# SYSTEMS ANALYSIS AND DESIGN

## 4.0 INTRODUCTION

System analysis and design deal with planning the development of information systems through understanding and specifying in detail what a system should do and how the components of the system should be implemented and work together. System analysis and design solve business problems through analyzing the requirements of information systems and designing such systems by applying analysis and design techniques.

System analysis and design is the most essential phase in the development of a system since the logical system design arrived at as a result of systems analysis which is in turn converted into physical system design.

This chapter will outline the general system and application architecture, requirements specification, functional and non-functional requirements, use case modelling, the database design, Entity Relationship diagram, Normalization, and User interface design.

## 4.1 GENERAL SYSTEM ARCHITECTURE

The architecture of applications is usually broken into logical chunks called "tiers", where every tier is assigned a role. A “tier” can also be referred to as a “layer”. There are three layers involved in the application namely **Presentation Layer**, **Business Layer** and **Data Layer**. Each layer is explained in detailed below:

- **Presentation Layer:**
  It is also known as Client layer. Top most layer of an application. This is the layer we see when we use a software. By using this layer, we can access the webpages. The main functionality of this layer is to communicate with Application layer. This layer passes the information which is given by the user in terms of keyboard actions, mouse clicks to the Application Layer. For example, login page of Gmail where an end user could see text boxes and buttons to enter user id, password and to click on sign-in.
  In simple words, it is to view the application.
- **Application Layer:**
  It is also known as Business Logic Layer which is also known as logical layer. As per the Gmail login page example, once user clicks on the login button, Application layer interacts with Database layer and sends required information to the Presentation layer. It

controls an application’s functionality by performing detailed processing. This layer acts as a mediator between the Presentation and the Database layer. Complete business logic will be written in this layer.
In simple words, it is to perform operations on the application.

- **Data Layer:**
  The data is stored in this layer. Application layer communicates with Database layer to retrieve the data. It contains methods that connects the database and performs required action e.g.: insert, update, delete etc.
  In simple words, it is to share and retrieve the data.

There are four categories of application architectures which all have the four layers explained above. The categories of the application architecture are:

### 4.1.1 ONE TIER ARCHITECTURE

One tier architecture has all the layers such as Presentation, Business, Data Access layers in a single software package. All traditional applications consist only of 1 tier, which resides on the client machine. Applications which handles all the three tiers such as MP3 player, MS Office are come under one tier application. The data is stored in the local system or a shared drive. This tier is also known as **Standalone application.**

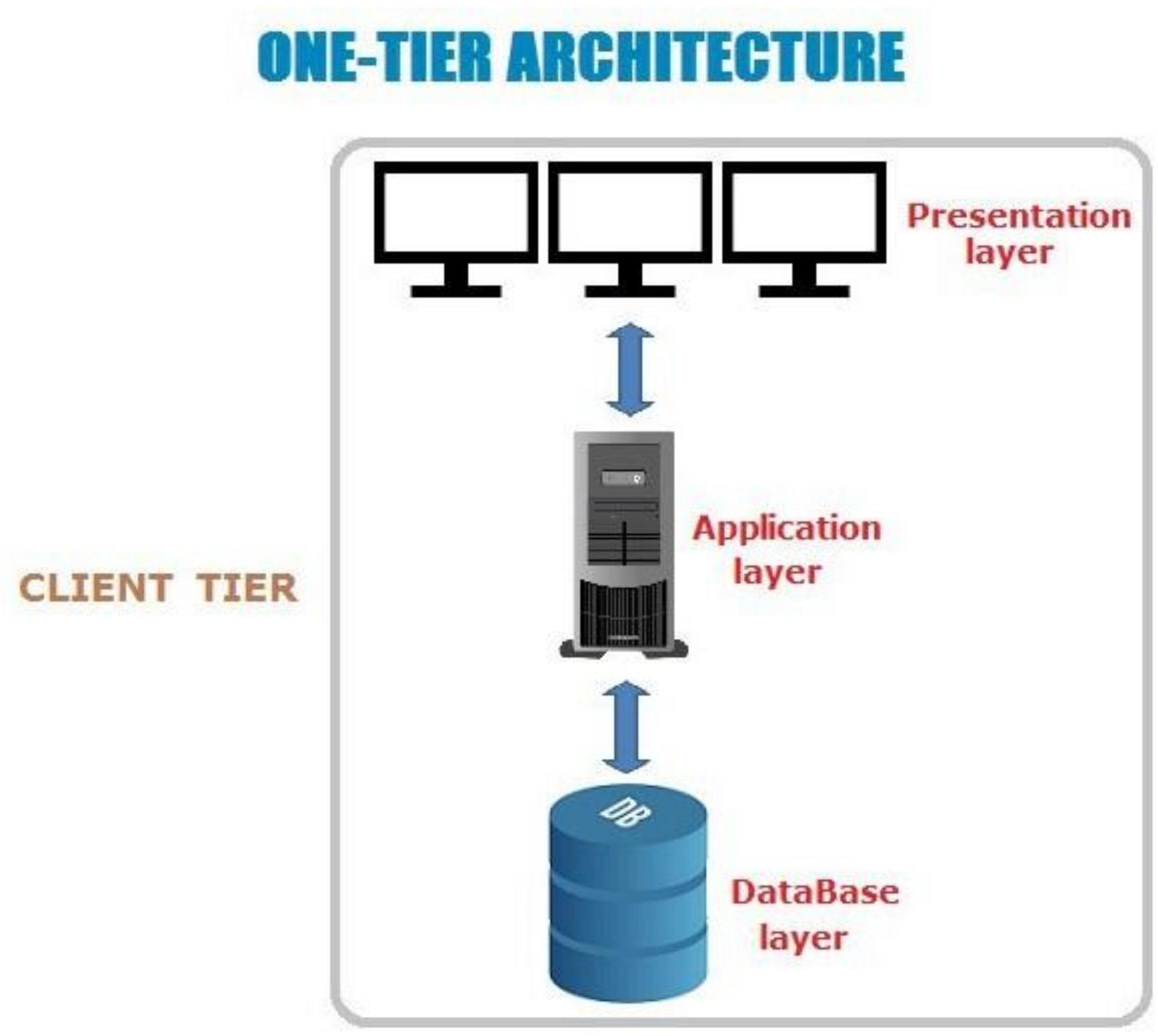


*Figure 4.0 One-Tier Architecture (Standalone Application)*

*Source:* (Gaur, 2018)

### 4.1.2 TWO TIER ARCHITECTURE

The Two-tier architecture is divided into two parts namely **Client Application (Client Tier)** and **Database (Data Tier).** Client system handles both Presentation and Application layers and Server system handles Database layer. The communication takes place between the Client and the Server. Client system sends the request to the Server system and the Server system processes the request and sends back the data to the Client System. It is also known as **client-server application**.

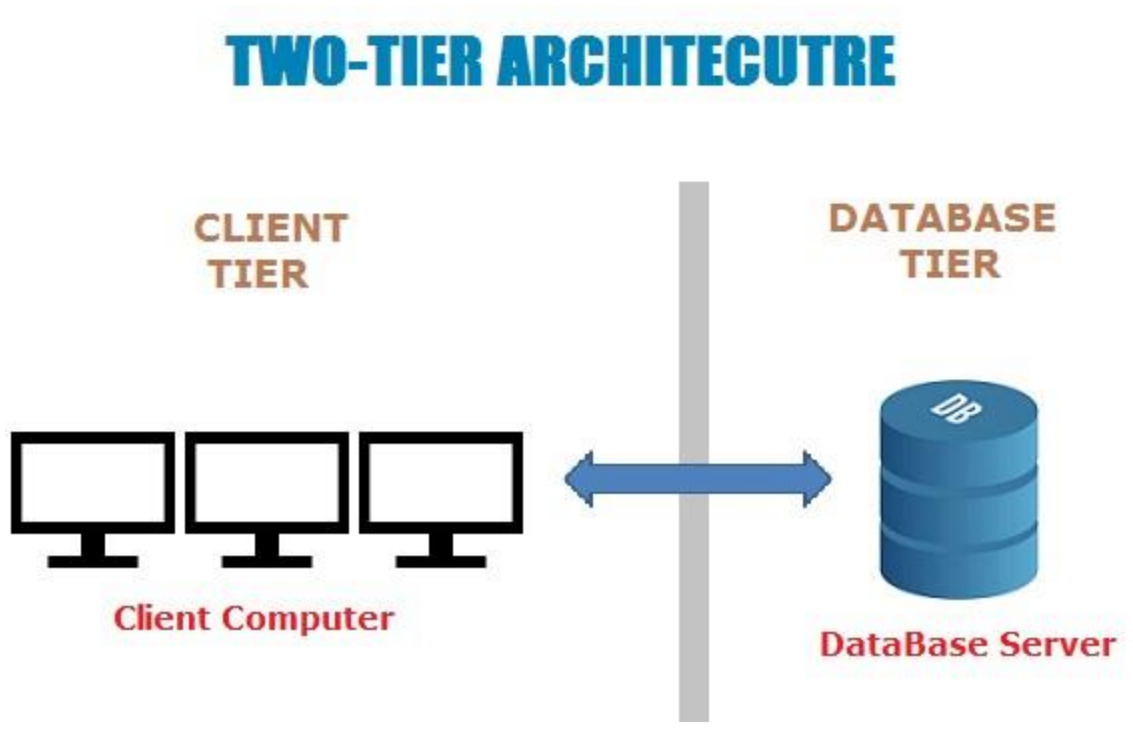


*Figure 4.1 Two-tier Architecture (Client-Server Application)*

*Source:* (Gaur, 2018)

## 4.1.3 THREE TIER ARCHITECTURE

The Three-tier architecture is divided into three parts namely: **Presentation layer (Client Tier), Application layer (Business Tier)** and **Database layer (Data Tier)**. Client system handles Presentation layer, Application server handles Application layer and Server system handles Database layer. A web browser is the first tier (presentation), an engine using some dynamic Web content technology (such as ASP, CGI, ColdFusion, Dart, JSP/Java, Node.js, PHP, Python or Ruby on Rails) is the middle tier (application logic), and a database is the third tier (storage). The web browser sends requests to the middle tier, which services them by making queries and updates against the database and generates a user interface. It is also known as **Web Based application.**

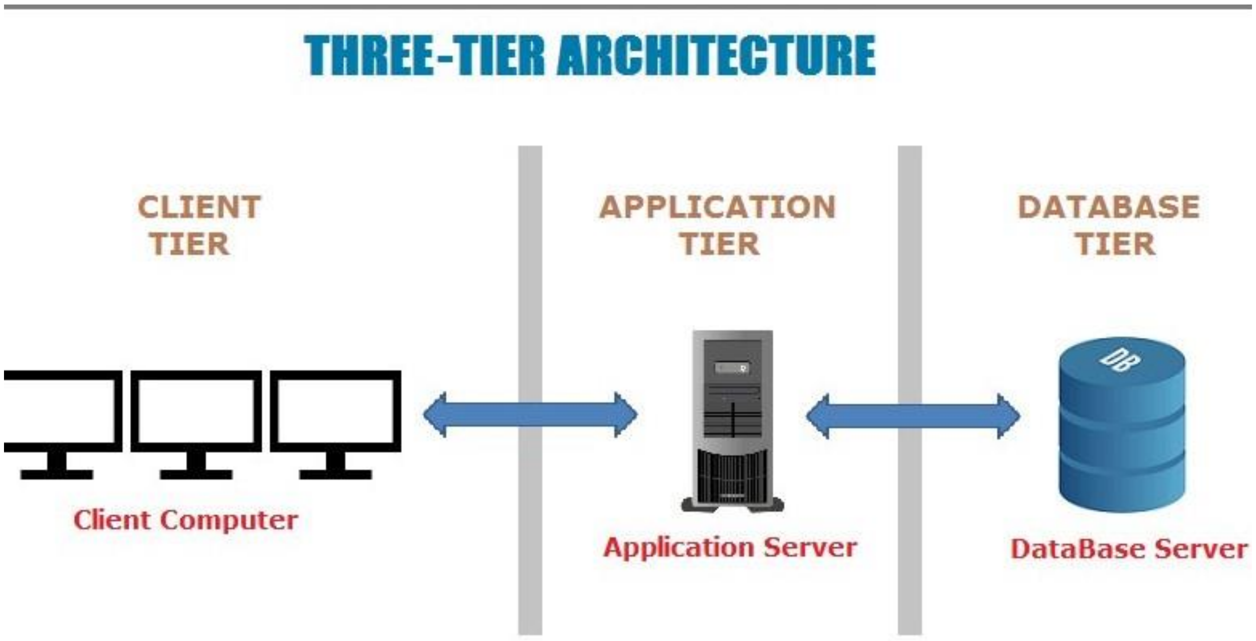


*Figure 4.2 Three-Tier Architecture (Web-Based Application)*

*Source:* (Gaur, 2018)

## 4.1.4 N-TIER ARCHITECTURE

It is similar to three tier architectures but number of application servers are increased and represented in individual tiers in order to distributed the business logic so that the logic will be distributed. N-Tier application is also known as **Distributed application**.

