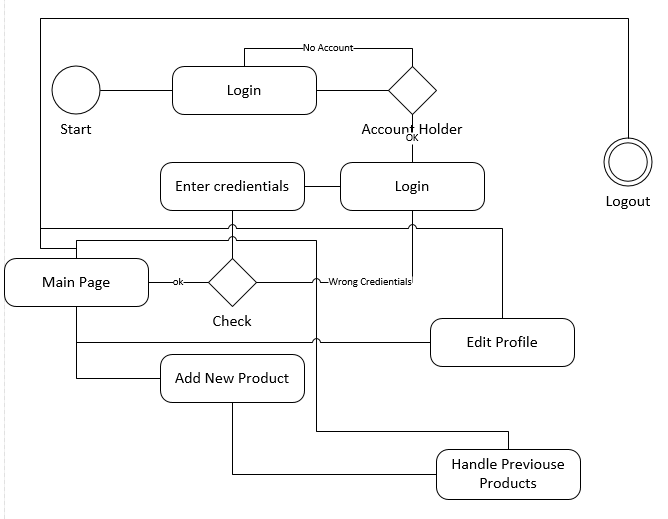
A diagram of the sequence of movements or actions of people or things involved in a complex system or activity.

### 4.4.1 User Activity Diagram



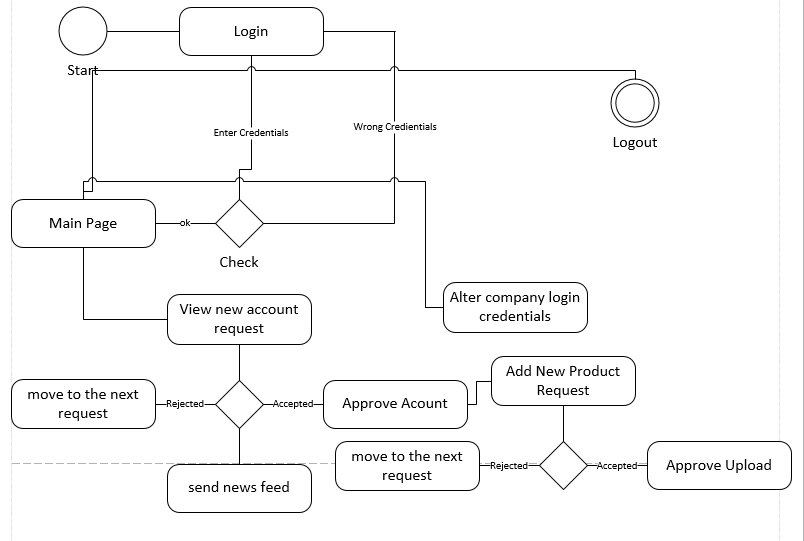
#### Figure 4.4.1 Activity User

### 4.4.2. Farmer Activity Diagram



#### Figure 4.4.2 Activity Farmer

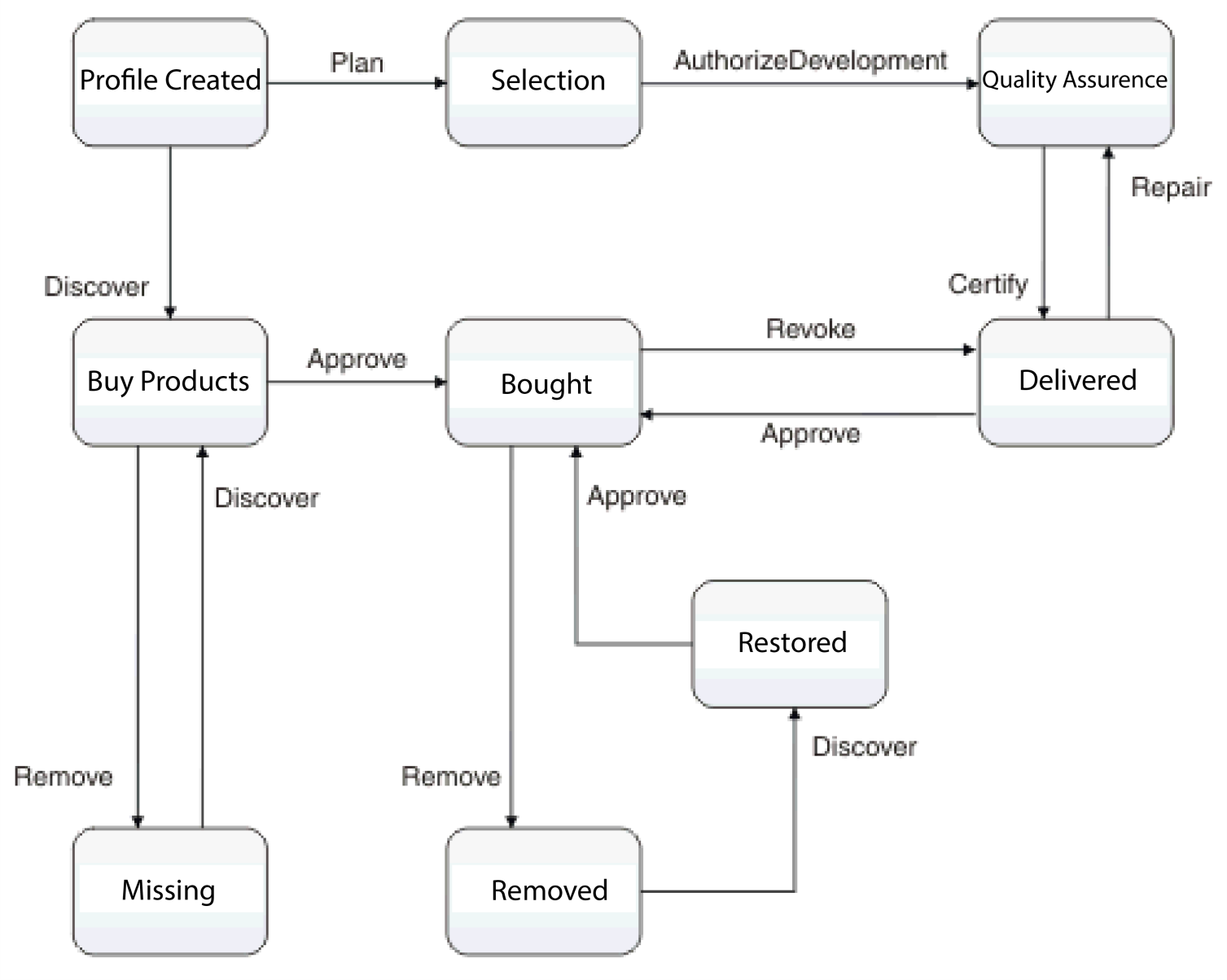
### 4.4.3. Admin Activity Diagram



#### Figure 4.4.3 Activity Admin

## 4.8. State Transition Diagram

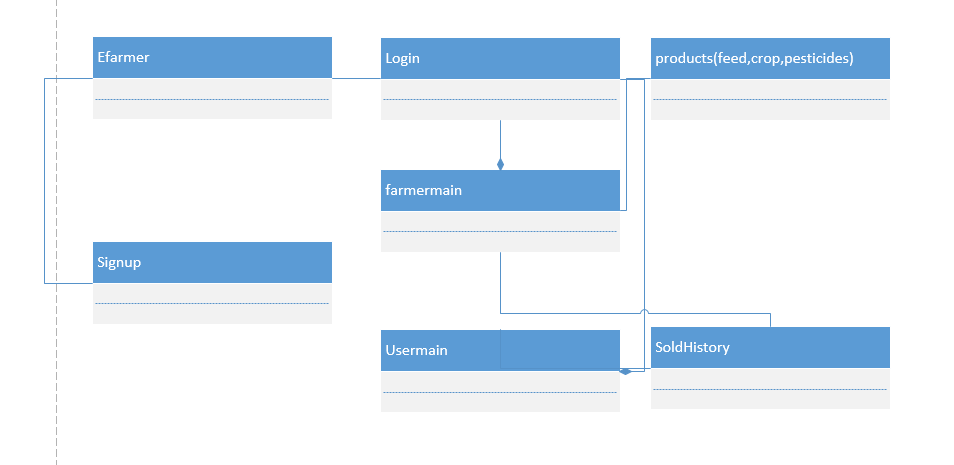
State-Transition Diagrams. State-transition diagrams describe all of the states that an object can have, the events under which an object changes state (transitions), the conditions that must be fulfilled before the transition will occur (guards), and the activities undertaken during the life of an object (actions). State-transition diagrams are very useful for describing the behavior of individual objects over the full set of use cases that affect those objects. State-transition diagrams are not useful for describing the collaboration between objects that cause the transitions.



#### Figure 4.8 State Transition Diagram

## 4.10 Domain Model Diagram

In software engineering, a domain model is a conceptual model of the domain that incorporates both behavior and data. In ontology engineering, a domain model is a formal representation of a knowledge domain with concepts, roles, datatypes, individuals, and rules, typically grounded in a description logic.



#### ­­ Figure 4.10 Domain Model Diagram