Ahsanullah University of Science and Technology

Department of Computer Science and Engineering

"BIPONEE"-An E-Commerce Web Application

Md. Toufiqul Islam 15.02.04.097
Susmoy Chakraborty 15.02.04.114
Nigar Sultana 15.01.04.002
Ovi Nazmul 13.02.04.081

23 October 2018

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Now a days E-commerce website is very popular in our country. It enables the customers to choose a product or service of their choice from anywhere in the country or the world. Online shops gives us the opportunity to shop 24x7 easily. Just a couple of clicks of the mouse, we can purchase shopping products very easily, which also saves our time. Besides, online shopping allows us to find many products that user wouldn't be able to find in a physical store.
Our project named "Biponee" is an E-Commerce store where there is a vast option of clothing, footwear, jewelry, accessories, electronics, appliance, books, restaurants, health beauty products etc. We are making this project for an online shop who are currently serving their service by a Facebook page only.

1.1 Project Goal

The goal of our project is to make it easier to sell product. Customer tracking, online payment system, customer log-in, order management, chatting with customer, cart system, efficient checkout system, promo code system, searching product is the key features of our project.
In addition there will be a section in our project where the admins can see the monthly buy and sell report of their shop.

1.2 Project Feasibility Analysis

1.2.1 Technical Feasibility

- Cost of computers, laptops, printers etc.
- Cost of domain, hosting.
- Internet quality, availability of internet.
- Cost of internet.
- 24x7 availability.

1.2.2 Operational Feasibility

- User Friendly GUI.
- Tracking the product.
- Monthly buy and sell report.
- Chatting system.

- Easily add or delete product.

1.2.3 Economical Feasibility

- Cost of Developers, Programmers etc.
- Cost of training the workers to operate with the system.
- Cost of Employee for product delivery.

1.3 Cost benefit analysis

Tangible cost is a quantifiable cost related to an identifiable source or asset. It can be directly connected to a material item used to conduct operations or run any business. Besides intangible cost is an unquantifiable cost relating to an identifiable source. Intangible costs represent a variety of expenses such as losses in productivity, customer goodwill, loss of brand value or damage to corporate reputation.

Tangible benefits are those measured in monetary terms and intangible benefits can not be measured in monetary terms but they do have a very significant business impact.

	Cost	**Benefit**
Tangible	-Direct Project Cost -Staff and office space -Implementation Cost	-Making Product order and delivery faster -Lower inventory cost -Improve the rate of productivity -Revenues from advertisers -Improved stock control
Intangible	-Loss due to server maintenance -Loss due to training the new employees	-Getting huge number of customers -Online payment system -Customer product tracking -Improve Customer satisfaction -Graphical representation of monthly sales report -Improved production scheduling -Rating, 00comment system for customer

Table 1.1: Cost benefit analysis

1.4 Cash-flow analysis

Cash flow is the net amount of cash and cash-equivalents being transferred into and out of a business. It is the difference between revenue and total cost of a company. The cash flow analysis is given below:

Year 2018				
	Jun-Jul	**Aug-Sept**	**Oct-Nov**	**Dec-Jan**
Revenue	145000	232000	257000	294000
Costs				
Software Development Cost	10000	10000	0	0
Training	30000	35000	45000	52000
Equipments	130000	150000	110000	115000
Maintain	0	50000	82000	92000
Total Cost	170000	245000	237000	259000
Cash Flow	-25000	-13000	20000	35000
Cumulative Cash Flow	-25000	-38000	-18000	17000

Table 1.2: Cash-flow analysis

1.5 Present Value Analysis

We know that,

$$P.V = \frac{F.V}{(1+i)^n}$$

Here,
P.V = Present value
F.V = Future value
i = Interest rate
n = Time or period

Let us assume that for our project,
P.V = 20000
n = 4
i =0.08
So,

$$F.V = P.V * (1+i)^n$$

$$F.V = 20000 * (1+0.08)^4$$

$$F.V = 27210$$

1.6 Project Scheduling

Project scheduling is a mechanism to communicate what tasks need to get done and which organizational resources will be allocated to complete those tasks in what timeframe. A project schedule is a document collecting all the work needed to deliver the project on time. Our project scheduling is given below:

Activity	**Description**	**Predecessors**	**Time(Days)**
A	Project Initiation	None	10
B	Report Submission	None	1
C	Interview & Questioning	A, B	7
D	Project Planning	C	5
E	Analyzing System Needs	C, D	12
F	Diagram Designing	E	12
G	Data Analysis	C, F	7
H	Database Designing	F, G	10
I	Coding	E, G, H	28
J	Documentation	I	5
K	Testing	I	8
L	Bug Fixing	K	5
M	Deployment & Training	L	10

Table 1.3: Project Scheduling

1.7 Project Scheduling Chart

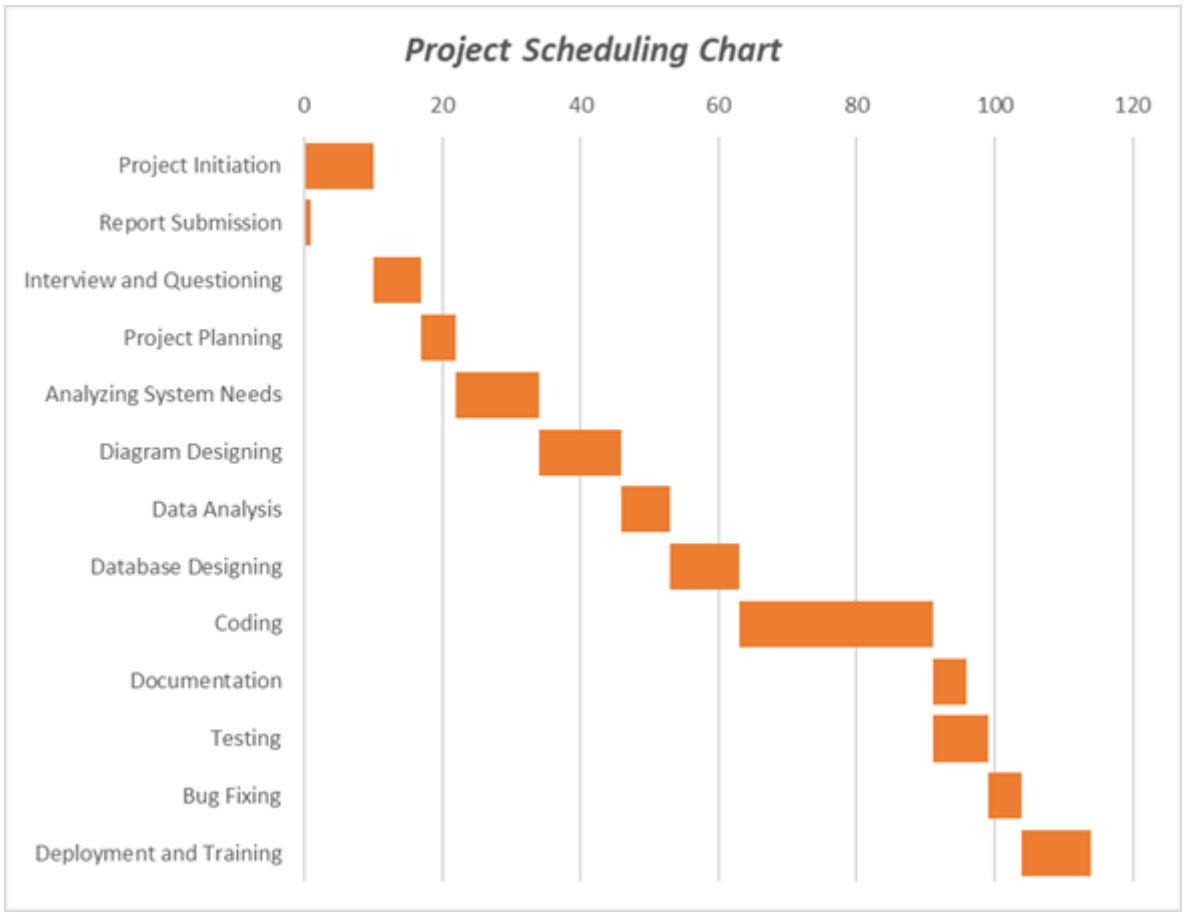

Figure 1.1: Project Scheduling Chart

1.8 Conclusion

Our project “Biponee” will take around 4 months to complete. After that we submit this project to an E-commerce company. We hope “Biponee” will contribute a lot to the E-commerce of our country.

Chapter 2

Information Gathering

2.1 Selection the objectives of Information Gathering Interview

"Reverie Bangladesh" is a clothing (brand) which are currently selling various kinds of t-shirts through their Facebook page. After getting better response from customers and clients, they want to grow up their business.

So they offered us to develop a web based system named "Biponee" where they can sell variety of products such as men's clothing, women's clothing, watch and jewelry, electronics etc.

So, to know about how their existing system works, how they maintain it, we called an interview on 24 May, 2018. This interview helps us to gather information from them which is helpful for us to improve features to this project.

2.2 Questionnaires and Interview pattern

To know the client requirements, we need to gather information from them. There are various way to gather information. Some of them are describing below:

Interview: An interview is a conversation where questions are asked and answers are given. Interviews usually take place face to face and in person. We can modify it. In an interview we gather information by asking questions to the interviewee. There are two types of questions, which are:

1. Open-ended Question: Open-ended questions are ones that require more than one word answers. "Open" actually describes the interviewee's options for responding. They are open. The response can be two words or two paragraphs.

2. Closed Question: Closed questions can be answered in only one word or with a short, specific piece of information. A closed question limits the response available to the interviewee.

Questionnaire: A questionnaire is a research instrument consisting of a series of questions (or other types of prompts) for the purpose of gathering information from respondents. Questionnaires are also sharply limited by the fact that respondents must be able to read the questions and respond to them. In this process we can not modify the question and it is not necessary to face to face interaction.

For information gathering we use interview method, because for our project we need to interact with the client face to face for analyze their requirements. We used Pyramid structure for our system.

2.3 Selection of interview Personnel

We have selected Mr. Rahee Zaman, founder of “Reverie Bangladesh” for the interview. For our project it is necessary to know how they currently running their system, what are the difficulties they are facing to running this system, what are the requirements and features they want in “Biponee”. As a founder of ”Reverie Bangladesh”, Mr. Rahee Zaman knows everything about their existing system. So the whole interview was taken from him.

2.4 Summary of the total information and list of activities

After interviewing Mr. Rahee Zaman, we identify the drawbacks of their current system. We also know about some new features we need to integrate in “Biponee”. Besides they shared with us about the difficulties they are facing right now and about the future plan of Reverie Bangladesh.

The list of activities are:

- Knowing about order management
- Knowing about delivery system
- Current payment system
- Current inventory management
- Reasons behind shifting to BIPONEE
- Features
- Future plan

2.5 Conclusion

After the information gathering process, we have a complete idea about how they manage their sections, what they want in “Biponee”, how they take/serve the orders from/to the customer, communicate with the customer etc. Hopefully by our project “Biponee” they will manage their business smartly, will achieve customer satisfaction, and grow up their business rapidly.