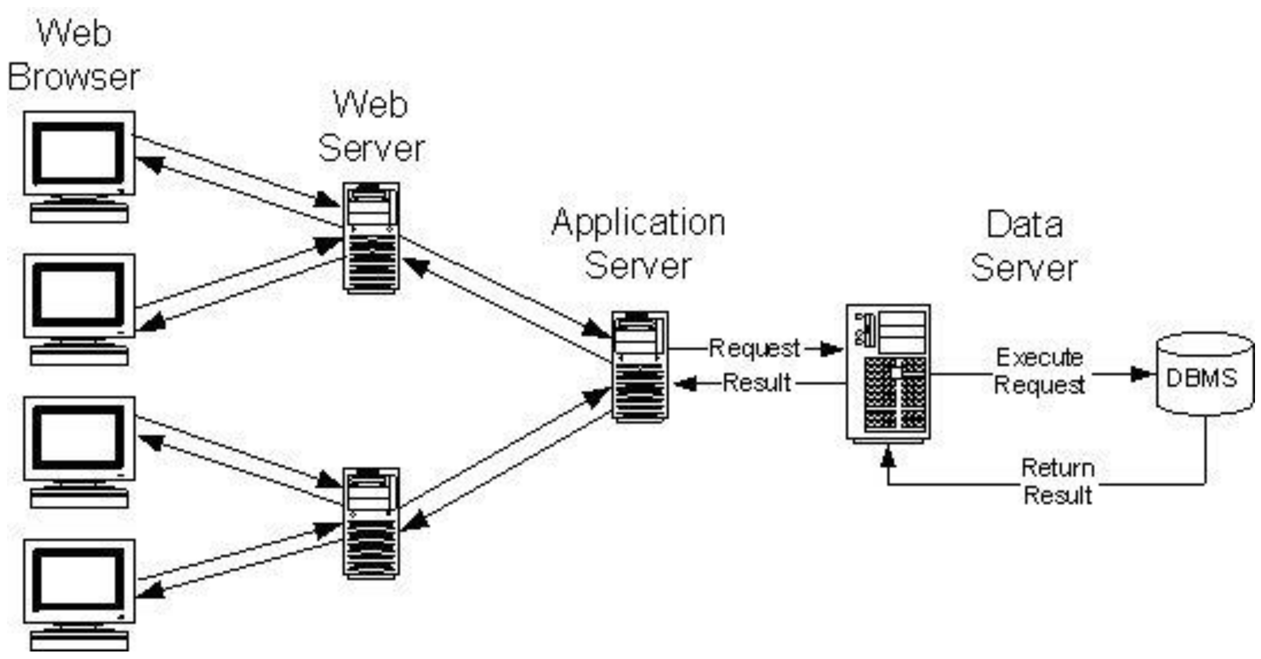


*Figure 4.3 N-Tier Application (Distributed Application)*

*Source:* (Kramek, 2018)

Web applications lend themselves to an n-tiered approach by nature. Though many variations are possible, the most common structure is the three-tiered application. This project uses the three-tiered architecture because it is made up of three layers; the **presentation layer** would be the **web browser**; the **application layer** would be the **Django framework** and the **third layer** which is the **database layer** would be **Postgresql**.

## 4.2 GENERAL APPLICATION ARCHITECTURE

### 4.2.1 THE MODEL-VIEW-CONTROLLER DESIGN PATTERN

Although originally developed for desktop computing, MVC has been widely adopted as an architecture for World Wide Web applications in major programming languages. Several web frameworks have been created that enforce the pattern. These software frameworks vary in their interpretations, mainly in the way that the MVC responsibilities are divided between the client and server. MVC is short for **Model, View, and Controller.** MVC is a popular way of organizing code. The big idea behind MVC is that each section of code has a purpose, and those purposes are different. Some of the code holds the data of an app, some of the code makes the app look nice, and some of the code controls how the app functions. MVC is a way to organize code's core functions into their own, neatly organized boxes. This makes thinking about the app, revisiting the app, and sharing the app with others much easier and cleaner.

MVC has been around as a concept for a long time but has seen exponential growth since the advent of the Internet because it is the best way to design client-server applications.

- The **model(M)** is a model or representation of your data. It's not the actual data, but an interface to the data. The model allows you to pull data from your database without knowing the intricacies of the underlying database. The model usually also provides an abstraction layer with your database, so that you can use the same model with multiple databases.

- The **view(V)** is what you see. It's the presentation layer for your model. On your computer, the view is what you see in the browser for a Web app, or the UI for a desktop app. The view also provides an interface to collect user input.

- The **controller(C)** controls the flow of information between the model and the view. It uses programmed logic to decide what information is pulled from the database via the model and what information is passed to the view. It also gets information from the user via the view and implements business logic: either by changing the view, or modifying data through the model, or both.

This application follows the MVC pattern closely, however it does use its own logic in the implementation because the Django framework uses a different logic. Because the "C" is handled by the framework itself and also a lot happens in the Django models, templates and views, Django is often referred to as an **MTV framework.** In the MTV development pattern:

- **M stands for "Model,"** the data access layer. This layer contains anything and everything about the data: how to access it, how to validate it, which behaviors it has, and the relationships between the data.
- **T stands for "Template,"** the presentation layer. This layer contains presentation-related decisions: how something should be displayed on a Web page or other type of document.

- **V stands for "View,"** the business logic layer. This layer contains the logic that accesses the model and defers to the appropriate template(s). You can think of it as the bridge between models and templates.

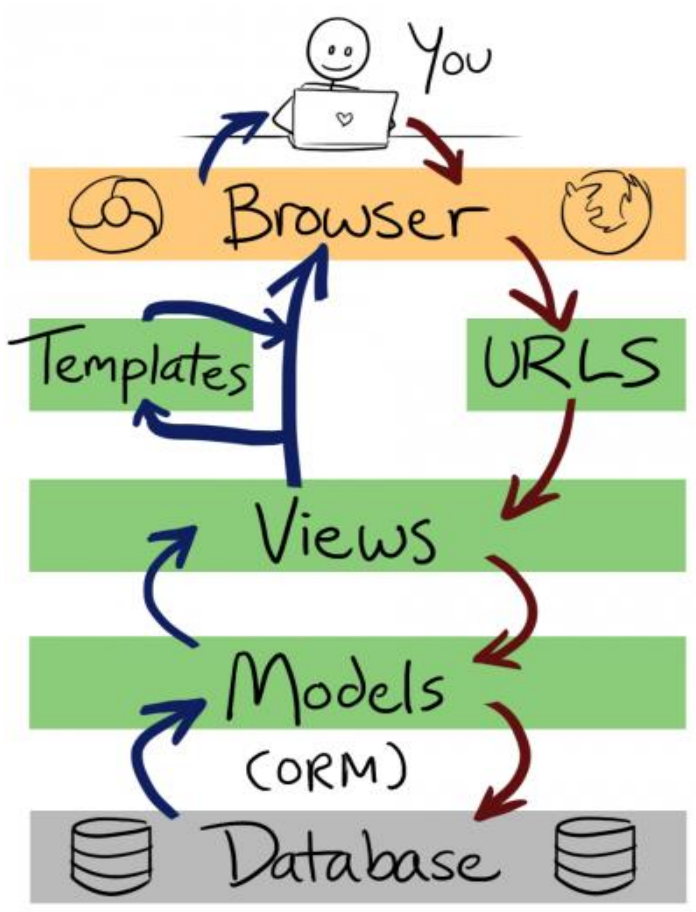


*Figure 4.4 E-Shopper Application Architecture (MVT Application Design Patten)*

*Source:* (Openhatch, 2018)

## 4.3 REQUIREMENT SPECIFICATION

A System Requirements Specification (SRS) (also known as a Software Requirements Specification) is a document or set of documentation that describes the features and behavior of a system or software application. It includes a variety of elements that attempts to define the intended functionality required by the customer to satisfy their different users. What the software will do and how it will be expected to function is fully described under requirements specification. In addition to specifying how the system should behave, the specification also defines at a high-level the main business processes that will be supported, what simplifying assumptions have been made and what key performance parameters will need to be met by the system. It outlines functional and non- functional requirements and may include a set of use cases that describe user interactions that the system must provide. Requirements specification allows a thorough analysis of requirements before design can start hence, reducing later redesign.

### 4.3.1 FUNCTIONAL REQUIREMENTS

Functional requirements describe in detail a system's intended capabilities, appearance and interactions with users. It usually consists of a hierarchical organization of requirements, with the business/functional requirements at the highest-level and the detailed system requirements listed as their child items. The functional requirements for a system usually involves the user interface and describe each of the possible user input actions and the system's response actions. Functional requirements therefore specify particular results of a system and it drives the application architecture of a system. Generally, the requirements are written as statements such as "System needs the ability to do x" with supporting detail and information included as necessary. It serves as a kind of guideline and continuing reference point as the system is being developed.

The system will have the following functional requirements:

- An administrator will be able to login and customize the system to the needs of the restaurant.
- Through the administrator's panel, the system allows the manager to generate reports of orders, delete products and orders, and approve comments among others.
- The administrator can create multiple accounts for different users to perform specific actions like accepting orders, managing product stock etc.
- The administrator can check product inventory, accept orders and approve product reviews.
- Online Shoppers will be able to create an account so as to login to their portal and see their Order history etc.
- Online Shoppers will be able to place orders and receive notifications of their others in their emails.

- Online Shoppers can choose the language of their choice.
- Online Shoppers can review products with comments.
- Online Shoppers can also pay online before their orders are placed successfully

### 4.3.2 NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements essentially specify how a system should behave or work and also are a constraint upon the system's behavior. They specify criteria that judge the operation of a system rather than specific behaviors. They also describe various attributes which affect the functionality's effectiveness. It specifies the requirements that essentially embody the technical environment that the product needs to operate in and include the technical constraints that it needs to operate under. These technical requirements are critical in determining how the higher-level functional requirements will get decomposed into the more specific system requirements. Non-functional requirements do not alter a system's functionality that is, the functional requirements remain the same regardless of the attributes attached to them. Non-functional requirements make up a significant part of the specification in that, users or clients may judge a system based on its non-functional requirements.

Below are the non-functional requirements our system will achieve:

- **Performance**

The system will be an interactive one hence the delays involved will be less in that there are no immediate delays in every action-response of the system.

- **Reliability**

As the system is meant for online shopping, the system will be reliable in that it will consistently perform according to its specification so as to enable users make place orders as and when they want to without any system failure.

- **Maintainability**

A system should be developed in such a way that it can evolve to meet the changing needs of a customer. The system will be developed in such a way that it can be customized to meet a particular vendor's needs or preference.

- **Ease of use**

The system will ensure ease of use in that little training time will required to know how to use the system. The user interface will also be friendly so users will have ease when using the system.

- **Scalability**

It is the ability of a system to continue to function well when it is changed in size or volume to meet a user need. The system will be developed to fit onto any device such as a smartphone, laptop, tablet or desktop.

- **Security**

It is a system attribute that reflects the ability of a system to protect itself against external attacks, which may be deliberate or accidental. The system will be developed in such a way that proper login mechanism would be put in place to protect user accounts.

- **Safety**

The system will ensure safety so that information about orders placed is securely transmitted to the server without any changes in the information. PayPal Payment System is recognized internationally for it secure environment for online secured payments.

## 4.4 USE CASE MODELLING

UML (Unified Modeling Language) is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. A use-case model is a model of how different types of users interact with the system to solve a problem. It therefore describes the goals of the users, the interactions between the users and the system and the required behavior in satisfying these goals.

### 4.4.1 USE CASE DIAGRAMS

A use-case diagram describes a system's functional requirements in terms of use cases. It is a model of the system's intended functionality (use cases) and its environment (actors). Use cases enable you to relate what you need from a system to how the system delivers on those needs.

Think of a use-case model as a menu, much like the menu you'd find in a restaurant. By looking at the menu, you know what's available to you, the individual dishes as well as their prices. You also know what kind of cuisine the restaurant serves: Italian, Mexican, Chinese, and so on. By looking at the menu, you get an overall impression of the dining experience that awaits you in that restaurant. The menu, in effect, "models" the restaurant's behavior. It shows a subset of the model elements relevant for a particular purpose. Below is a table of use case symbols and their representations.

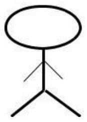
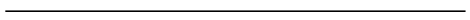
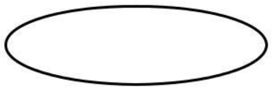
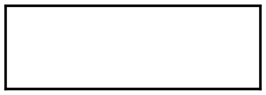
| SYMBOL | TERM AND REPRESENTATION |
|---|---|
|  | **Actor:** An actor represents roles for users of a system, including human users and other systems. An actor is external to a system. |
|  | **Association:** An association correspond to a sequence of actions between the actors and use case in achieving the use case. |
|  | **Use case:** A use case represents a user goal that can be achieved by accessing the system. |
|  | **System Boundary:** The system boundary defines the confines or scope of a system within which the use cases are placed |

*Table 4.0 Use case symbols and their meanings.*

The actors involved with the system are:

**Customer:** This actor is one of the main beneficiary of the system. The customer can order products using the system.

**Manager:** This actor is also another beneficiary of the system. Customizing the system to his preference, editing products, approving orders and product reviews are some of the activities of this actor.

**System:** This actor is responsible for generating reports and sending confirmation messages among others. It however requires the administrator inputs to generate the reports.

Below are the use case diagrams for the system together with their descriptions:

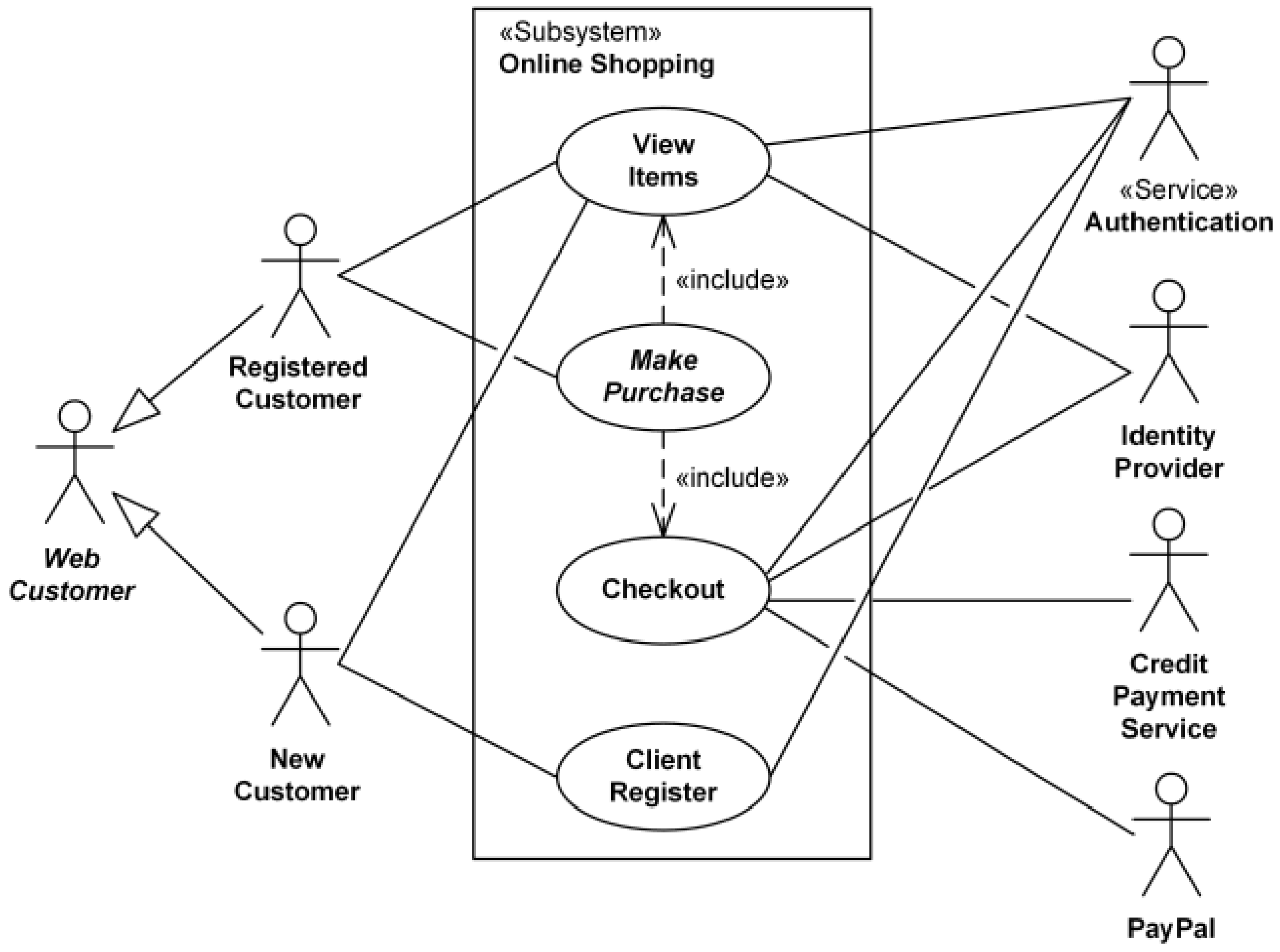


*Figure 4.5 Use case diagram for customer*

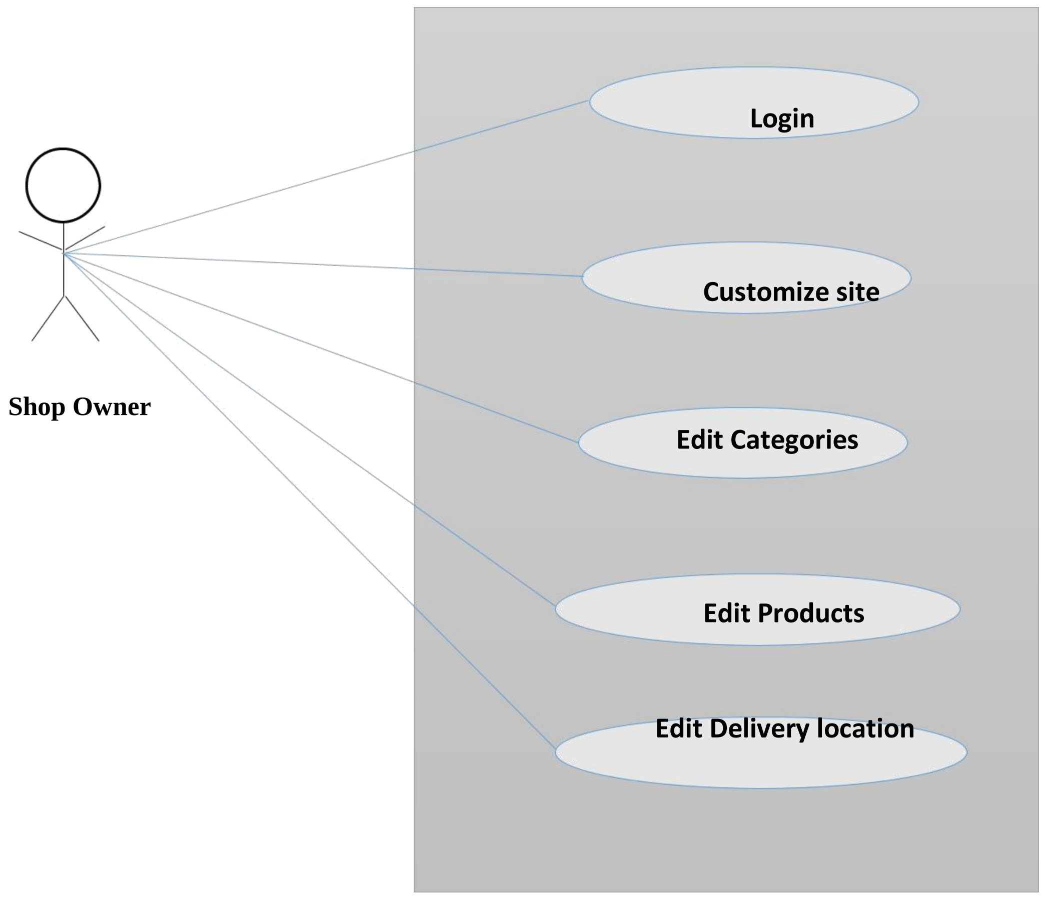


*Figure 4.6 Use case diagram for Shop Owner*

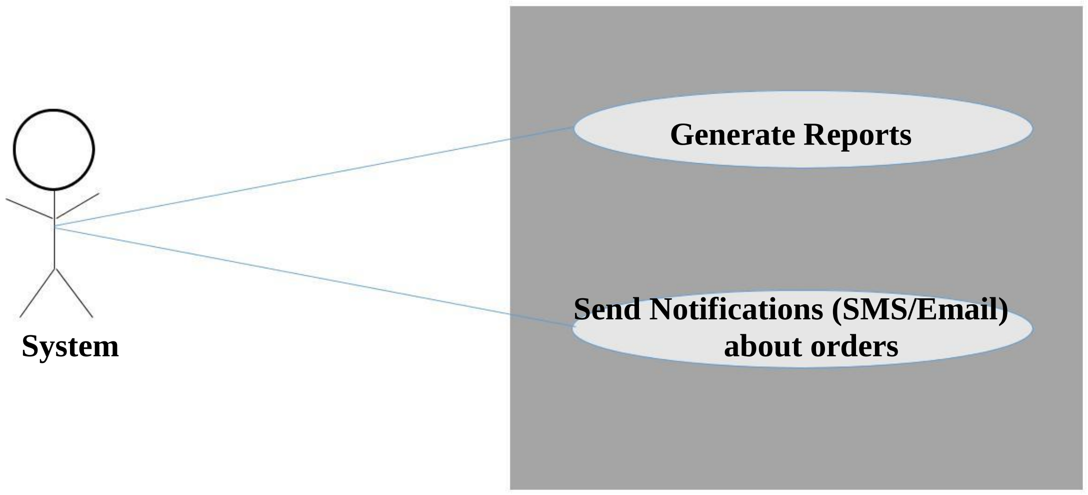


*Figure 4.7 Use case diagram for system*

**Application Use Case Scenarios**

*Table 4.1 Customer Registration*

| **Use Case Name** | **Register** |
|---|---|
| Actor | Customer |
| Description | Enables new customers to create an account by entering their credentials. |
| Steps Performed | 1. Click on sign up<br>2. Enter personal details and password.<br>3. Click on sign up button. |

*Table 4.2 Customer login*

| **Use Case Name** | **Login** |
|---|---|
| Actor | Customer |
| Precondition | Customer should have already registered. |
| Description | Enables existing customers to have access to the system's functionality using valid credentials. |
| Steps Performed | 1. Click on login<br>2. Enter username and password. |

| | |
|---|---|
| | 3. Click on login. |

*Table 4.3 View Products*

| **Use Case Name** | **View Products** |
|---|---|
| Actor | Customer |
| Description | Enables a customer to view products available together with their prices. |
| Steps Performed | Go to the website of the using a web browser. |

*Table 4.4 Placing Orders*

| **Use Case Name** | **Placing an Order** |
|---|---|
| Actor | Customer |
| Precondition | Customer can login or choose not to login. |
| Description | Enables customers to order products which is successful after they have proceeded with the |

| | |
|---|---|
| | payment. |
| Steps Performed | 1. Enter quantity<br>2. Click on add to cart button to add product to cart.<br>3. Click on Proceed to checkout.<br>4. Enter confirmation and payment details and click on Place now.<br>5. Proceed with payment using either debit card. |

*Table 4.5 Manager Login*

| **Use Case Name** | **Login** |
|---|---|
| Actor | Manager |
| Description | Enables a manager to access the system's functionality available to perform managerial duties. |
| Steps Performed | 1. Proceed to admin Panel.<br>2. Enter username and password.<br>3. Click on login. |

*Table 4.6 Editing Products*

| **Use Case Name** | **Edit Products** |
|---|---|
| Actor | Manager |
| Description | Enables a manager to make changes to available products such as adding new product, deleting product not available as well as upload images for the various products. |
| Precondition | Manager should login. |
| Steps Performed | 1. Click on products.<br>2. Make changes to the product by adding, editing or deleting. |

*Table 4.7 Edit Product Category*

| **Use Case Name** | **Edit Product category** |
|---|---|
| Actor | Manager |
| Description | Enables the manager to group products based on its category. |

| | |
|---|---|
| Precondition | Manager should login. |
| Steps Performed | 1. Click on categories<br>2. View all categories.<br>3. Make the necessary changes by adding, editing or deleting categories. |

*Table 4.8 Report Generation*

| **Use Case Name** | **Generate Report** |
|---|---|
| Actor | System |
| Description | Generates reports on periodic basis. |
| Steps Performed | 1. Click on report on the manager's interface.<br>2. Enter the period, that is start and end date desired for report. |

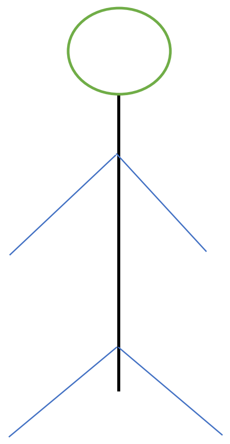
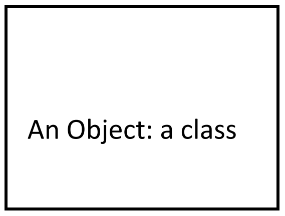
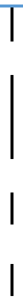
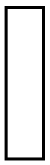
*Table 4.9 Send SMS/Email*

| **Use Case Name** | **Send SMS/Email** |
|---|---|
| Actor | System |
| Description | Sends order confirmation to the customer and order details to the vendor via Email. |
| Steps Performed | When an order is successful after payment, the system automatically sends an email to the customer. |

### 4.4.2 SEQUENCE DIAGRAM

Sequence diagrams are the most common kind of interaction diagrams that shows how actors and objects interact to realize a use case scenario. It focuses on the message interchange between a number of lifelines. The Sequence Diagram models the collaboration of objects based on a time sequence. It shows how the objects interact with others in a particular scenario of a use case. We normally draw a sequence diagram if we have a use case, to describe how the main components of the system interact. Then again sequence diagram helps us identify messages arriving at an interface of a component, to describe how the internal parts of the component interact.