Chapter 3

Data Flow Diagram and Use case Diagram

3.1 Data Flow Diagram

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modelling its process aspects. A DFD is often used as a preliminary step to create an overview of the system without going into great detail, which can later be elaborated. Activities of the project

- Login/Sign up
- Browse sections
- Cart system
- Product tracking

3.1.1 Main Process

- Biponee - An online shopping system

3.1.2 Sub-process

- Sign up
- Login
- Edit section
- Edit products
- Search products
- Add to cart
- Checkout
- View orders
- Show customers information
- Product tracking
- Analysis

3.1.3 Entity

- Customer
- Visitor
- Admin

3.1.4 Data store

- Customers
- Admins
- Carts
- Orders
- Products
- Sections

3.1.5 Levels of Data Flow Diagram

- Context level DFD
- Level 1 DFD
- Level 2 DFD

3.1.6 Context level DFD

A context diagram is a top level (also known as "Level 0") data flow diagram. It only contains one process node ("Process 0") that generalizes the function of the entire system in relationship to external entities.

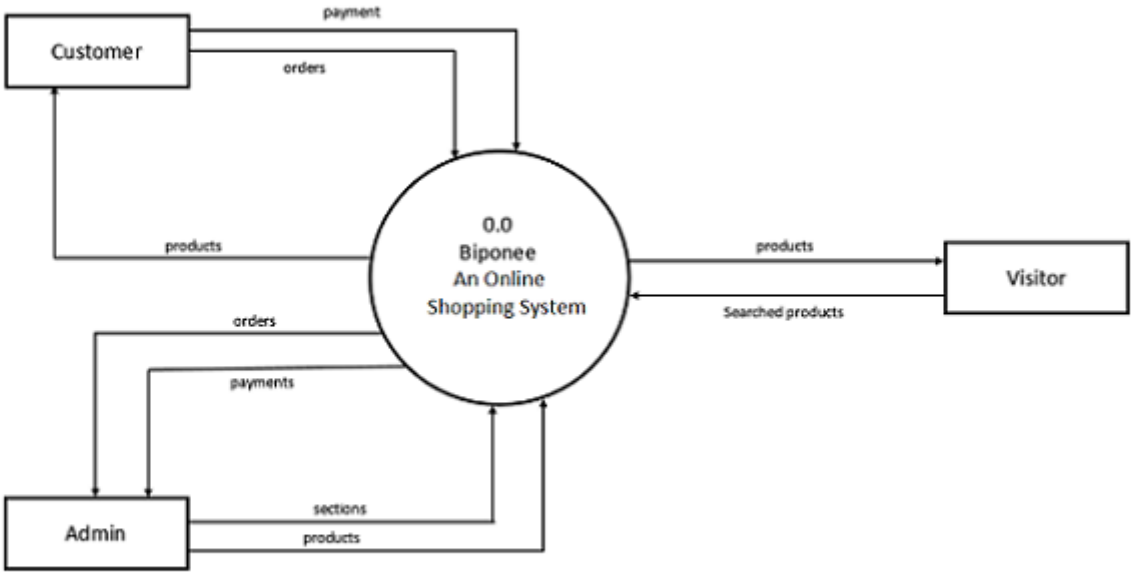

Figure 3.1: Context Level DFD of Biponee

3.1.7 level 1 DFD

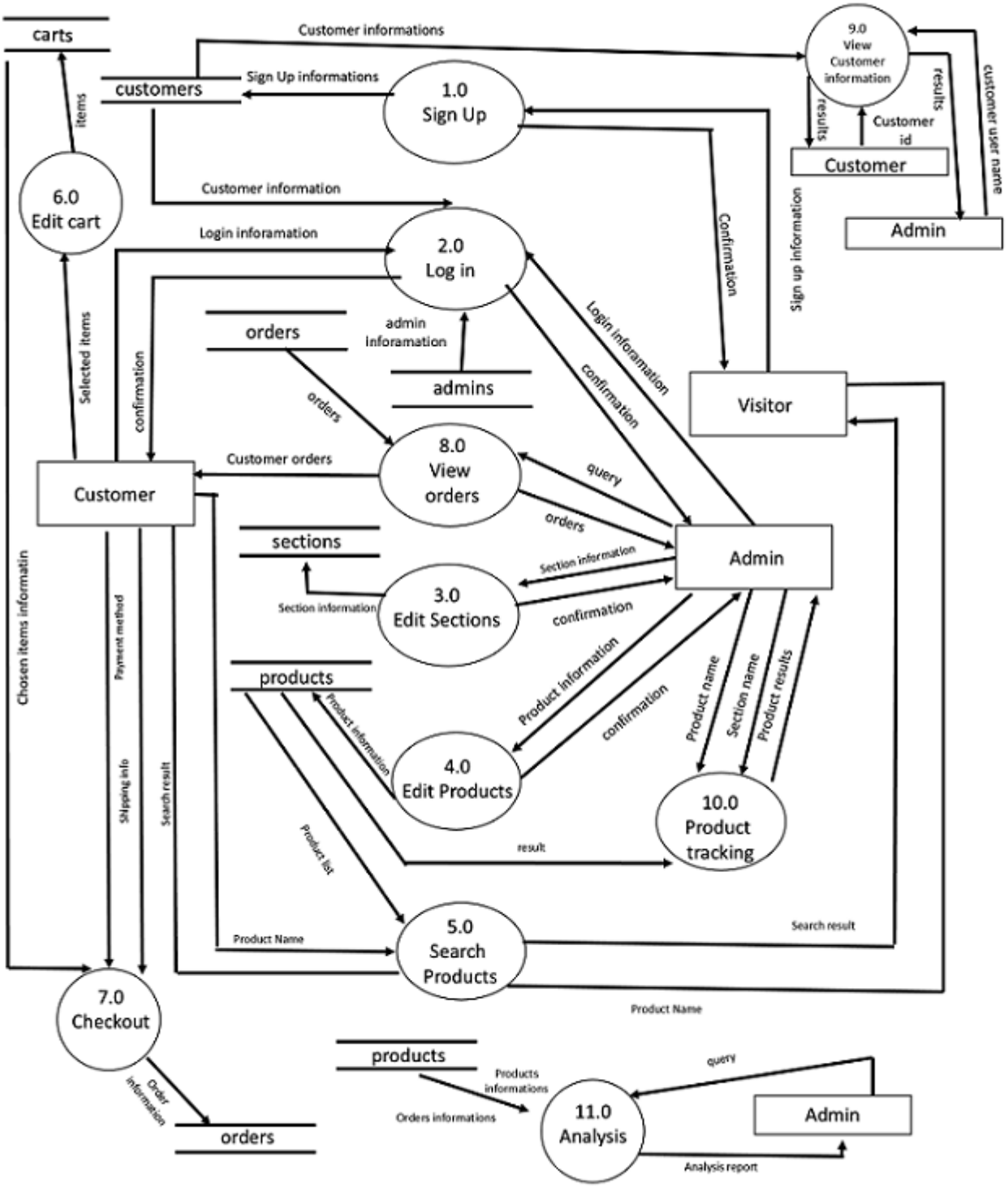

Figure 3.2: Level 1 DFD of Biponee

3.1.8 Level 2 DFD

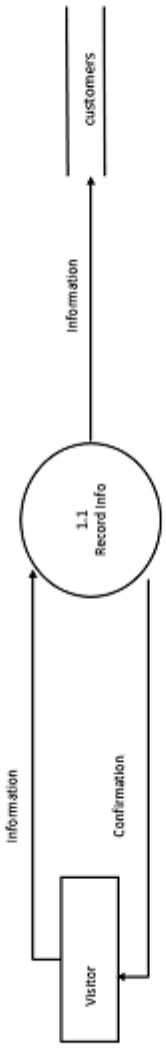

Figure 3.3: Sub-processes of Sign Up