*Table 4.10 Sequence diagram symbols and their meanings*

| SYMBOL | TERM & MEANING |
|---|---|
|  | **An actor**: Is an entity or system that derives benefit from and is external to the system. Participates in a sequence by sending and/or receiving messages. |
|  | **An object**: Participates in a sequence by sending and/or receiving messages. |
|  | **A lifeline**: Represents the life of an object during a sequence. |
|  | **A focus of control**: Is a long narrow rectangle placed on or above a lifeline. Denotes when an object is sending or receiving messages. |
| | |

| | |
|---|---|
| A Message () → | **A message**: Conveys information from one object to another. |
| X | **Object destruction**: An X is placed at the end of an object’s lifeline to show that it is going out of existence |

**Sequence Diagram for Customer Use Case**

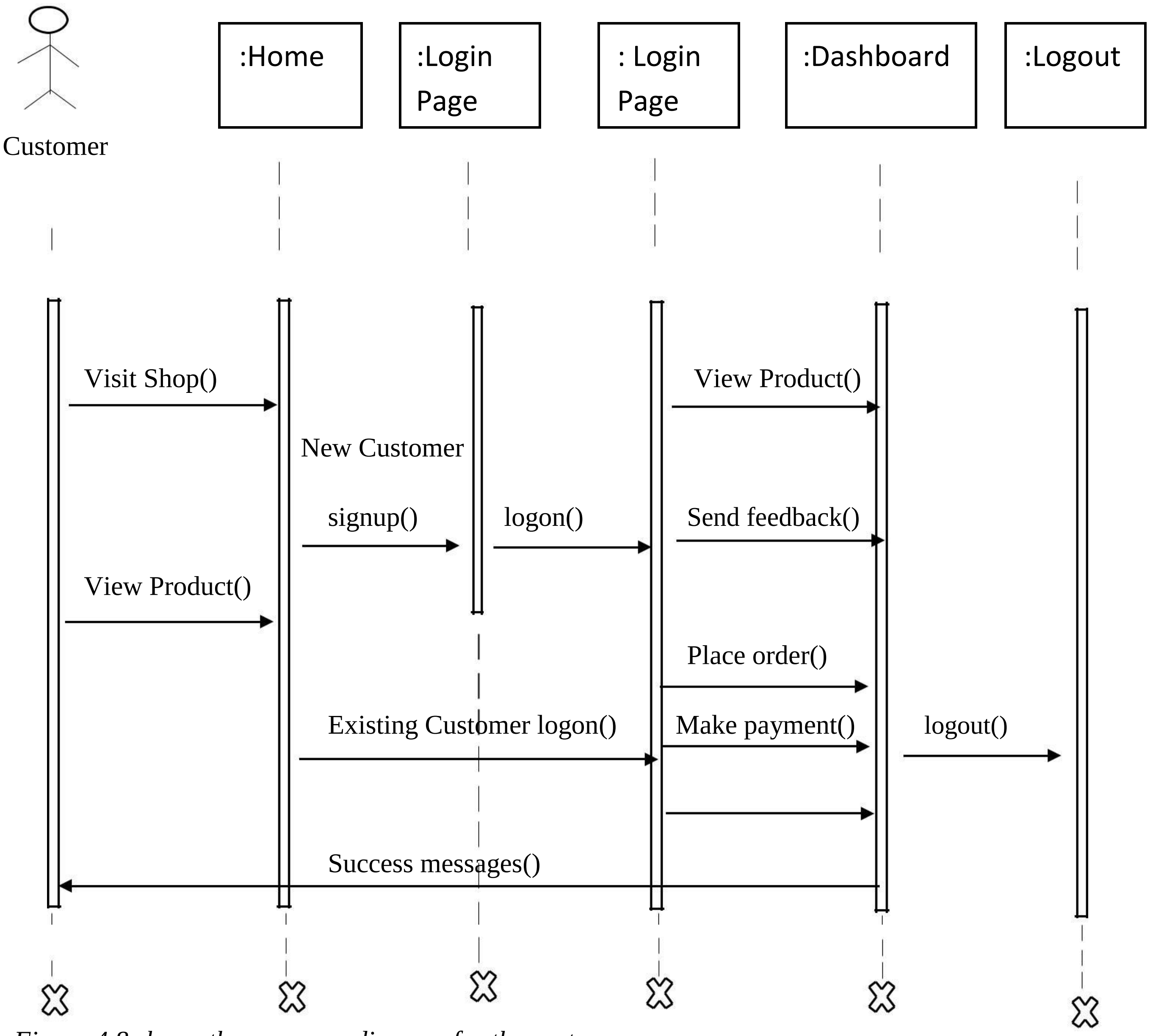


*Figure 4.8 shows the sequence diagram for the customer use case.*

**Sequence Diagram for Manager Use Case**

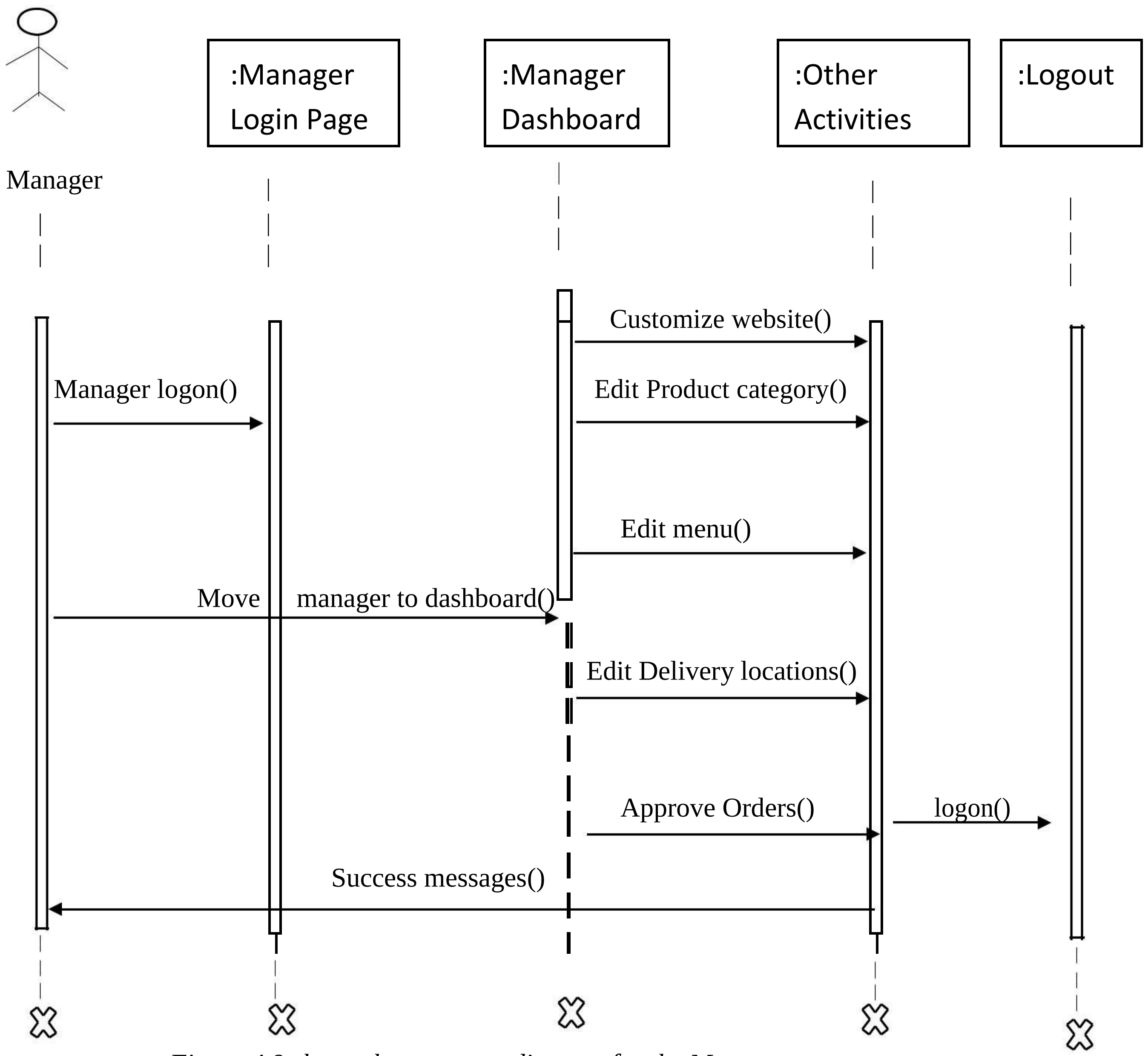


*Figure 4.9 shows the sequence diagram for the Manager use case.*

### 4.4.3 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. It describes the flow of control of the target system, such as the exploring complex business rules and operations, describing the use case also the business process. In the Unified Modeling Language, activity diagrams are intended to model both computational and organizational processes (i.e. workflows).

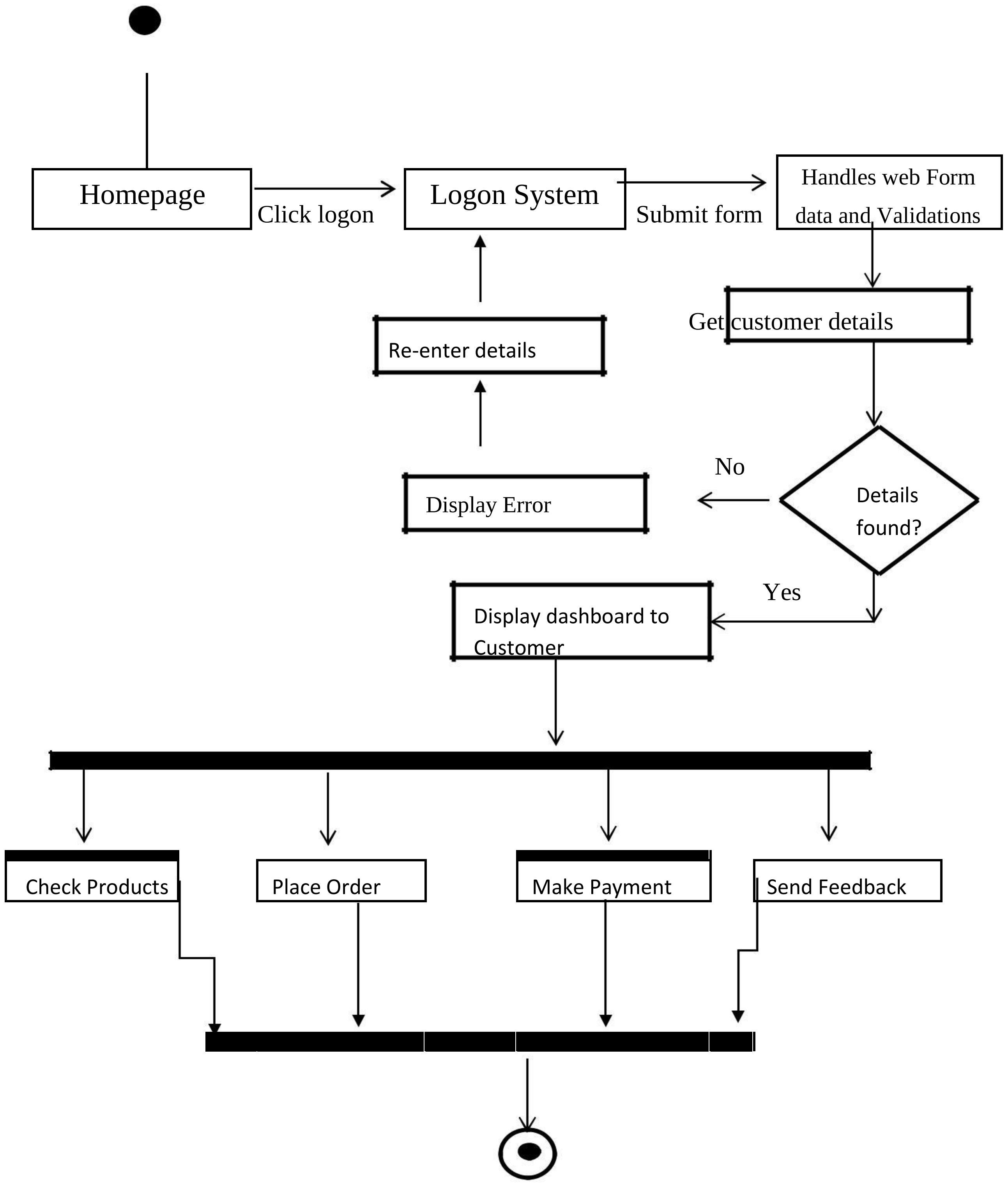


*Figure 4.10 shows the activity diagram for the customer's use case*

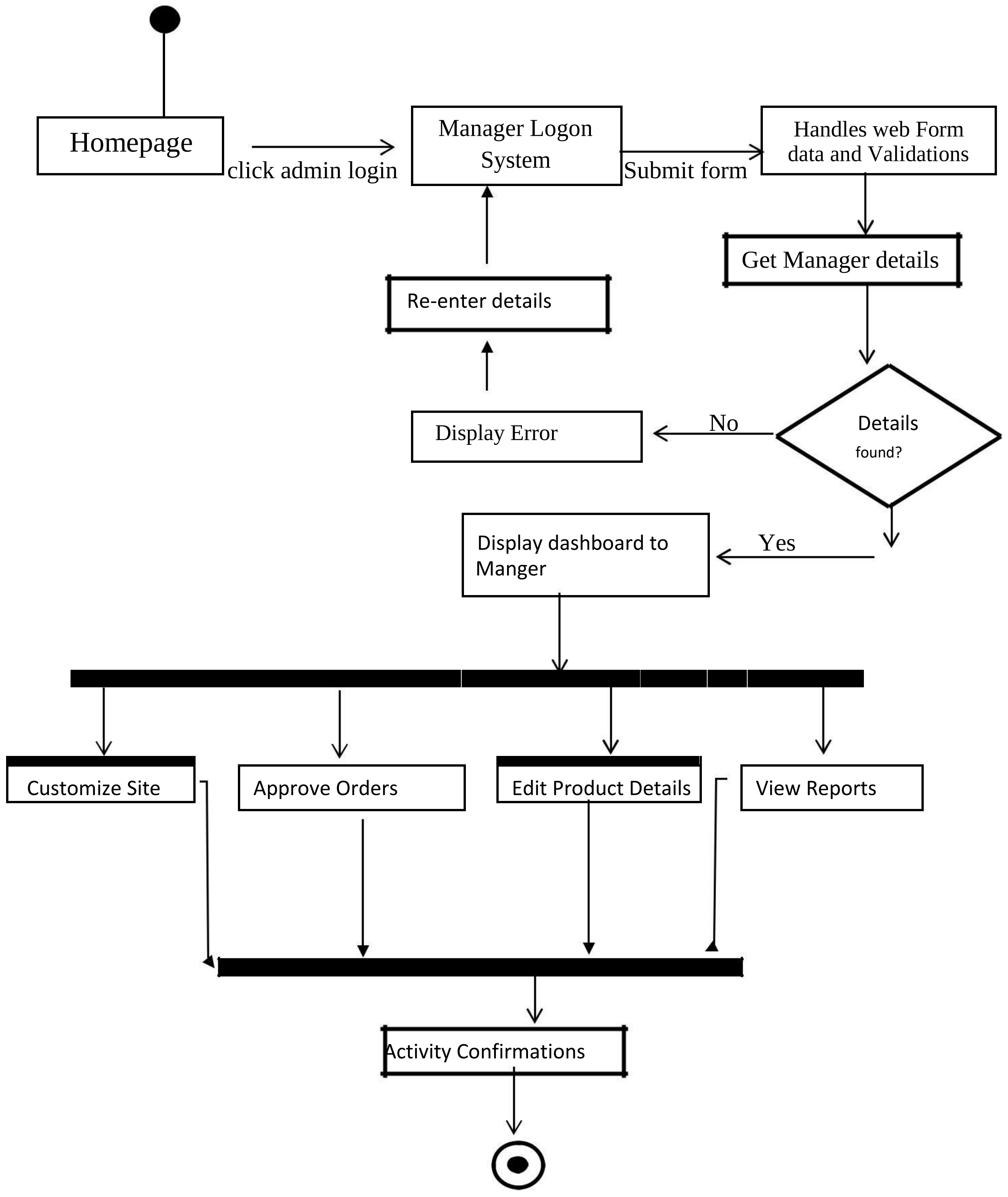


*Figure 4.11 shows the activity diagram for the Manager use case*

### 4.4.4 CLASS DIAGRAM

The class diagram is a central modeling technique that runs through nearly all object-oriented methods. This diagram describes the types of objects in the system and various kinds of static relationships which exist between them.

➢ **Relationships**

There are three principal kinds of relationships which are important:

- **Association** - represent relationships between instances of types (a person works for a company a company has a number of offices.
- **Inheritance** - the most obvious addition to ER diagrams for use in OO. It has an immediate correspondence to inheritance in OO design.
- **Aggregation** - Aggregation, a form of object composition in object-oriented design.

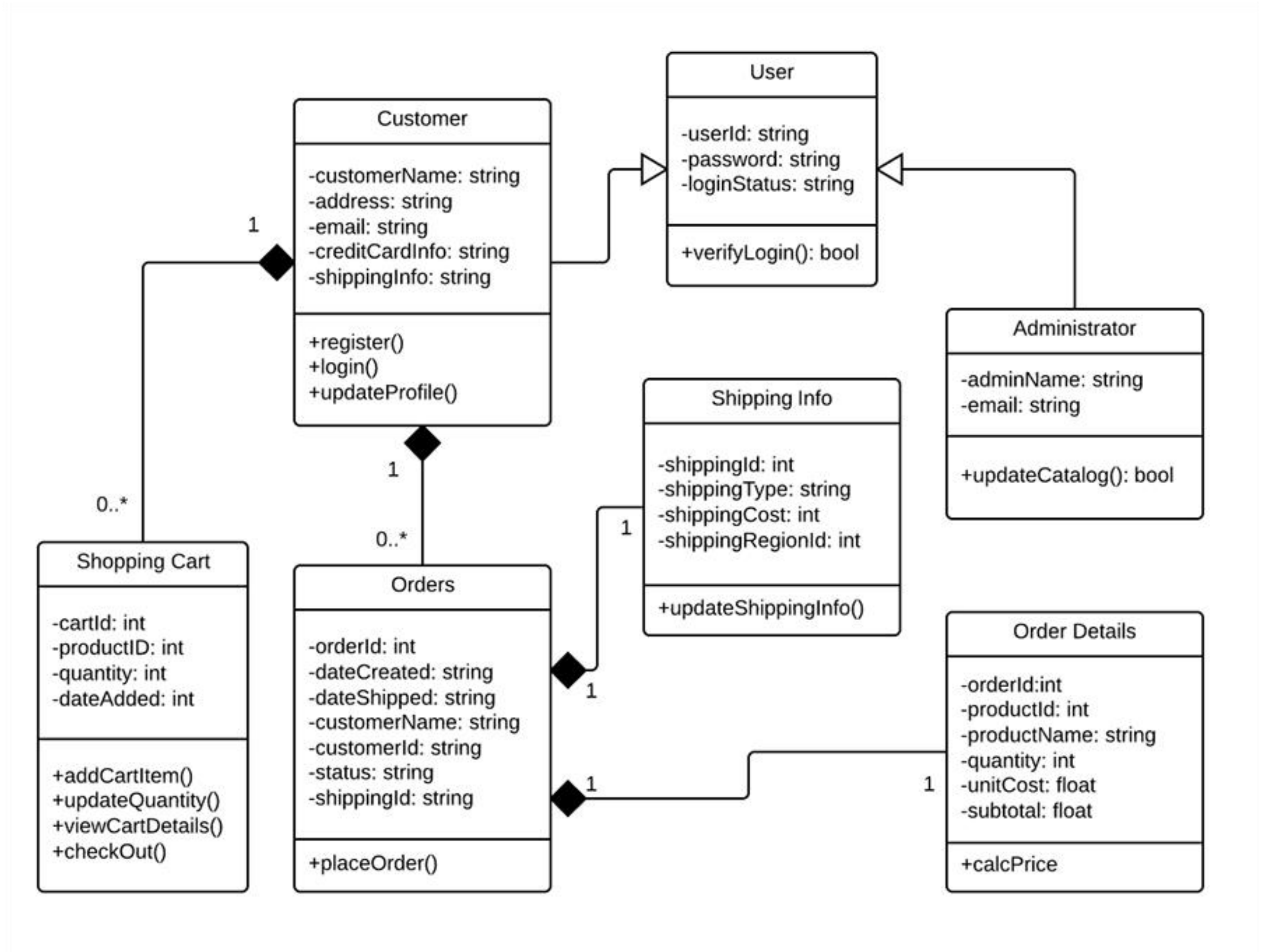


*Figure 4.12 shows the class diagram for the ecommerce system*

## 4.5 DATABASE DESIGN

Database is any collection of data, or information, that is specially organized for rapid search and retrieval by a computer. Databases are structured to facilitate the storage, retrieval, modification, and deletion of data in conjunction with various data-processing operations. A database is stored as a file or a set of files on magnetic disk or tape, optical disk, or some other secondary storage device. The information in these files may be broken down into records, each of which consists of one or more fields. Fields are the basic units of data storage, and each field typically contains information pertaining to one aspect or attribute of the entity described by the database. Records are also organized into tables that include information about relationships between its various fields.

Database Design is a collection of processes that facilitate the designing, development, implementation and maintenance of enterprise data management systems. The main objectives of database designing are to produce logical and physical designs models of the proposed database system.

The logical model concentrates on the data requirements and the data to be stored independent of physical considerations. It does not concern itself with how the data will be stored or where it will be stored physically.

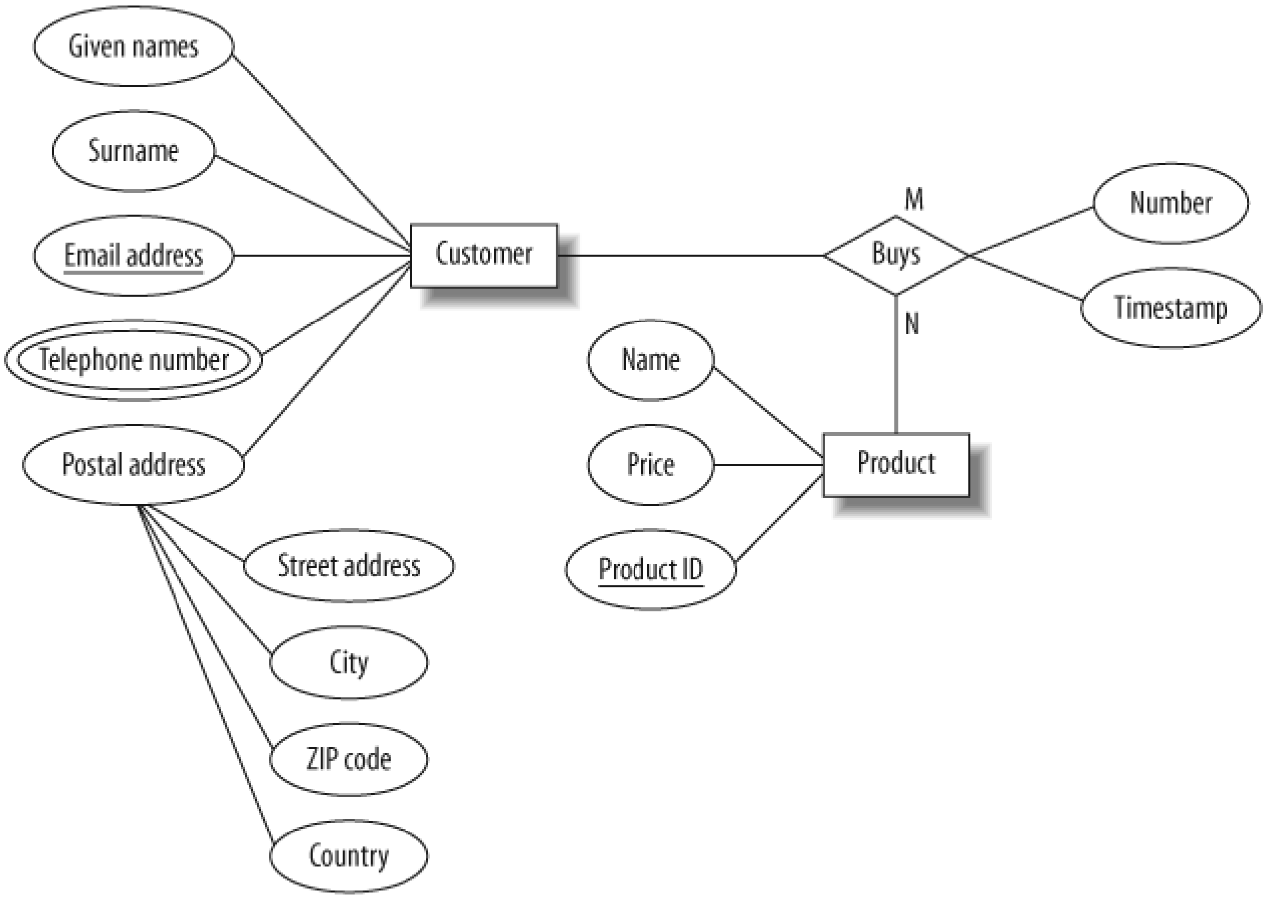
The physical data design model involves translating the logical design of the database onto physical media using hardware resources and software systems such as database management systems (DBMS).