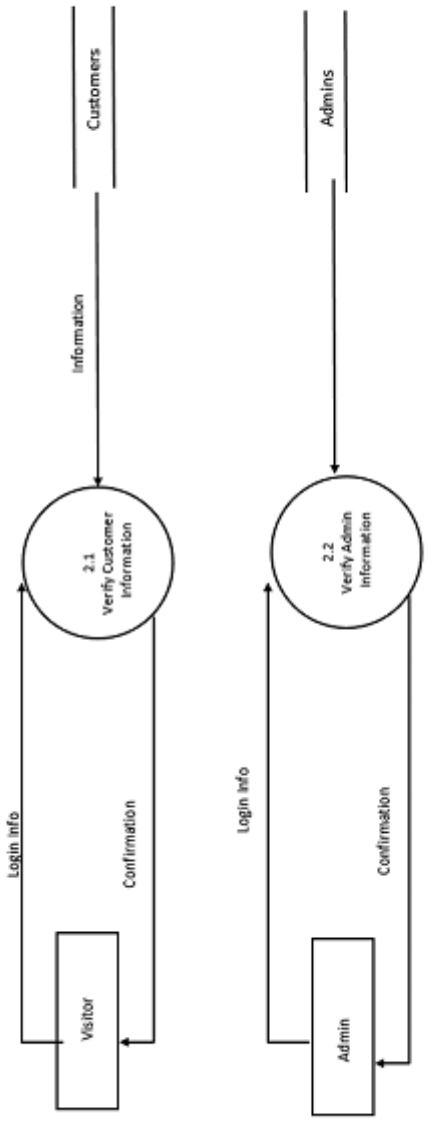

Figure 3.4: Sub-processes of Log in

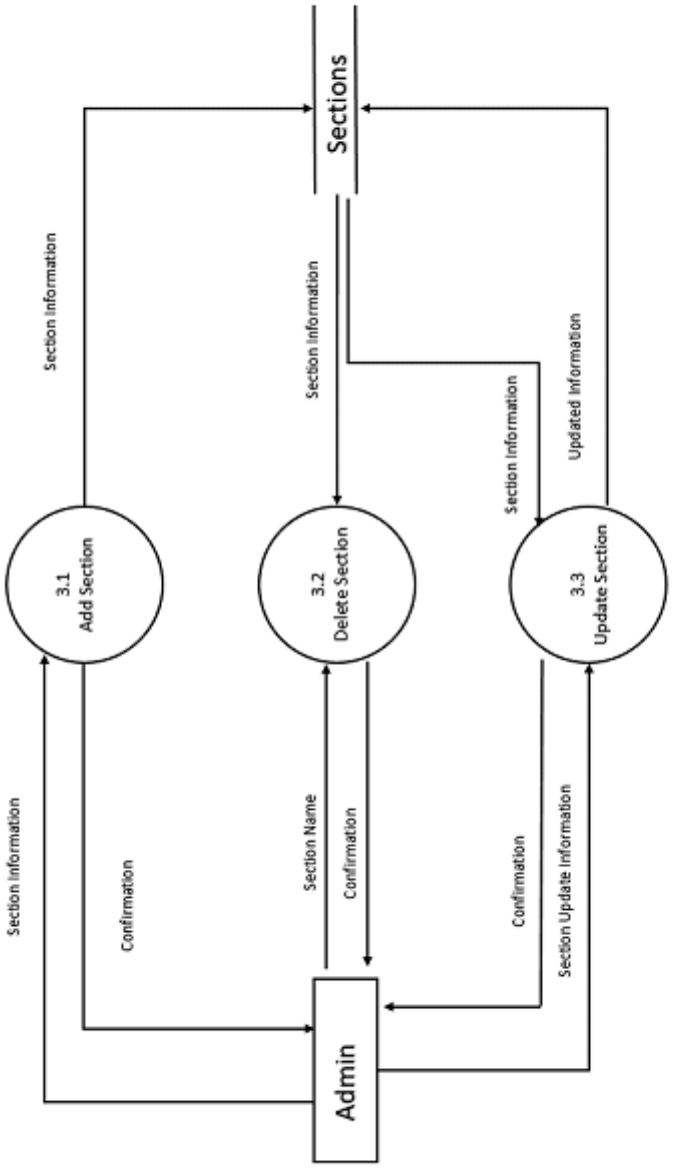

Figure 3.5: Sub-processes of Edit Sections

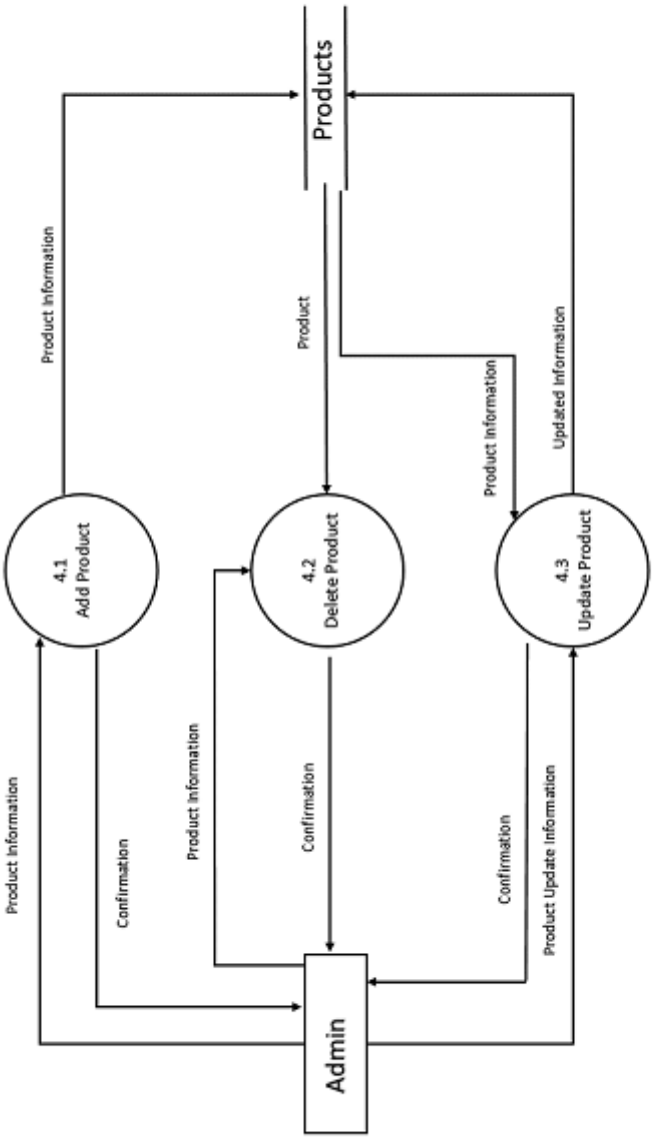

Figure 3.6: Sub-processes of Edit products

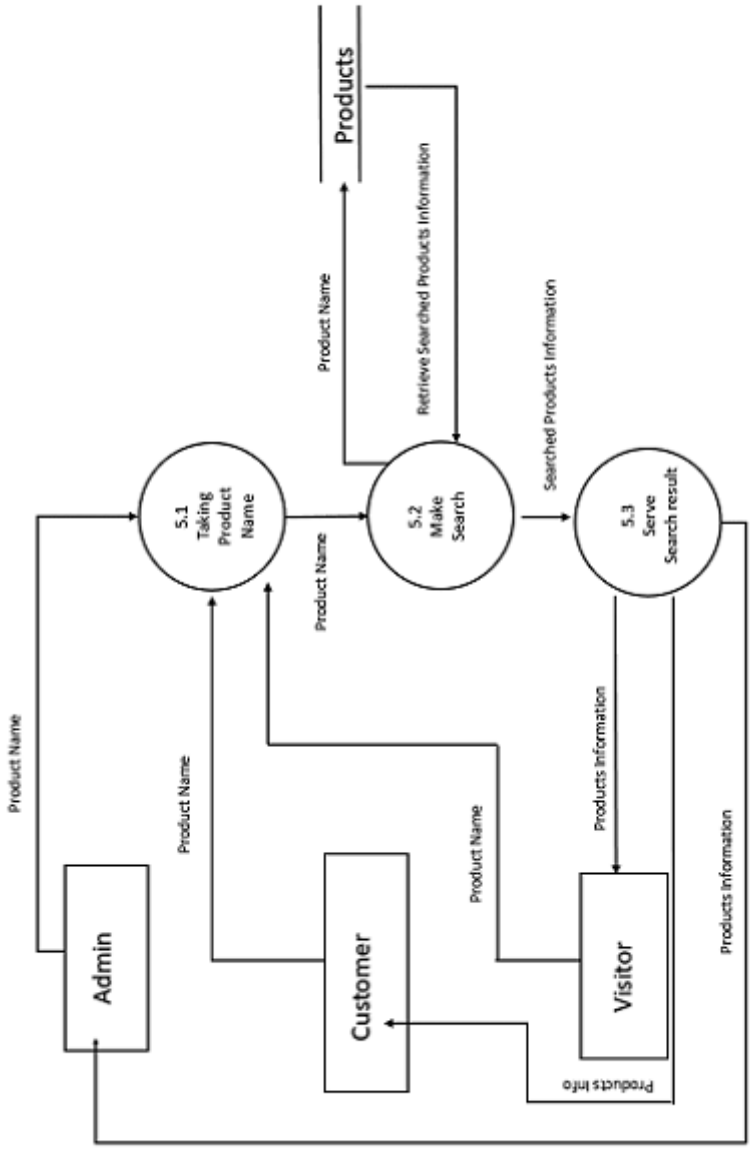

Figure 3.7: Sub-processes of Search Products

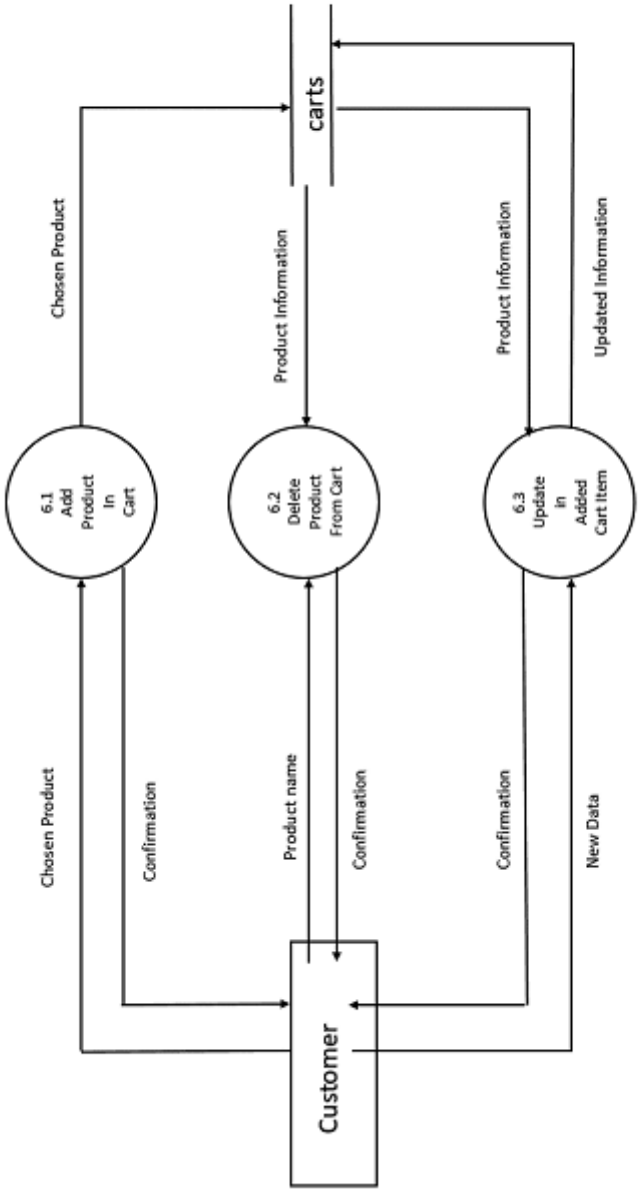

Figure 3.8: Sub-processes of Edit cart

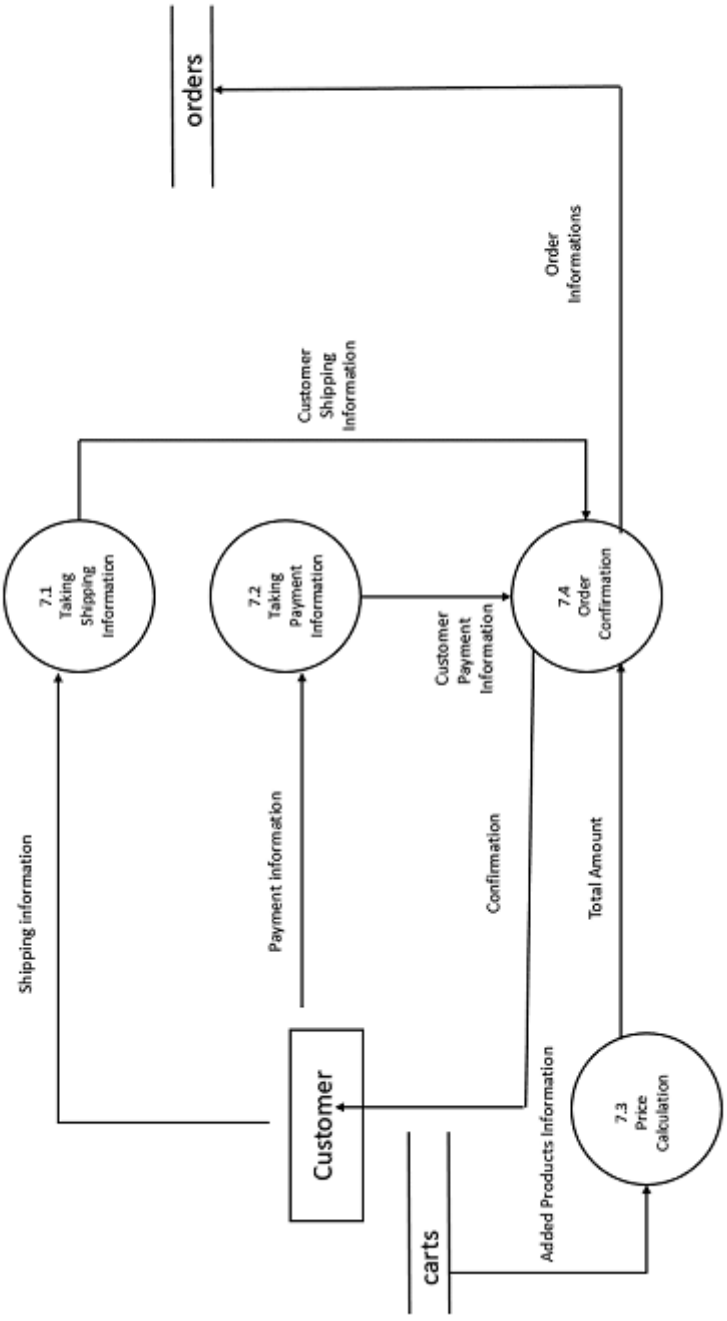

Figure 3.9: Sub-processes of Order confirmation

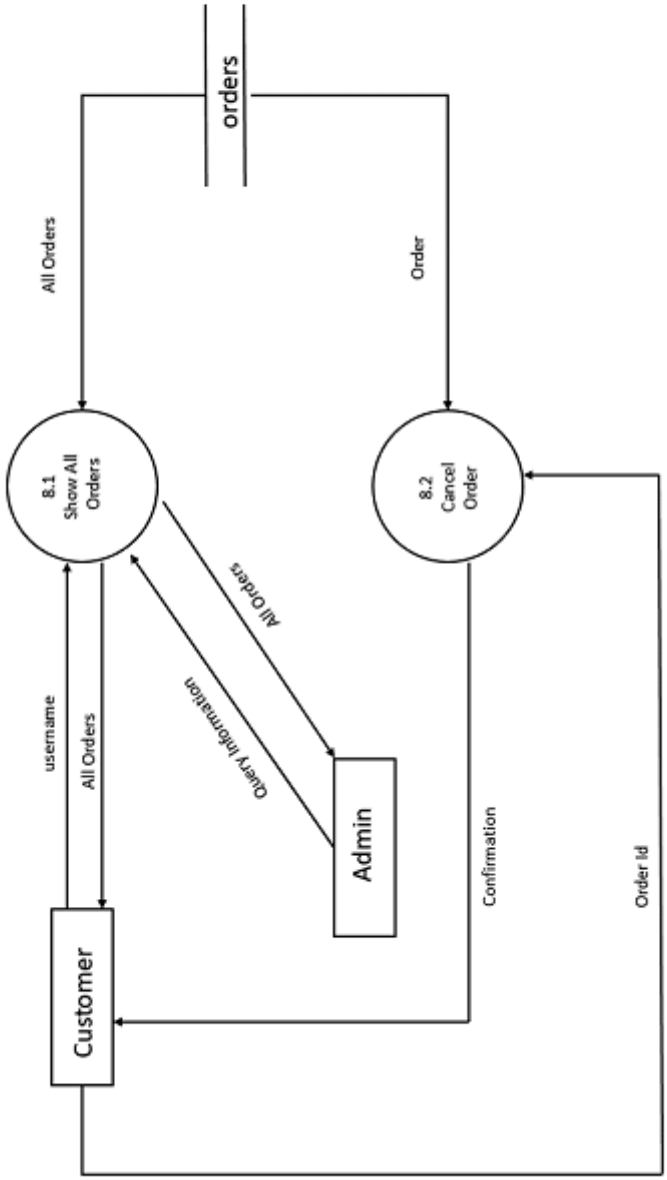

Figure 3.10: Sub-processes of View orders

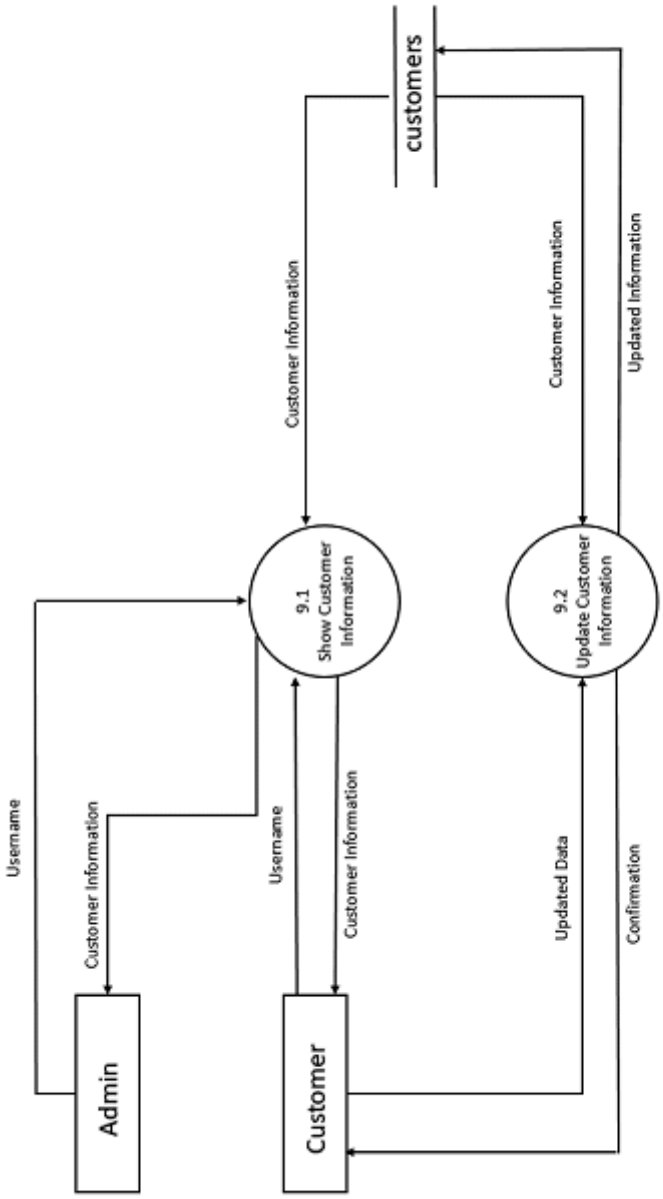

Figure 3.11: Sub-processes of View Customer information

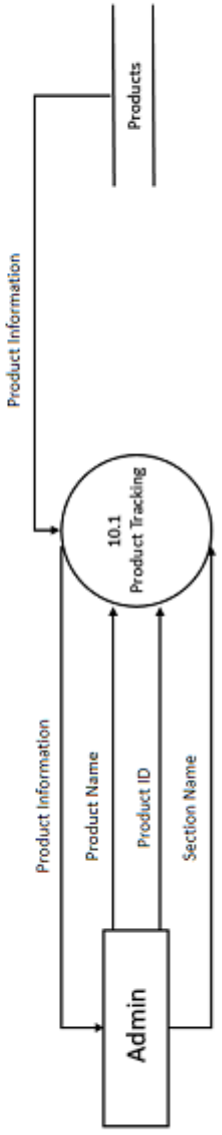

Figure 3.12: Sub-processes of Product tracking

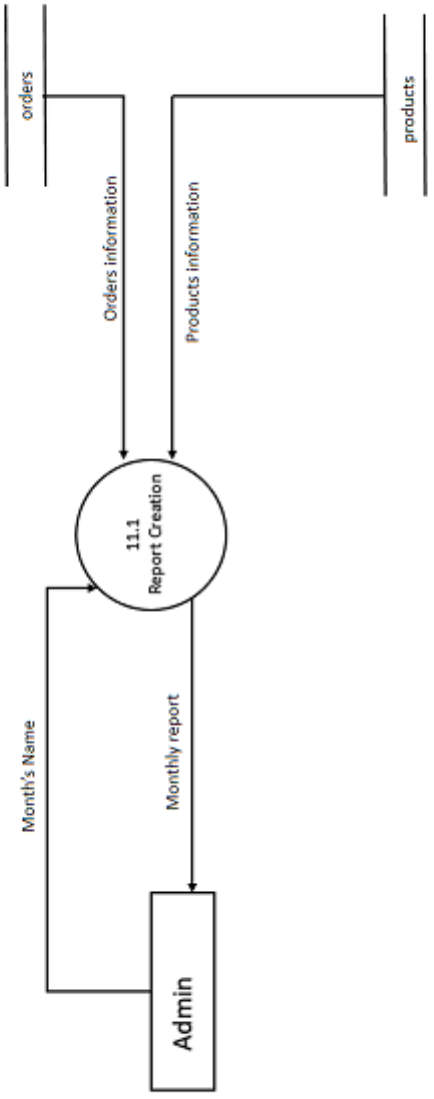

Figure 3.13: Sub-processes of Analysis

3.2 Use Case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved.

3.2.1 Actors

An actor specifies a role played by a user or any other system that interacts with the subject.
The list of actors used by us:

- Customer
- Visitor
- Admin
- Biponee Website

3.2.2 Use Cases

Use case is the sequence of actions that the system performs that yields an observable result of value to an actor.

- Sign up
- Login
- Browse sections
- Add to cart
- Checkout
- Edit sections
- Edit products
- View orders
- View customer informations
- Product tracking
- Analysis
- Confirmation
- Verify password
- Display login error
- Total calculation
- Update sections table
- Update products table
- Report calculations

3.2.3 Details of Use Case Models

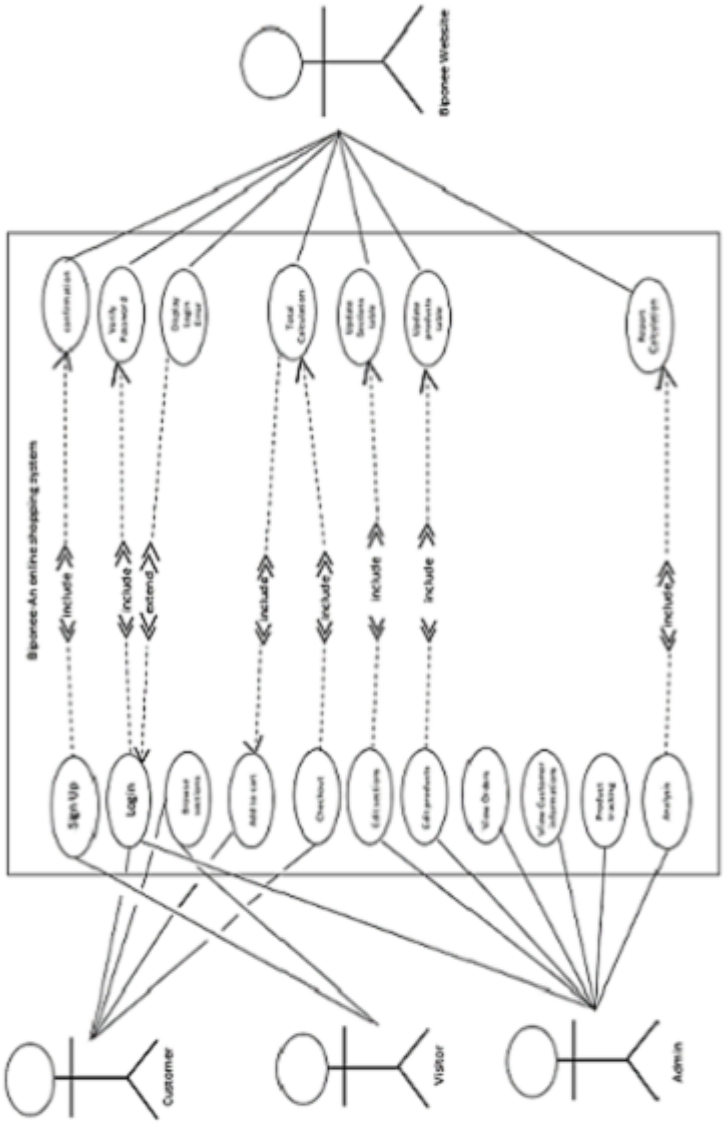

Figure 3.14: Use Case Diagram of Biponee

3.2.4 Conclusion

A Data flow diagram shows what kind of information will be input to and output from the system, how the data will advance through the system, and where the data will be stored. It does not show informations about process timing. From the DFD we can gather the clear concept of data flow of our project.

We also draw the use case diagram of Biponee. we identify the actors and the use cases of our project from the diagram. We also identify how actually the actors communicate with the use cases from this diagram.

We hope this use case diagram and data flow diagram concept will help us for develop Biponee efficiently.

Chapter 4

Entity Relationship Diagram and Class Diagram

4.1 Entity-Relationship Diagram

An entity relationship diagram (ERD), also known as an entity relationship model, is a graphical representation of an information system that depicts the relationships among people, objects, places, concepts or events within that system. An ERD is a data modeling technique that can help define business processes and be used as the foundation for a relational database.