A well-designed database gives access to up-to-date and accurate information. The Entity-Relationship (ER) model, Unified Modeling Language (UML), Relational Model (RM) among others are some of the models used by database designers. The Entity-Relationship (ER) model was used for our database design.

### 4.5.1 ENTITY RELATIONSHIP MODEL

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.

An ER diagram is a means of visualizing how the information a system produces is related.
There are five main components of an ERD:



*4.25 ER diagram for the customer model*

**4.6 NORMALIZATION**

Database Normalization is a technique of organizing the data in the database. Normalization is a systematic approach of decomposing tables to eliminate data redundancy(repetition) and undesirable characteristics like Insertion, Update and Deletion Anomalies. It is a multi-step process that puts data into tabular form, removing duplicated data from the relation tables.

Normalization is used for mainly two purposes: **Eliminating redundant(useless) data** and **ensuring data dependencies make sense i.e. data is logically stored.**

- **Insertion Anomaly:**

  It occurs when certain attributes cannot be inserted or entered into the database without the presence of other attributes.
- **Update Anomaly:**

  An update anomaly exists when one or more instances of duplicated data is updated but not all.
- **Deletion Anomaly:**

  This is when certain attributes are lost because of the deletion of other attributes.

### 4.6.1 NORMALIZATION RULE

Normalization rules are divided into the following normal forms:

1. First Normal Form
2. Second Normal Form
3. Third Normal Form
4. Boyce and Codd Normal Form (BCNF)

- First Normal Form (1NF):

  The rule for First Normal Form states that no two rows of data must contain repeating group of information, therefore each set of columns must have a unique value. To ensure tables in the project are in First Normal Form, it should follow the following 4 rules:

    1. It should only have single(atomic) valued attributes/columns.
    2. Values stored in a column should be of the same domain
    3. All the columns in a table should have unique names.
    4. And the order in which data is stored, does not matter.

- Second Normal Form (2NF):

  A table is said to be in 2NF if it is in 1NF and there is no partial dependency of any column on the primary key. To ensure tables in the project are in the Second Normal Form:

1. It should be in the First Normal form.
2. And, it should not have Partial Dependency.

- ➢ Third normal Form (3NF):

  It applies that a table must be in 2NF and every non-prime attribute of the table must be dependent on the primary key and hence cannot be determined by another non-prime attribute. To ensure tables in the project are in the Third Normal Form:

  1. It is in the Second Normal form.
  2. And, it doesn't have Transitive Dependency.

- ➢ Boyce and Codd Normal Form (BCNF)

  Boyce and Codd Normal Form is a higher version of the Third Normal form. This form deals with certain type of anomaly that is not handled by 3NF. A 3NF table which does not have multiple overlapping candidate keys is said to be in BCNF. To ensure tables in the project are in the BCNF, following conditions must be satisfied:

  1. R must be in 3rd Normal Form
  2. and, for each functional dependency ($X \rightarrow Y$), X should be a super Key.

The database for this project is normalized in the **Third Normal Form (3NF)** to ensure data redundant data is eliminated that is, all data is stored in only one place and also to ensure data dependencies are logical which means all related data items are stored together.

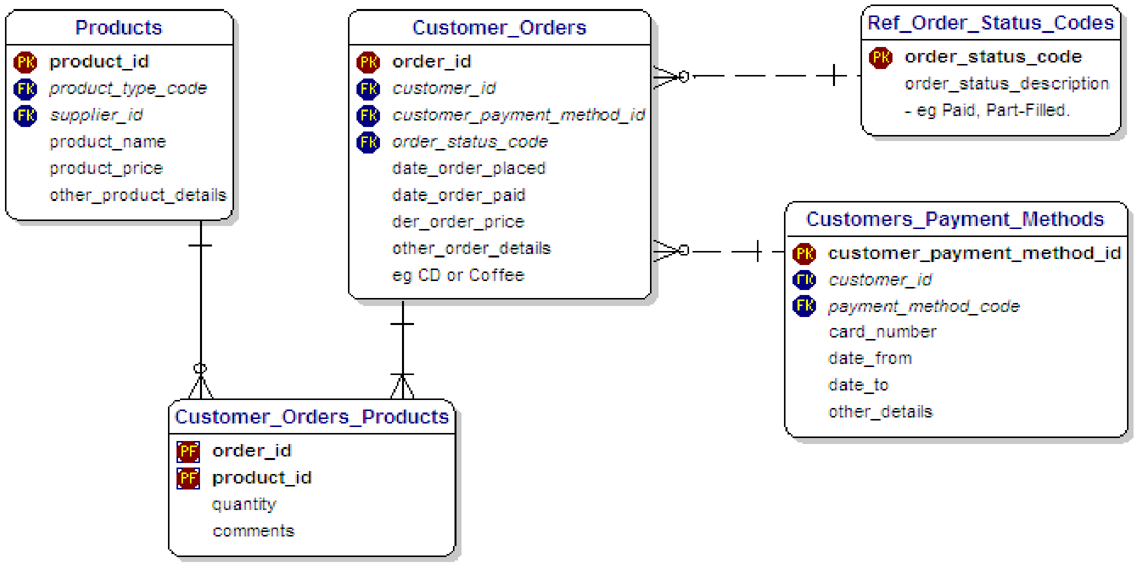


*Figure 4.26 Normalized database for all tables*

## 4.7 USER INTERFACE DESIGN

User interface design or UI design generally refers to the visual layout of the elements that a user might interact with in a website, or technological product. This could be the control buttons of a radio, or the visual layout of a webpage. User interface designs must not only be attractive to potential users but must also be functional and created with users in mind. The goal of user interface design is to make the user's interaction as simple and efficient as possible, in terms of accomplishing user goals (user-centered design). Good user interface design facilitates finishing the task at hand without drawing unnecessary attention to itself.

During the system development phase, several UI design features like input controls which includes textboxes, radio buttons, checkboxes, date fields, dropdown lists etc. For example, when a customer wants to place an order, recommendations will be made on the products they select to help the customer order with ease. The use of navigation components such as pagination, search field, slider, icons, tags etc. will be used to make user interactions with the website easier. Information components like notifications, messages boxes, contact forms etc. will also be used to help the user get quick feedbacks from the system.

The following best User interface design practices used in Software and website Development will be used in the user interface (UI) design for this project:

- **Clarity:**

  The interface will be designed with the customer who will make the orders in mind. The information content is conveyed quickly and accurately. The interface will be simple for the customer to navigate easily.

- **Consistency:**

  A unique design, conformity with user's expectation in mind. The UI design elements that would be used in the system implementation will be consistent throughout the site. This will help the user to be familiar with certain UI elements so that they will not be using different elements for similar operations.

- **Conciseness**:

Users will not be overloaded with extraneous information**.** Consideration would be made for the spatial relationships between items on the page and structure the page based on importance. Careful placement of items would help draw attention to the most important pieces of information and can aid scanning and readability.

- **Strategically use color and texture:**
  Direct attention would be drawn towards or redirected away from items using color, light, contrast, etc. The intention is to blend colors that will make the website nice and simple.
- **Detectability:**
  Carefully consideration will be made on how to use typeface to help draw the user's attention towards information required. Different sizes, fonts, and arrangement of the text to help increase scan ability, legibility and readability of the user.
- **Comprehensibility:**
  It is imperative to inform users of location, actions, changes in state, or errors. The use of various UI elements to communicate status and, if necessary, next steps can reduce frustration for users and that is exactly what will be done during the implementation of the system. This makes the meaning clearly understandable, unambiguous, interpretable, and recognizable.
- **Thinking about the defaults:**
  By carefully thinking about and anticipating the goals people bring to our site, we can create defaults that reduce the burden on the user. This becomes particularly important when it comes to form design where you might have an opportunity to have some fields pre-chosen or filled out.

## 4.8 SUMMARY

This chapter focused on the analysis and design of the system which were the general system and application architecture, requirements specification, functional and non-functional requirements, use case modelling, the database design, Entity Relationship diagram, Normalization, and User interface design. Outline for the requirement specification of the system which included both functional and non-functional requirements and the hardware requirements were explained. Further explanation on the various Unified Modelling languages used in modelling the system were discussed. This included use case diagrams, activity diagrams, class diagrams and sequence diagrams. These diagrams provided a pictorial view of the relationships between the various entities or users of our system. Also, the design of the system's database including the Entity-Relationship (ER) diagram and the importance of database normalization were elaborated including the database being normalized in the third normal form (3NF). Finally, the user interface was looked at including how the user interface will look and why it was necessary for the interface to be user friendly.

# CHAPTER FIVE

# SYSTEM IMPLEMENTATION

## 5.0 INTRODUCTION

System implementation and deployment is the part in which the system will be implemented and deployed into real life to be used. The system can only be implemented and deployed when the system design and analysis is completed.

This chapter details out the implementation of the system. This include mapping logical design onto physical platform, the construction which consist of screen shots of forms, databases and reports among others. The chapter will also outline the various testing that were performed on the system, that is unit and system testing and the results obtained.

## 5.1 MAPPING LOGICAL DESIGN ONTO PHYSICAL PLATFORM

Logical design involves arranging data into a series of logical relationships called entities and attributes. Here, we defined the various fields for the database schema. Customer first names, e-mail, shipping address and other details were converted into first_name, email and shipping_address respectively. The Django framework and the other development tools were used to develop the various model of the project and linked them together. The next sub-topic gives a graphical view of the various outputs after transforming the logical design into executable codes.

### 5.1.1 HARDWARE REQUIREMENTS

The minimum hardware requirement in develop this system are listed as below:

| Hardware Description | Minimum Requirements |
|---|---|
| Processor | Intel Pentium D 3.4GHz / AMD Athlon II X2 250 u (Minimum)<br>Intel Core 2 Duo E4400 2.0GHz / AMD Athlon 64 X2 Dual Core 4600+ (Recommended) |
| Memory | 1 GB RAM Recommended, 256 MB RAM (Minimum) |
| Hard disk space | Up to 3 GB Recommended |
| Display | 65536 colors, set to at least 1024 X 768 Resolution |

Table 5.0 Table of Hardware Requirements

### 5.1.2 SOFTWARE REQUIREMENTS

The minimum software requirement in develop this system are listed as below:

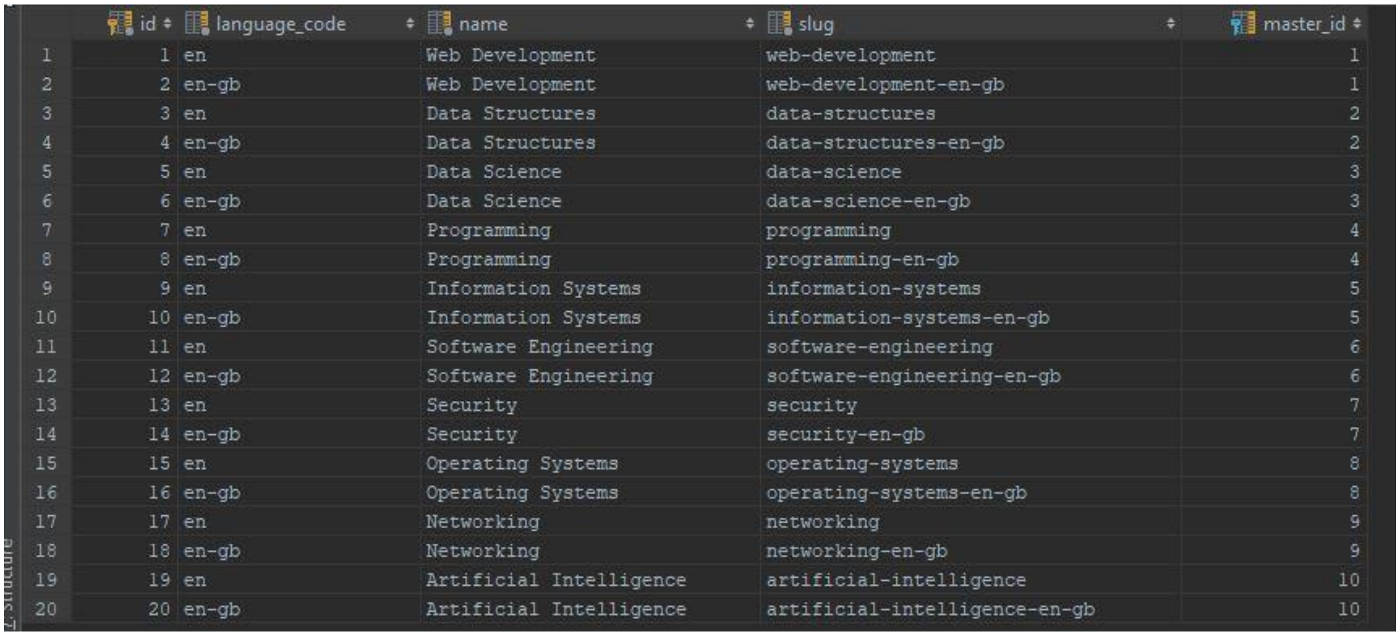
| Software Description | Minimum Requirements |
|---|---|
| Operating System (OS) | All 32-bits Microsoft Windows (95/98/2000/XP/7/8) |
| Browser | Mozilla Firefox (15.0 & above), Internet Explorer (8.0 & above), Google Chrome (20.0 & above). |