4.1.1 Entity

- Admin
- User
- Section
- Products
- Cart
- Orders

4.1.2 Attribute

- Admin: AdminID, FirstName, LastName, Password
- User: Email, FirstName, LastName, Password, UserID
- Section: SectionID, SectionName
- Products: Quantity, Sizes, Description, Category, Price, SectionID, ProductCode, ProductName, ProductID, ImgLink, Bandname
- Cart: Quantity, UserID, OrderID, ProductID, SubTotal, CartItemID
- Orders: orderID, FirstName, PaymentMethod, Total, CartItemID, UserID, Adress, Phone, LastName

4.1.3 Relationship

- Admin manages Order
- Section contains product
- Cart contain product

- Order contains cart
- User places order

4.1.4 Diagram

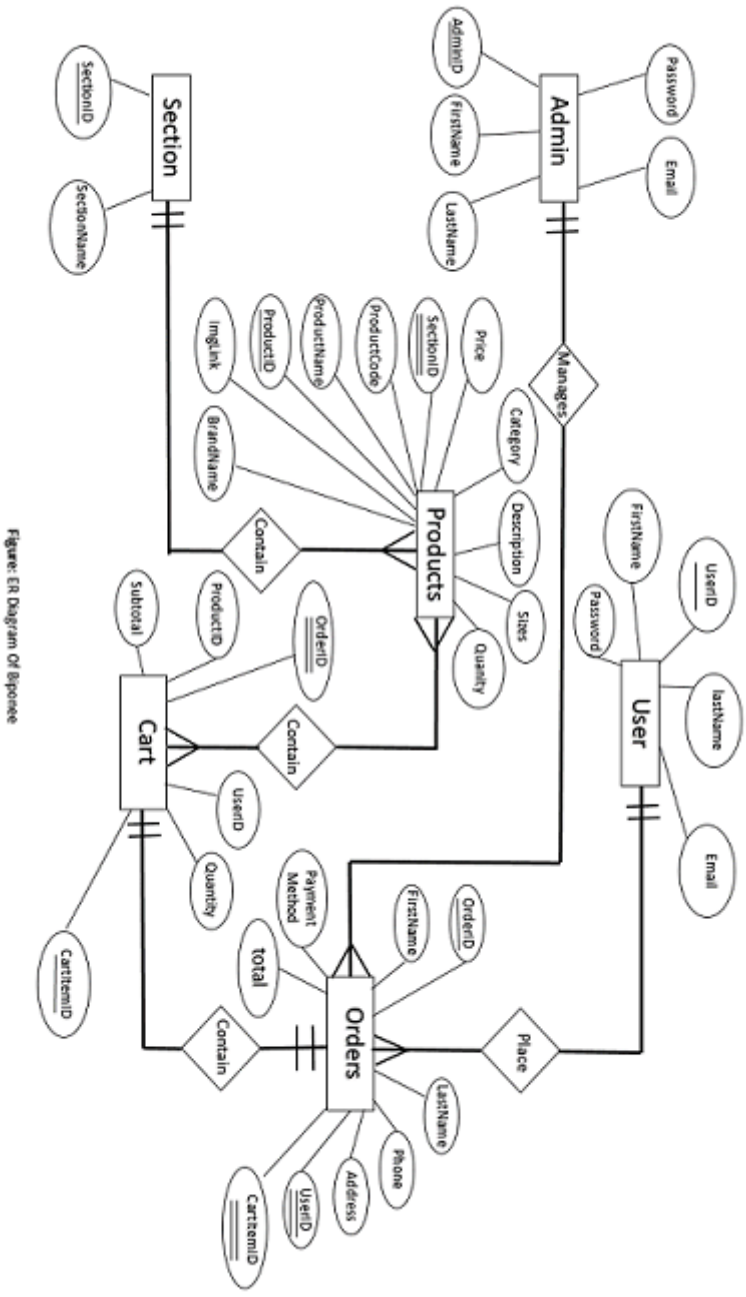

Figure: ER Diagram Of Biponee

Figure 4.1: ERD of Biponee

4.2 Class Diagram

Class diagrams are one of the most useful types of diagrams in UML as they clearly map out the structure of a particular system by modeling its classes, attributes, operations, and relationships between objects.

4.2.1 Class

- User
- Admin
- ClothingsProduct
- Section
- Order
- Product
- CartItem
- OthersProduct

4.2.2 Relationship

- Inheritance
- Association
- Aggregation
- Composition

4.2.3 Multiplicity

- 0 to 1 (Zero to one)
- 0...* (Zero to many)
- 1...* (One to many)
- m...n (Specific number range)

4.2.4 Diagram

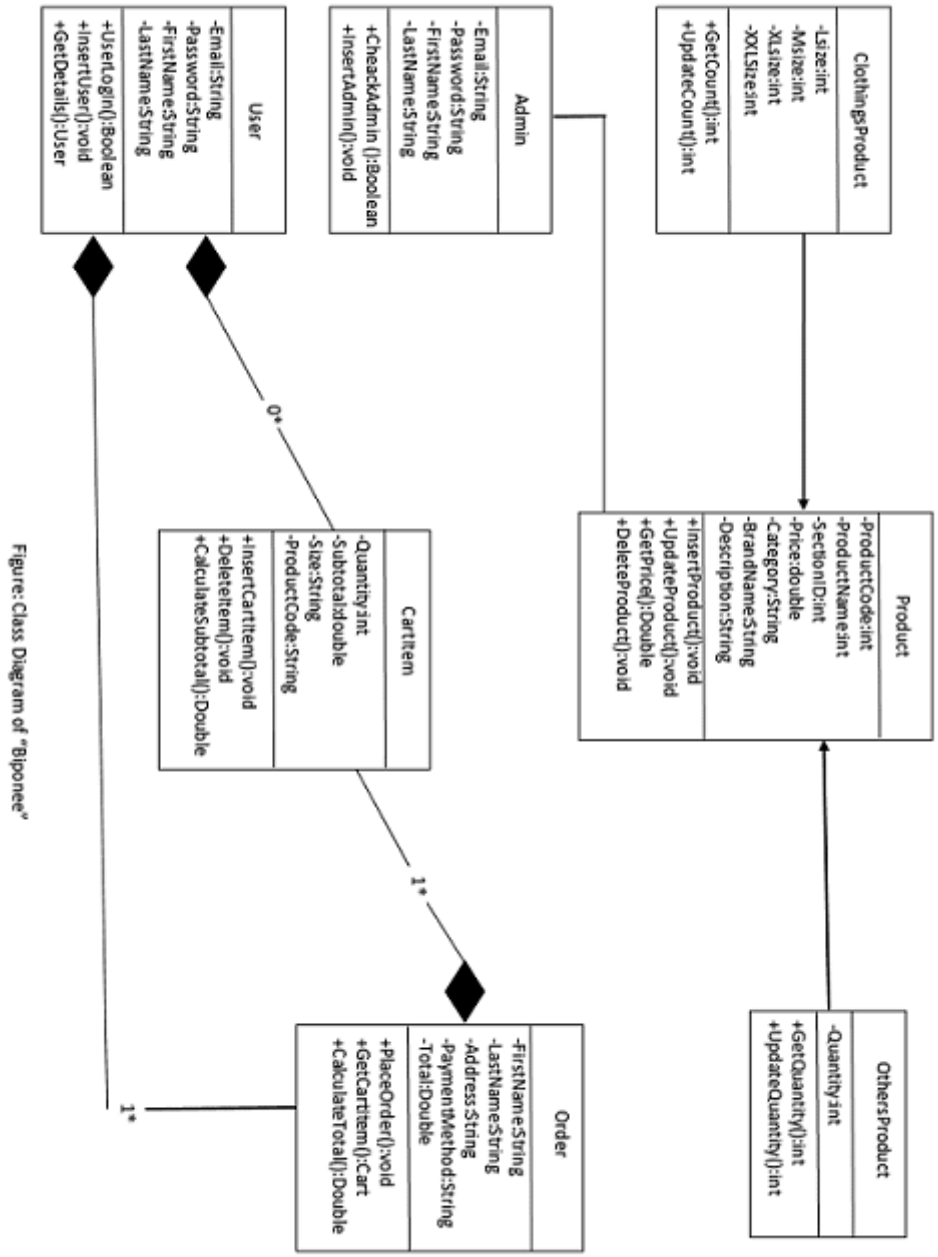

Figure: Class Diagram of "Biponee"

Figure 4.2: Class Diagram of Biponee

4.3 Conclusion

Entity relationship diagrams provide a visual starting point for database design that can also be used to help determine information system requirements throughout an organization. After a relational database is rolled out, an ERD can still serve as a referral point, should any debugging or business process re-engineering be needed later. We draw our ER Diagram which give us the clear Reflection of our project database.
In Class Diagram a class defines the methods and variables in an object, which is a specific entity in a program or the unit of code representing that entity. Class diagrams are useful in all forms of object-oriented programming (OOP).

Chapter 5

Discussion

For developing 'Biponee' we follow different phases.At first we analyze the project feasibility. Here we analyze three kinds of project feasibility. They are operational feasibility, technical feasibility and economical feasibility.
After that we analyze cost benefit and present value and future value of the project finally we make a project scheduling chart at this step.
Then we go to information gathering phase. In this phase we collect information from our client "Reverie Bangladesh" and gain the complete idea about how they manage their current system.
In the next step we design the data flow diagram and Use case diagram. From data flow diagram we identified what kind of data will be the input and output of our project, how data will advance in this system and where the data will be stored. From use case diagram we identify the use case and the actors of our project.
Then we design the ER diagram and class diagram. In ER diagram we identify the entities and relationships of our project. The ER diagram provide the visual starting point of our project database. The class diagram defines the method and variables of an object. This class diagram is useful in all forms of object oriented programming.

The technologies we used in frontend development are:

- HTML
- CSS
- Javascript
- jQuery
- AJAX

The technologies we used in backend are:

- C
- SQL

The IDE we used in this project development:

- Microsoft Visual Studio 2018
- Microsoft SQL Server Management Studio 17

The architecture we used:

- MVC Architecture
- Layer Architecture

The framework we used:

- Asp.net MVC Framework

In the future we will add more section in our project. Try to implement Artificial intelligence(AI), and Machine learning to make our project more efficient.

Appendices

Appendix A

Implementation

A.1 Model

A.1.1 AdminC.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace Biponee.Models
{
    public class AdminC
    {
        public int ID { get; set; }
        public String Email { get; set; }
        public String Password { get; set; }
        public String FirstName { get; set; }
        public String LastName { get; set; }

        public AdminC(int iD, String email, String password,
        String firstName, String lastName)
        {
            ID = iD;
            Email = email;
            Password = password;
            FirstName = firstName;
            LastName = lastName;
        }

        public AdminC(String email, String password,
        String firstName, String lastName)
        {
            Email = email;
            Password = password;
            FirstName = firstName;
            LastName = lastName;
        }

        public AdminC()
        {
        }
    }
}
```

A.1.2 UserC.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace Biponee.Models
{
    public class UserC
    {
        public int Id { get; set; }
        public String FirstName { get; set; }
        public String LastName { get; set; }
        public String Email { get; set; }
        public String Password { get; set; }

        public UserC(int id, string firstName,
        string lastName, string email, string password)
        {
            Id = id;
            FirstName = firstName;
            LastName = lastName;
            Email = email;
            Password = password;
        }

        public UserC( string firstName,string lastName,
        string email, string password)
        {

            FirstName = firstName;
            LastName = lastName;
            Email = email;
            Password = password;
        }


    }
}
```

A.1.3 CartItemC.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace Biponee.Models
{
    public class CartItemC
    {
        public int OrderId { get; set; }
        public int ProductId { get; set; }
        public String ProductImage { get; set; }
        public String ProductName { get; set; }
        public String ProductSize { get; set; }
        public Double UnitPrice { get; set; }
        public int Qty { get; set; }
        public Double Subtotal { get; set; }

        public CartItemC(int orderId,int productId,
        string productImage, string productName,
        string productSize,double unitPrice, int qty, double subtotal)
        {
            OrderId = orderId;
            ProductId = productId;
            ProductImage = productImage;
            ProductName = productName;
            ProductSize = productSize;
            UnitPrice = unitPrice;
            Qty = qty;
            Subtotal = subtotal;
        }
        public CartItemC( int productId, string productImage,
        string productName, string productSize,
        double unitPrice, int qty, double subtotal)
        {

            ProductId = productId;
            ProductImage = productImage;
            ProductName = productName;
            ProductSize = productSize;
            UnitPrice = unitPrice;
            Qty = qty;
            Subtotal = subtotal;
        }


    }
}
```

A.1.4 OrderC.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace Biponee.Models
{
    public class OrderC
    {
        public int OrderId { get; set; }
        public String FirstName { get; set; }
        public String LastName { get; set; }
        public String Phone { get; set; }
        public String Address { get; set; }
        public String State { get; set; }
        public String City { get; set; }
        public int UserId { get; set; }
        public String PaymentMethod { get; set; }
        public Double Total { get; set; }

       public OrderC(int orderId, string firstName,
       string lastName, string phone,
       string address, string state,
       string city, int userId,
       string paymentMethod, double total)
        {
            OrderId = orderId;
            FirstName = firstName;
            LastName = lastName;
            Phone = phone;
            Address = address;
            State = state;
            City = city;
            UserId = userId;
            PaymentMethod = paymentMethod;
            Total = total;
        }
        public OrderC(string firstName, string lastName,
        string phone, string address,
        string state, string city, int userId,
        string paymentMethod, double total)
        {
            FirstName = firstName;
            LastName = lastName;
            Phone = phone;
            Address = address;
            State = state;
            City = city;
            UserId = userId;
            PaymentMethod = paymentMethod;
            Total = total;
        }
    }
}
```

A.1.5 ClothingProduct.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace Biponee.Models.Products
{
    public class ClothingProduct : Product
    {
        public int LCount { get; set; }
        public int MCount { get; set; }
        public int XLCount { get; set; }
        public int XXLCount { get; set; }

        public ClothingProduct(String productName,
        String productCode, int sectionId,
        Double price,String category,
        String description,
        String imageLink,String brandName,
            int lCount,int mCount,int xlCount,
            int xxlCount)
            :base(productName,
            productCode,sectionId,price,
            category,description,imageLink,
            brandName)
        {
            this.LCount = lCount;
            this.MCount = mCount;
            this.XLCount = xlCount;
            this.XXLCount = xxlCount;
        }

        public ClothingProduct(int productId,
        String productName, String productCode,
        int sectionId, Double price,
        String category,
        String description,
        String imageLink,
        String brandName,
           int lCount, int mCount,
           int xlCount, int xxlCount)
           : base(productId,
           productName,
           productCode, sectionId, price,
           category,
           description, imageLink, brandName)
        {
            this.LCount = lCount;
            this.MCount = mCount;
            this.XLCount = xlCount;
            this.XXLCount = xxlCount;
        }

        public ClothingProduct():base()
        {
        }
```

```
        }
}
```

A.1.6 DailyNeedProduct.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace Biponee.Models.Products
{
    public class DailyNeedProduct : Product
    {
       public int Quantity { get; set; }
        public DailyNeedProduct(string productName,
        string productCode, int sectionId,
        double price,int quntity ,string category,
        string description, string imageLink,
        string brandName) : base(productName,
        productCode, sectionId, price, category,
        description, imageLink, brandName)
        {
            this.Quantity = quntity;
        }

        public DailyNeedProduct(int id,
        string productName, string productCode,
        int sectionId,
        double price, int quntity, string category,
        string description,
        string imageLink,
        string brandName) : base(id,productName,
        productCode, sectionId, price, category,
        description, imageLink, brandName)
        {
            this.Quantity = quntity;
        }

        public DailyNeedProduct() : base()
        {

        }
    }
}
```

A.1.7 ElectronicsProduct.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace Biponee.Models.Products
{
    public class ElectronicsProduct : Product

    {

        public int Quantity { get; set; }
        public ElectronicsProduct(string productName,
        string productCode, int sectionId, double price,
        int quantity,string category,
        string description,
        string imageLink,
        string brandName) : base(productName, productCode,
        sectionId, price, category,
        description, imageLink, brandName)
        {
            this.Quantity = quantity;
        }
        public ElectronicsProduct(int id,string productName,
        string productCode, int sectionId,
        double price, int quantity, string category,
        string description, string imageLink,
        string brandName) : base(id,
        productName, productCode,
        sectionId, price, category,
        description, imageLink, brandName)
        {
            this.Quantity = quantity;
        }

        public ElectronicsProduct():base()
        {
        }
    }
}
```

A.1.8 MobileProduct.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace Biponee.Models.Products
{
    public class MobileProduct : Product
    {
        public int Quantity { get; set; }
        public MobileProduct(string productName,
        string productCode, int sectionId,
        double price, int quantity,
        string category, string description,
```

```
        string imageLink, string brandName) :
        base(productName, productCode,
        sectionId, price, category,
        description, imageLink, brandName)
        {
            this.Quantity = quantity;
        }
        public MobileProduct(int id,string
        productName, string productCode,
        int sectionId, double price,
        int quantity, string category,
        string description,
        string imageLink,
        string brandName) : base(id,productName,
        productCode, sectionId, price, category,
        description,
        imageLink, brandName)
        {
            this.Quantity = quantity;
        }


        public MobileProduct() : base()
        {

        }
    }
}
```

A.1.9 Product.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace Biponee.Models.Products
{
    public class Product
    {
        public int ProductId { get; set; }
        public String ProductName { get; set; }
        public String ProductCode { get; set; }
        public int SectionId { get; set; }
        public Double Price { get; set; }
        public String Category { get; set; }
        public String Description { get; set; }
        public String ImageLink { get; set; }
        public String BrandName { get; set; }

        public Product(string productName,
        string productCode, int sectionId,
        Double price, string category,
        string description,
        string imageLink, string brandName)
        {
            ProductName = productName;
            ProductCode = productCode;
            SectionId = sectionId;
            Price = price;
            Category = category;
            Description = description;
            ImageLink = imageLink;
            BrandName = brandName;
        }

        public Product(int productId,
        string productName,
        string productCode, int sectionId,
        Double price, string category, string description,
        string imageLink, string brandName)
        {
            ProductId = productId;
            ProductName = productName;
            ProductCode = productCode;
            SectionId = sectionId;
            Price = price;
            Category = category;
            Description = description;
            ImageLink = imageLink;
            BrandName = brandName;
        }

        public Product()
        {
        }
    }
```

```
}
```

A.2 Database

A.2.1 AdminGetway.cs

```
using Biponee.Models;
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Linq;
using System.Web;
using System.Web.Configuration;

namespace Biponee.DLL
{
    public class AdminGetway
    {
        String connectionString = WebConfigurationManager.ConnectionStrings
        ["biponeeDbConnectioon"].ConnectionString;

        public List<AdminC> getAdminByEmailAnsPassword
        (String email, String Password)
        {
            SqlConnection connection = new
            SqlConnection(connectionString);
            String Query = "SELECT * FROM admins
            WHERE Email = '" + email + "'
            AND Password = '" + Password + "'";

            SqlCommand command = new SqlCommand(Query, connection);
            connection.Open();
            SqlDataReader reader = command.ExecuteReader();

            List<AdminC> list = new List<AdminC>();

            while (reader.Read())
            {
                int id = Convert.ToInt32(reader["AdminID"]);
                String adminEmail = reader["Email"].ToString();
                String password = reader["Password"].ToString();
                String firstName = reader["FirstName"].ToString();
                String lastName = reader["LastName"].ToString();

               list.Add( new AdminC(id, adminEmail, password, firstName, lastName));
            }

            return list;
        }

        public List<AdminC> getAdminById(int adminId)
        {
            SqlConnection connection = new
            SqlConnection(connectionString);
            String Query = "SELECT * FROM admins
            WHERE AdminID = " + adminId;

            SqlCommand command = new
            SqlCommand(Query, connection);
```

```
            connection.Open();
            SqlDataReader reader = command.ExecuteReader();

            List<AdminC> list = new List<AdminC>();

            while (reader.Read())
            {
                int id = Convert.ToInt32(reader["AdminID"]);
                String adminEmail = reader["Email"].ToString();
                String password = reader["Password"].ToString();
                String firstName = reader["FirstName"].ToString();
                String lastName = reader["LastName"].ToString();

                list.Add(new AdminC(id, adminEmail,
                password, firstName, lastName));
            }

            return list;
        }
    }
}
```

A.2.2 OrderGetway.cs

```
using System;
using System.Collections.Generic;
using Biponee.Models;
using System.Web;
using System.Web.Configuration;
using System.Data.SqlClient;

namespace Biponee.DAL
{
    public class OrderGetway
    {
        String connectionString =
        WebConfigurationManager.ConnectionStrings
        ["biponeeDbConnectioon"].ConnectionString;

        public int InsertOrder(OrderC order)
        {
            SqlConnection connection = new
            SqlConnection(connectionString);
            String Query = "INSERT
            INTO orders VALUES ('" + order.FirstName + "','"
            + order.LastName + "','" + order.Phone + "','" +
            order.Address + "','" + order.State + "','" +
            order.City + "'," +
            order.UserId + ",'" +
            order.PaymentMethod + "'," + order.Total + ")";
            SqlCommand command = new SqlCommand(Query, connection);
            connection.Open();
            int res = command.ExecuteNonQuery();
            connection.Close();

            return res;
        }
    }
}
```

A.2.3 ProductGetway.cs

```
using Biponee.Models;
using Biponee.Models.Products;
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Linq;
using System.Web;
using System.Web.Configuration;

namespace Biponee.DLL
{
    public class ProductGetway
    {
        String connectionString = WebConfigurationManager.
        ConnectionStrings["biponeeDbConnectioon"].ConnectionString;

        public int insertClothingProduct(ClothingProduct product)
        {
            SqlConnection connection =
            new SqlConnection(connectionString);
            String Query = "INSERT INTO products
            VALUES ('" + product.ProductName + "','" +
            product.ProductCode + "'," + product.SectionId + ","
            + product.Price + ",'" + product.Category + "','"
            + product.Description + "','" + product.ImageLink + "'," +
            product.LCount + "," + product.MCount + "," +
            product.XLCount + "," + product.XXLCount+",null" + ",'" +
            product.BrandName + "')";
            SqlCommand command = new SqlCommand(Query, connection);
            connection.Open();
            int res = command.ExecuteNonQuery();
            connection.Close();

            return res;
        }

        public int insertElectronicsProduct
        (ElectronicsProduct product)
        {
            SqlConnection connection = new
            SqlConnection(connectionString);
            String Query = "INSERT INTO products
            VALUES ('" +
            product.ProductName + "','" + product.ProductCode + "',"
            + product.SectionId + "," + product.Price + ",'"
            + product.Category +
            "','" + product.Description + "','" +
            product.ImageLink + "',null,null,null,null,"  +
            product.Quantity + ",'" + product.BrandName + "')";
            SqlCommand command = new SqlCommand(Query, connection);
            connection.Open();
            int res = command.ExecuteNonQuery();
            connection.Close();

            return res;
        }
```

```
public int insertDailyNeedsProduct
(DailyNeedProduct product)
{
    SqlConnection connection = new
    SqlConnection(connectionString);
    String Query = "INSERT INTO products VALUES
    ('" + product.ProductName + "','" +
    product.ProductCode + "'," + product.SectionId + "," +
    product.Price + ",'" + product.Category + "','" +
    product.Description + "','" + product.ImageLink + "',
    null,null,null,null," + product.Quantity + ",'" +
    product.BrandName + "')";
    SqlCommand command = new
    SqlCommand(Query, connection);
    connection.Open();
    int res = command.ExecuteNonQuery();
    connection.Close();

    return res;
}
public int insertMobileProduct
(MobileProduct product)
{
    SqlConnection connection = new
    SqlConnection(connectionString);
    String Query = "INSERT INTO products VALUES
    ('" + product.ProductName + "','" + product.ProductCode
    + "'," + product.SectionId + "," + product.Price + ",'"
    + product.Category +
    "','" + product.Description + "','"
    + product.ImageLink + "',
    null,null,null,null,"
    + product.Quantity + ",'" +
    product.BrandName + "')";
    SqlCommand command = new SqlCommand
    (Query, connection);
    connection.Open();
    int res = command.ExecuteNonQuery();
    connection.Close();

    return res;
}


public List<ClothingProduct> getAllClothingProduct()
{
    SqlConnection connection =
    new SqlConnection(connectionString);
    String Query = "SELECT * FROM products
    WHERE SectionId = 1";