Table 5.1 Table of Software Requirements

## 5.2 CONSTRUCTION

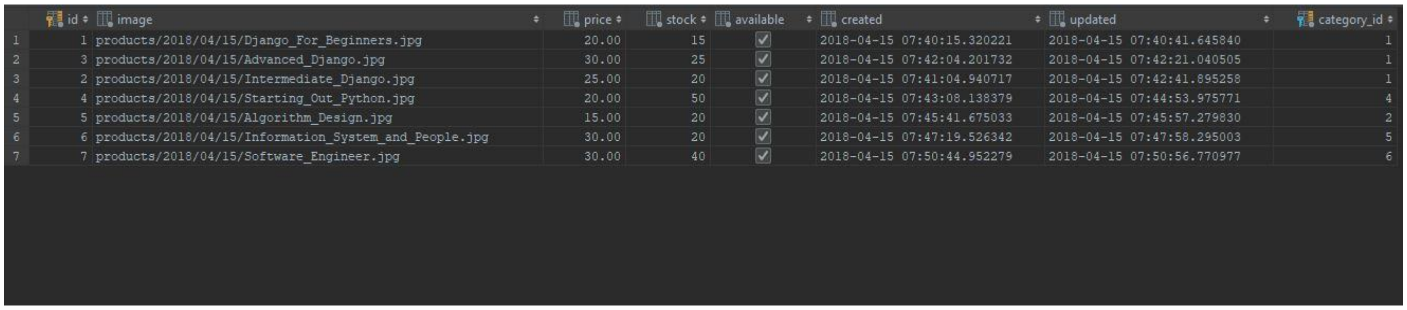
This section focuses on the pictorial view of the user interface including the various forms, databases and reports. Below are various screen shots showing how the system was constructed.

The screenshots of the various tables constructed are shown below:



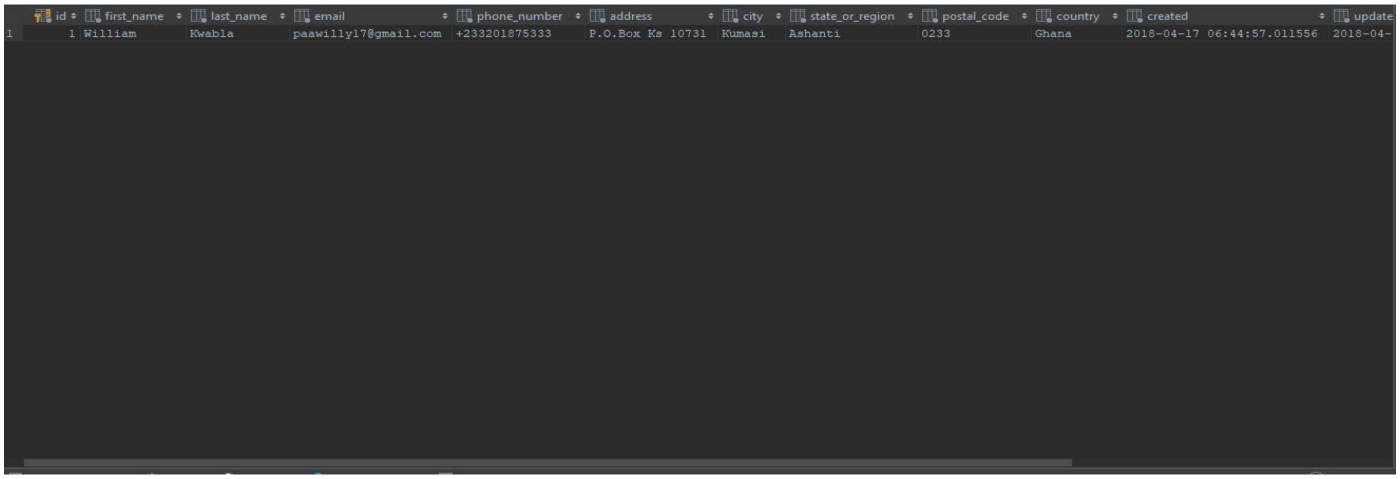
| | id | language_code | name | slug | master_id |
|---|---|---|---|---|---|
| 1 | 1 | en | Web Development | web-development | 1 |
| 2 | 2 | en-gb | Web Development | web-development-en-gb | 1 |
| 3 | 3 | en | Data Structures | data-structures | 2 |
| 4 | 4 | en-gb | Data Structures | data-structures-en-gb | 2 |
| 5 | 5 | en | Data Science | data-science | 3 |
| 6 | 6 | en-gb | Data Science | data-science-en-gb | 3 |
| 7 | 7 | en | Programming | programming | 4 |
| 8 | 8 | en-gb | Programming | programming-en-gb | 4 |
| 9 | 9 | en | Information Systems | information-systems | 5 |
| 10 | 10 | en-gb | Information Systems | information-systems-en-gb | 5 |
| 11 | 11 | en | Software Engineering | software-engineering | 6 |
| 12 | 12 | en-gb | Software Engineering | software-engineering-en-gb | 6 |
| 13 | 13 | en | Security | security | 7 |
| 14 | 14 | en-gb | Security | security-en-gb | 7 |
| 15 | 15 | en | Operating Systems | operating-systems | 8 |
| 16 | 16 | en-gb | Operating Systems | operating-systems-en-gb | 8 |
| 17 | 17 | en | Networking | networking | 9 |
| 18 | 18 | en-gb | Networking | networking-en-gb | 9 |
| 19 | 19 | en | Artificial Intelligence | artificial-intelligence | 10 |
| 20 | 20 | en-gb | Artificial Intelligence | artificial-intelligence-en-gb | 10 |

*Fig 5.0 Database Schema for categories table*



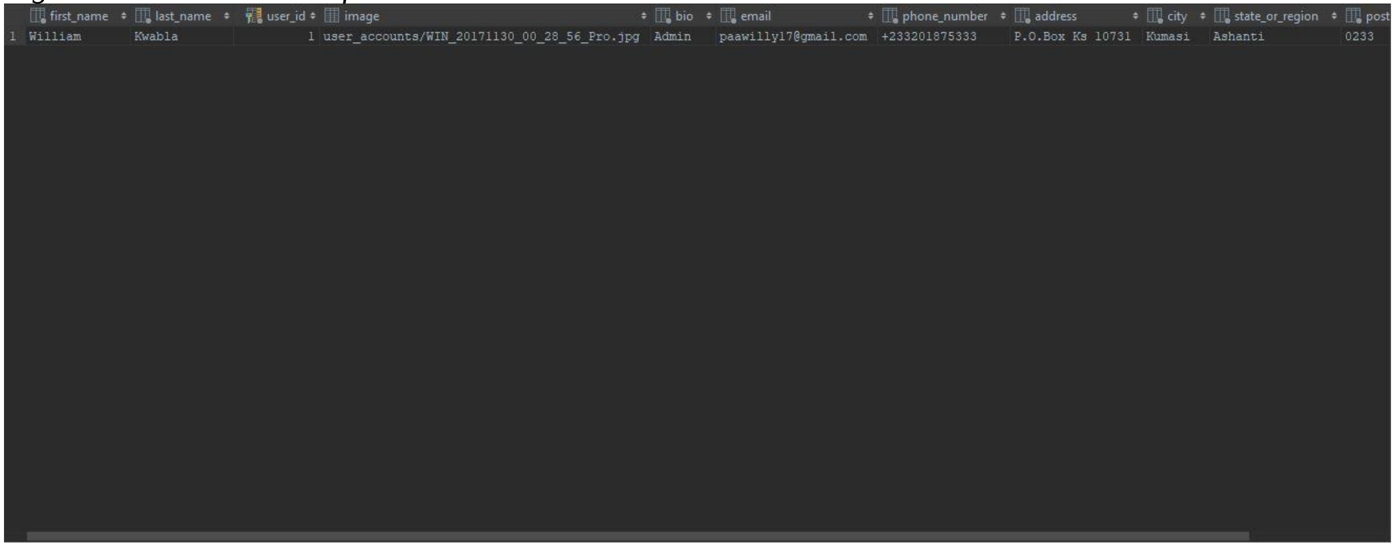
| | id | image | price | stock | available | created | updated | category_id |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | products/2018/04/15/Django_For_Beginners.jpg | 20.00 | 15 | ✓ | 2018-04-15 07:40:15.320221 | 2018-04-15 07:40:41.645840 | 1 |
| 2 | 3 | products/2018/04/15/Advanced_Django.jpg | 30.00 | 25 | ✓ | 2018-04-15 07:42:04.201732 | 2018-04-15 07:42:21.040505 | 1 |
| 3 | 2 | products/2018/04/15/Intermediate_Django.jpg | 25.00 | 20 | ✓ | 2018-04-15 07:41:04.940717 | 2018-04-15 07:42:41.895258 | 1 |
| 4 | 4 | products/2018/04/15/Starting_Out_Python.jpg | 20.00 | 50 | ✓ | 2018-04-15 07:43:08.138379 | 2018-04-15 07:44:53.975771 | 4 |
| 5 | 5 | products/2018/04/15/Algorithm_Design.jpg | 15.00 | 20 | ✓ | 2018-04-15 07:45:41.675033 | 2018-04-15 07:45:57.279830 | 2 |
| 6 | 6 | products/2018/04/15/Information_System_and_People.jpg | 30.00 | 20 | ✓ | 2018-04-15 07:47:19.526342 | 2018-04-15 07:47:58.295003 | 5 |
| 7 | 7 | products/2018/04/15/Software_Engineer.jpg | 30.00 | 40 | ✓ | 2018-04-15 07:50:44.952279 | 2018-04-15 07:50:56.770977 | 6 |

*Fig 5.1 Database Schema for product table*



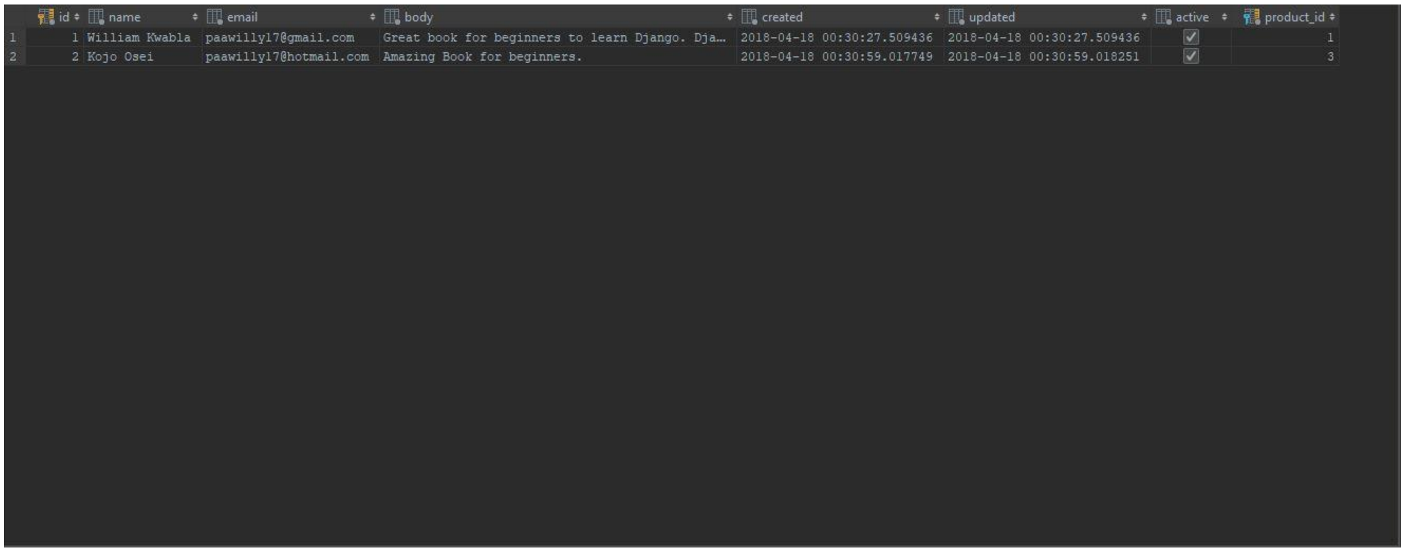
| | id | first_name | last_name | email | phone_number | address | city | state_or_region | postal_code | country | created | update |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | William | Kwabla | paawilly17@gmail.com | +233201875333 | P.O.Box Ks 10731 | Kumasi | Ashanti | 0233 | Ghana | 2018-04-17 06:44:57.011556 | 2018-04- |