    SqlCommand command = new
    SqlCommand(Query, connection);
    connection.Open();
    SqlDataReader reader = command.ExecuteReader();
```

```
            List<ClothingProduct> list =
            new List<ClothingProduct>();

            while (reader.Read())
            {
                int id = (int)reader["ProductId"];
                String name = reader["ProductName"].ToString();
                String code = reader["ProductCode"].ToString();
                int secId = (int)reader["SectionId"];
                Double price = Convert.ToDouble(reader["Price"].
                ToString());
                String category = reader["Category"].ToString();
                String description = reader["Description"].ToString();
                String imgLink = reader["ImageLink"].ToString();
                int lCount = (int)reader["LCount"];
                int mCount = (int)reader["MCount"];
                int xlCount = (int)reader["XLCount"];
                int xxlCount = (int)reader["XXLCount"];
                String brandName = reader["BrandName"].ToString();

            list.Add(new ClothingProduct(id,name, code,
            secId, price, category, description,
            imgLink, brandName, lCount,
            mCount, xlCount,xxlCount));
            }

            return list;
        }

    public List<Product> getAllProduct(String pCode)
    {
        SqlConnection connection = new
        SqlConnection(connectionString);
        String Query = "SELECT * FROM products
        WHERE ProductCode ='"+ pCode + "'";

        SqlCommand command = new SqlCommand(Query, connection);
        connection.Open();
        SqlDataReader reader = command.ExecuteReader();
        List<Product> list = new List<Product>();

        while (reader.Read())
        {
            int id = (int)reader["ProductId"];
            String name = reader["ProductName"].ToString();
            String code = reader["ProductCode"].ToString();
            int secId = (int)reader["SectionId"];
            Double price = Convert.ToDouble(reader["Price"].ToString());
            String category = reader["Category"].ToString();
            String description = reader["Description"].ToString();
            String imgLink = reader["ImageLink"].ToString();
            int lCount = (int)reader["LCount"];
            int mCount = (int)reader["MCount"];
            int xlCount = (int)reader["XLCount"];
            int xxlCount = (int)reader["XXLCount"];
            String brandName = reader["BrandName"].ToString();

            list.Add(new Product(id, name, code,
```

```
            secId, price, category,
            description, imgLink, brandName));
        }

        return list;
    }


    public List<ElectronicsProduct> getElectronicProduct()
    {
        SqlConnection connection = new
        SqlConnection(connectionString);
        String Query = "SELECT * FROM products
        WHERE SectionId = 2";

        SqlCommand command = new SqlCommand(Query, connection);
        connection.Open();
        SqlDataReader reader = command.ExecuteReader();


        List<ElectronicsProduct> list =
        new List<ElectronicsProduct>();

        while (reader.Read())
        {
            int id = (int)reader["ProductId"];
            String name = reader["ProductName"].ToString();
            String code = reader["ProductCode"].ToString();
            int secId = (int)reader["SectionId"];
            Double price = Convert.ToDouble(reader["Price"].ToString());
            String category = reader["Category"].ToString();
            String description = reader["Description"].ToString();
            String imgLink = reader["ImageLink"].ToString();
            int Quantity = (int)reader["Quantity"];
            String brandName = reader["BrandName"].ToString();
            list.Add(new ElectronicsProduct(id,name, code,
            secId, price, Quantity, category,
            description, imgLink, brandName));
        }

        return list;
    }

    public List<ElectronicsProduct> getElectronicProduct
    (String productCode)
    {
        SqlConnection connection =
        new SqlConnection(connectionString);
        String Query = "SELECT * FROM products
        WHERE SectionId = 2 AND
        ProductCode='"+productCode+"'";

        SqlCommand command =
        new SqlCommand(Query, connection);
        connection.Open();
        SqlDataReader reader =
        command.ExecuteReader();
```

```
        List<ElectronicsProduct> list = new
        List<ElectronicsProduct>();

        while (reader.Read())
        {

            int id = (int)reader["ProductId"];
            String name = reader["ProductName"].ToString();
            String code = reader["ProductCode"].ToString();
            int secId = (int)reader["SectionId"];
            Double price =
            Convert.ToDouble(reader["Price"].ToString());
            String category = reader["Category"].ToString();
            String description = reader["Description"].ToString();
            String imgLink = reader["ImageLink"].ToString();
            int Quantity = (int)reader["Quantity"];
            String brandName = reader["BrandName"].ToString();
            list.Add(new ElectronicsProduct(id,name,
            code, secId, price, Quantity,
            category, description, imgLink, brandName));
        }

        return list;
    }

    public List<DailyNeedProduct> getDailyNeedProduct
    (String productCode)
    {
        SqlConnection connection =
        new SqlConnection(connectionString);
        String Query = "SELECT * FROM products
        WHERE SectionId = 3 AND ProductCode='" + productCode + "'";

        SqlCommand command = new
        SqlCommand(Query, connection);
        connection.Open();
        SqlDataReader reader = command.ExecuteReader();


        List<DailyNeedProduct> list =
        new List<DailyNeedProduct>();

        while (reader.Read())
        {

            int id = (int)reader["ProductId"];
            String name = reader["ProductName"].ToString();
            String code = reader["ProductCode"].ToString();
            int secId = (int)reader["SectionId"];
            Double price = Convert.ToDouble(reader["Price"]
            .ToString());
            String category = reader["Category"].ToString();
            String description = reader["Description"]
            .ToString();
            String imgLink = reader["ImageLink"].ToString();
            int Quantity = (int)reader["Quantity"];
            String brandName = reader["BrandName"].ToString();
```

```
            list.Add(new DailyNeedProduct(id, name, code,
            secId, price, Quantity,
            category, description, imgLink, brandName));
        }

        return list;
    }

    public List<MobileProduct> getMobileProduct
    (String productCode)
    {
        SqlConnection connection =
        new SqlConnection(connectionString);
        String Query = "SELECT * FROM products
        WHERE SectionId = 4 AND ProductCode='" +
        productCode + "'";

        SqlCommand command =
        new SqlCommand(Query, connection);
        connection.Open();
        SqlDataReader reader = command.ExecuteReader();


        List<MobileProduct> list =
        new List<MobileProduct>();

        while (reader.Read())
        {

            int id = (int)reader["ProductId"];
            String name = reader["ProductName"].ToString();
            String code = reader["ProductCode"].ToString();
            int secId = (int)reader["SectionId"];
            Double price = Convert.ToDouble(reader["Price"]
            .ToString());
            String category = reader["Category"].ToString();
            String description = reader["Description"]
            .ToString();
            String imgLink = reader["ImageLink"].ToString();
            int Quantity = (int)reader["Quantity"];
            String brandName = reader["BrandName"].ToString();
            list.Add(new MobileProduct(id, name, code, secId,
            price, Quantity, category, description,
            imgLink, brandName));
        }

        return list;
    }


    public List<DailyNeedProduct> getADailyNeedProduct()
    {
        SqlConnection connection = new
        SqlConnection(connectionString);
        String Query = "SELECT * FROM products
        WHERE SectionId = 3";

        SqlCommand command = new
```

```
        SqlCommand(Query, connection);
        connection.Open();
        SqlDataReader reader = command.ExecuteReader();


        List<DailyNeedProduct> list =
        new List<DailyNeedProduct>();

        while (reader.Read())
        {
            int id = (int)reader["ProductId"];
            String name = reader["ProductName"].ToString();
            String code = reader["ProductCode"].ToString();
            int secId = (int)reader["SectionId"];
            Double price = Convert.ToDouble(reader["Price"]
            .ToString());
            String category = reader["Category"].ToString();
            String description = reader["Description"].ToString();
            String imgLink = reader["ImageLink"].ToString();
            int Quantity = (int)reader["Quantity"];
            String brandName = reader["BrandName"].ToString();
            list.Add(new DailyNeedProduct(id,name, code,
            secId, price, Quantity,
            category, description,
            imgLink, brandName));
        }

        return list;
    }

    public List<DailyNeedProduct> getADailyNeedProduct
    (String productCode)
    {
        SqlConnection connection =
        new SqlConnection(connectionString);
        String Query = "SELECT * FROM
        products WHERE SectionId = 3
        AND ProductCode = '"+ productCode+"'";

        SqlCommand command = new SqlCommand(Query, connection);
        connection.Open();
        SqlDataReader reader = command.ExecuteReader();


        List<DailyNeedProduct> list = new List<DailyNeedProduct>();

        while (reader.Read())
        {
            int id = (int)reader["ProductId"];
            String name = reader["ProductName"].ToString();
            String code = reader["ProductCode"].ToString();
            int secId = (int)reader["SectionId"];
            Double price = Convert.ToDouble(reader["Price"].
            ToString());
            String category = reader["Category"].
            ToString();
            String description = reader["Description"].
            ToString();
```

```
            String imgLink = reader["ImageLink"]
            .ToString();
            int Quantity = (int)reader["Quantity"];
            String brandName = reader["BrandName"].ToString();
            list.Add(new DailyNeedProduct(id,name, code,
            secId, price, Quantity, category,
            description, imgLink, brandName));
        }

        return list;
    }

    public List<MobileProduct> getAllMobileProduct()
    {
        SqlConnection connection =
        new SqlConnection(connectionString);
        String Query = "SELECT * FROM products
        WHERE SectionId = 4";

        SqlCommand command = new SqlCommand(Query, connection);
        connection.Open();
        SqlDataReader reader = command.ExecuteReader();


        List<MobileProduct> list = new List<MobileProduct>();

        while (reader.Read())
        {
            int id = (int)reader["ProductId"];
            String name = reader["ProductName"].ToString();
            String code = reader["ProductCode"].ToString();
            int secId = (int)reader["SectionId"];
            Double price = Convert.ToDouble(reader["Price"]
            .ToString());
            String category = reader["Category"].ToString();
            String description = reader["Description"].ToString();
            String imgLink = reader["ImageLink"].ToString();
            int Quantity = (int)reader["Quantity"];
            String brandName = reader["BrandName"].ToString();
            list.Add(new MobileProduct(id,name, code,
            secId, price, Quantity,
            category, description,
            imgLink, brandName));
        }

        return list;
    }


    public List<MobileProduct> getAMobileProduct(String productCode)
    {
        SqlConnection connection = new SqlConnection(connectionString);
        String Query = "SELECT * FROM products
        WHERE SectionId = 4 AND
        ProductCode ='"+productCode+"'";

        SqlCommand command = new SqlCommand(Query, connection);
        connection.Open();
```

```
    SqlDataReader reader = command.ExecuteReader();


    List<MobileProduct> list = new List<MobileProduct>();

    while (reader.Read())
    {
        int id = (int)reader["ProductId"];
        String name = reader["ProductName"].ToString();
        String code = reader["ProductCode"].ToString();
        int secId = (int)reader["SectionId"];
        Double price =
        Convert.ToDouble(reader["Price"].ToString());
        String category = reader["Category"].ToString();
        String description = reader["Description"].ToString();
        String imgLink = reader["ImageLink"].ToString();
        int Quantity = (int)reader["Quantity"];
        String brandName = reader["BrandName"].ToString();
        list.Add(new MobileProduct(id,name, code,
        secId, price, Quantity, category,
        description, imgLink, brandName));
    }

    return list;
}



public List<ClothingProduct>
getaClothingProduct(String productCode)
{
    SqlConnection connection =
    new SqlConnection(connectionString);
    String Query = "SELECT * FROM products
    WHERE ProductCode ='" + productCode + "'";

    SqlCommand command =
    new SqlCommand(Query, connection);
    connection.Open();
    SqlDataReader reader =
    command.ExecuteReader();


        List<ClothingProduct> list =
        new List<ClothingProduct>();

        while (reader.Read())
        {
        int id = (int)reader["ProductId"];
        String name = reader["ProductName"].ToString();
        String code = reader["ProductCode"].ToString();
        int secId = (int)reader["SectionId"];
        Double price = Convert.ToDouble(reader["Price"]
        .ToString());
        String category = reader["Category"]
        .ToString();
        String description =
        reader["Description"].ToString();
```

```
            String imgLink = reader["ImageLink"].ToString();
            int lCount = (int)reader["LCount"];
            int mCount = (int)reader["MCount"];
            int xlCount = (int)reader["XLCount"];
            int xxlCount = (int)reader["XXLCount"];
            String brandName = reader["BrandName"].ToString();

            list.Add(new ClothingProduct(id,name, code,
            secId, price, category,
            description, imgLink,
            brandName, lCount, mCount,
            xlCount, xxlCount));
            }

            return list;

        }




        public List<Product> GetAllProduct()
        {
            SqlConnection connection =
            new SqlConnection(connectionString);
            String Query = "SELECT * FROM products";
            SqlCommand command =
            new SqlCommand(Query, connection);
            connection.Open();
            SqlDataReader reader =
            command.ExecuteReader();

            List<Product> list =
            new List<Product>();

            while (reader.Read())
            {
                int id = Convert.ToInt32(reader["ProductId"]);
                String name = reader["ProductName"].ToString();
                String code = reader["ProductCode"].ToString();
                int secId = (int)reader["SectionId"];
                double price =Convert.ToDouble(reader["Price"].
                ToString());
                String category = reader["Category"].ToString();
                String description = reader["Description"].
                ToString();
                String imgLink = reader["ImageLink"].
                ToString();
                String lCount = reader["LCount"].ToString();
                String mCount = reader["MCount"].ToString();
                String xlCount = reader["XLCount"].ToString();
                String xxlCount = reader["XXLCount"].ToString();
                String quantiy = reader["Quantity"].ToString();
                String brandName = reader["BrandName"].ToString();
                list.Add(new Product(id,name,code,
                secId,price,category,
                description,imgLink,brandName));
```

```
        }

        return list;
    }
    public List<Product> GetAllProduct(int pid)
    {
        SqlConnection connection =
        new SqlConnection(connectionString);
        String Query = "SELECT * FROM products
        WHERE ProductId ="+ pid;
        SqlCommand command =
        new SqlCommand(Query, connection);
        connection.Open();
        SqlDataReader reader =
        command.ExecuteReader();

        List<Product> list =
        new List<Product>();

        while (reader.Read())
        {
            int id = Convert.ToInt32(reader["ProductId"]);
            String name = reader["ProductName"].ToString();
            String code = reader["ProductCode"].ToString();
            int secId = (int)reader["SectionId"];
            double price = Convert.ToDouble(reader["Price"].
            ToString());
            String category = reader["Category"].
            ToString();
            String description = reader["Description"]
            .ToString();
            String imgLink = reader["ImageLink"]
            .ToString();
            String lCount = reader["LCount"].ToString();
            String mCount = reader["MCount"].ToString();
            String xlCount = reader["XLCount"].ToString();
            String xxlCount = reader["XXLCount"].ToString();
            String quantiy = reader["Quantity"].ToString();
            String brandName = reader["BrandName"].ToString();
            list.Add(new Product(id, name, code,
            secId, price, category,
            description, imgLink, brandName));
        }

        return list;
    }

    public int UpdateProductInfo(ClothingProduct product)
    {
        SqlConnection connection =
        new SqlConnection(connectionString);
        String Query = "UPDATE products SET
        productName = '" + product.ProductName + "',
        ProductCode='" + product.ProductCode +
            "',Price=" + product.Price + ",Category ='" +
            product.Category + "',Description='"
            + product.Description + "',LCount="
            + product.LCount + ",MCount=" +
```

```
            product.MCount + ",XLCount=" + product.XLCount + "
            ,XXLCount=" + product.XXLCount + ",BrandName='"
            + product.BrandName + "' WHERE ProductId="
            +product.ProductId;
        SqlCommand command = new SqlCommand(Query, connection);
        connection.Open();
        int res = command.ExecuteNonQuery();
        connection.Close();

        return res;

    }

    public int UpdateProductInfo
    (ElectronicsProduct product)
    {
        SqlConnection connection = new
        SqlConnection(connectionString);
        String Query = "UPDATE products
        SET productName = '" + product.ProductName + "'
        ,ProductCode='" + product.ProductCode +
            "',Price=" + product.Price + ",Category ='"
            + product.Category + "',Description='" +
            product.Description + "',Quantity=" +
            product.Quantity +
           ",BrandName='" + product.BrandName + "'
           WHERE ProductId=" + product.ProductId;
        SqlCommand command =
        new SqlCommand(Query, connection);
        connection.Open();
        int res = command.ExecuteNonQuery();
        connection.Close();

        return res;
    }

    public int UpdateProductInfo
    (DailyNeedProduct product)
    {
        SqlConnection connection =
        new SqlConnection(connectionString);
        String Query = "UPDATE products
        SET productName = '" +
        product.ProductName + "',ProductCode='" + product.ProductCode +
            "',Price=" + product.Price + ",Category ='" +
            product.Category + "',Description='" +
            product.Description + "',Quantity="
            + product.Quantity +
           ",BrandName='" + product.BrandName +
           "' WHERE ProductId=" + product.ProductId;
        SqlCommand command =
        new SqlCommand(Query, connection);
        connection.Open();
        int res = command.ExecuteNonQuery();
        connection.Close();

        return res;
    }
```

```
        public int UpdateProductInfo
        (MobileProduct product)
        {
            SqlConnection connection =
            new SqlConnection(connectionString);
            String Query = "UPDATE products SET productName = '" +
            product.ProductName + "',ProductCode='" +
            product.ProductCode +
                "',Price=" + product.Price + ",Category ='" +
                product.Category + "',Description='" +
                product.Description + "',Quantity=" +
                product.Quantity +
               ",BrandName='" + product.BrandName + "'
               WHERE ProductId=" + product.ProductId;
            SqlCommand command = new SqlCommand(Query, connection);
            connection.Open();
            int res = command.ExecuteNonQuery();
            connection.Close();

            return res;
        }

        public int DeleteProduct(int productId)
        {
            SqlConnection connection =
            new SqlConnection(connectionString);
            String Query = "DELETE
            FROM products WHERE ProductId=" + productId;
            SqlCommand command = new SqlCommand(Query,connection);
            connection.Open();
            int res = command.ExecuteNonQuery();
            connection.Close();
            return res;
        }




    }
}
```

A.2.4 SectionGetway.cs

```
using Biponee.Models;
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Linq;
using System.Web;
using System.Web.Configuration;

namespace Biponee.DLL
{
    public class SectionGetway
    {
        String connectionString = WebConfigurationManager.
        ConnectionStrings["biponeeDbConnectioon"]
        .ConnectionString;
        public List<SectionC> getAllSections()
        {
            SqlConnection connection =
            new SqlConnection(connectionString);
            String Query = "SELECT * FROM sections";

            SqlCommand command = new SqlCommand(Query, connection);
            connection.Open();
            SqlDataReader reader = command.ExecuteReader();

            List<SectionC> list = new List<SectionC>();

            while (reader.Read())
            {
                int id = Convert.ToInt32(reader["SectionID"]);
                String name = reader["SectionName"].ToString();


                list.Add(new SectionC(id,name));
            }

            return list;
        }
    }
}
```

A.2.5 UserGetway.cs

```
using Biponee.Models;
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Linq;
using System.Web;
using System.Web.Configuration;

namespace Biponee.DLL
{
    public class UserGetway
    {
        String connectionString = WebConfigurationManager.
        ConnectionStrings["biponeeDbConnectioon"].ConnectionString;
        public int InsertUser(UserC user)
        {


                SqlConnection connection =
                new SqlConnection(connectionString);
                String Query = "INSERT INTO users VALUES
                ('" + user.FirstName + "','" + user.LastName + "','" +
                user.Email + "','" + user.Password +"')";
                SqlCommand command = new SqlCommand(Query, connection);
                connection.Open();

                int res = command.ExecuteNonQuery();
                connection.Close();
                return res;

        }

        public List<UserC> getUser(String email,String password)
        {

            SqlConnection connection = new
            SqlConnection(connectionString);
            String Query = "SELECT * FROM
            users WHERE Email = '" + email + "'
            AND Password = '" + password + "'";

            SqlCommand command =
            new SqlCommand(Query, connection);
            connection.Open();
            SqlDataReader reader =
            command.ExecuteReader();

            List<UserC> list = new List<UserC>();

            while (reader.Read())
            {
                int id = Convert.ToInt32(reader["UserId"]);
                String firstName = reader["FirstName"].ToString();
                String lastName = reader["LastName"].ToString();
                String Email = reader["Email"].ToString();
```

```
                String Password = reader["Password"].ToString();

                list.Add(new UserC(id, firstName,
                lastName, Email, Password));
            }

            return list;
        }

        public List<UserC> getUser(String email)
        {

            SqlConnection connection = new
            SqlConnection(connectionString);
            String Query = "SELECT * FROM users
            WHERE Email = '" + email + "'";

            SqlCommand command = new
            SqlCommand(Query, connection);
            connection.Open();
            SqlDataReader reader =
            command.ExecuteReader();

            List<UserC> list = new List<UserC>();

            while (reader.Read())
            {
                int id = Convert.ToInt32(reader["UserId"]);
                String firstName = reader["FirstName"]
                .ToString();
                String lastName = reader["LastName"].
                ToString();
                String Email = reader["Email"].ToString();
                String Password = reader["Password"].
                ToString();

                list.Add(new UserC(id, firstName,
                lastName, Email, Password));
            }

            return list;
        }

    }
}
```