*Fig 5.2 Database Schema for orders table*



| | first_name | last_name | user_id | image | bio | email | phone_number | address | city | state_or_region | post |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | William | Kwabla | 1 | user_accounts/WIN_20171130_00_28_56_Pro.jpg | Admin | paawilly17@gmail.com | +233201875333 | P.O.Box Ks 10731 | Kumasi | Ashanti | 0233 |

*Fig 5.3 Database Schema for Customer Profiles table*

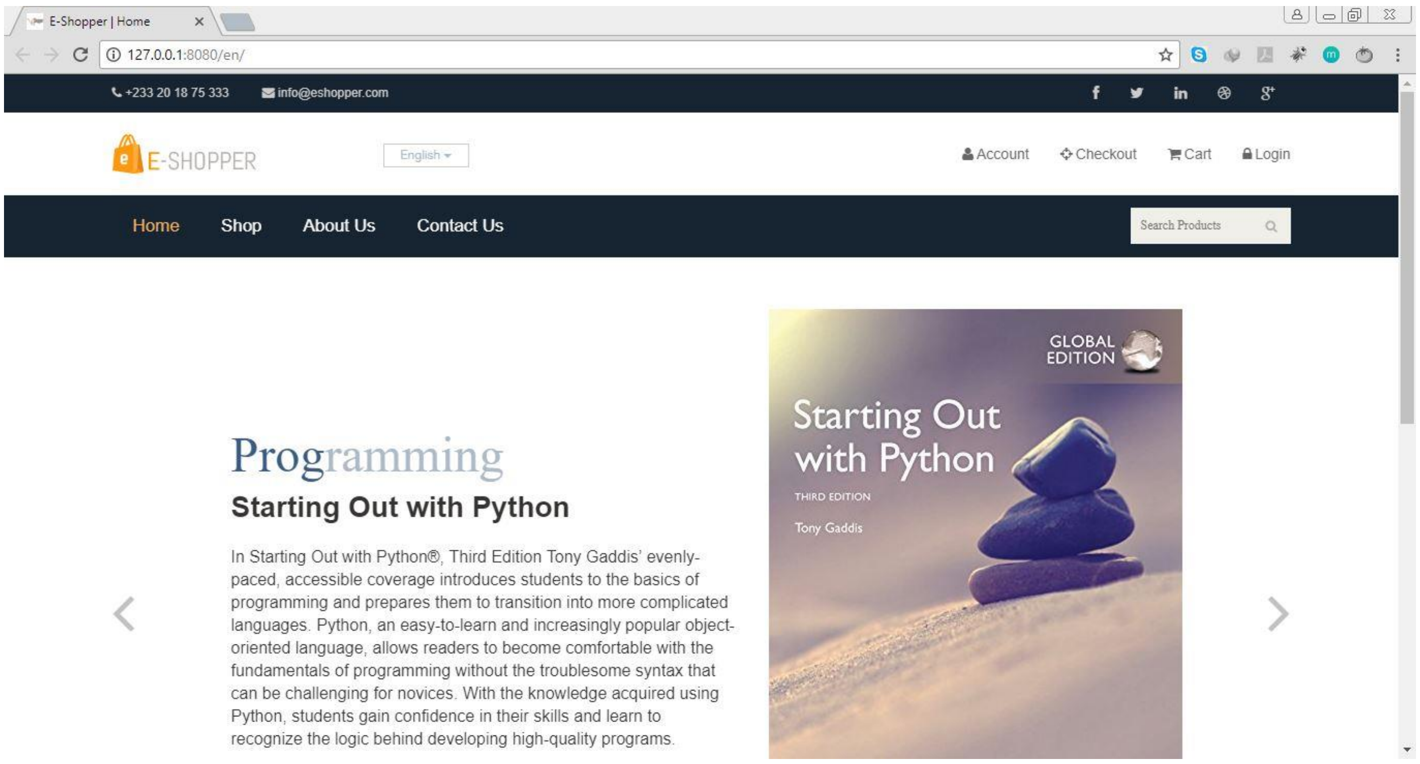


| | id | name | email | body | created | updated | active | product_id |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | William Kwabla | paawilly17@gmail.com | Great book for beginners to learn Django. Dja… | 2018-04-18 00:30:27.509436 | 2018-04-18 00:30:27.509436 | ✓ | 1 |
| 2 | 2 | Kojo Osei | paawilly17@hotmail.com | Amazing Book for beginners. | 2018-04-18 00:30:59.017749 | 2018-04-18 00:30:59.018251 | ✓ | 3 |

*Fig 5.4 Database Schema for Product Review table*

## 5.2.1 USER INTERFACES

As we mentioned in the previous chapter about how the user interface would be designed, below are the screenshots of the various user interfaces of the system.



Programming

Starting Out with Python

In Starting Out with Python®, Third Edition Tony Gaddis' evenly-paced, accessible coverage introduces students to the basics of programming and prepares them to transition into more complicated languages. Python, an easy-to-learn and increasingly popular object-oriented language, allows readers to become comfortable with the fundamentals of programming without the troublesome syntax that can be challenging for novices. With the knowledge acquired using Python, students gain confidence in their skills and learn to recognize the logic behind developing high-quality programs.

*Fig 5.5 Homepage of the application with a slideshow of some of the books sold.*

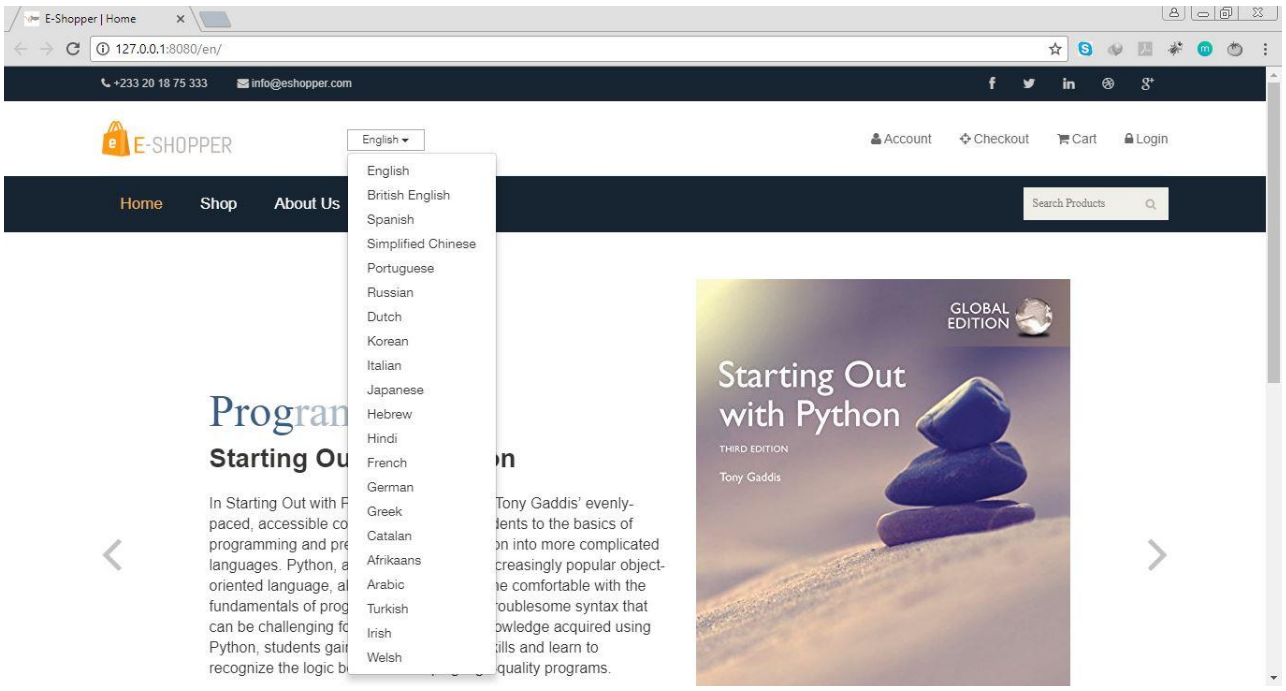
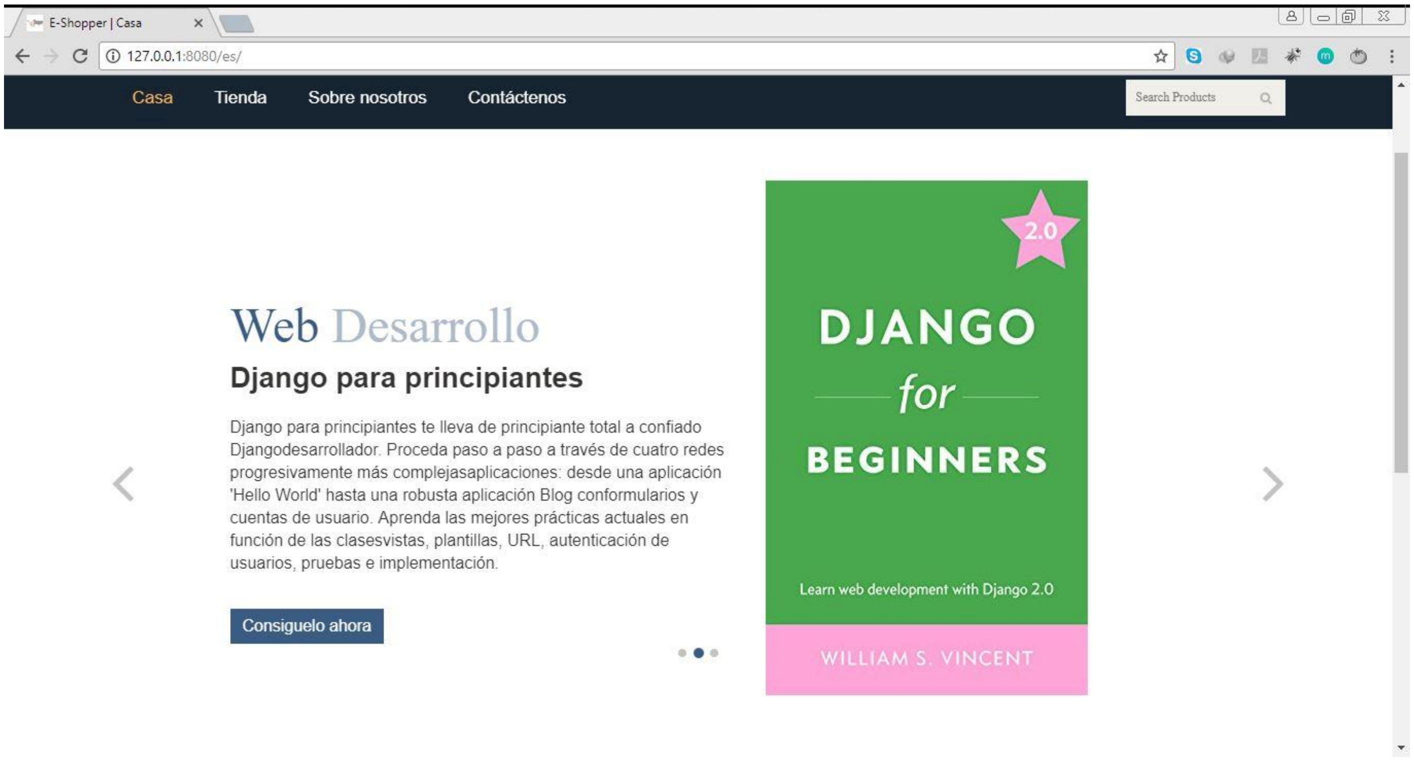


*Fig 5.6 Homepage with a dropdown of 21 languages available for customers to choose before they start shopping.*



Web Desarrollo

Django para principiantes

Django para principiantes te lleva de principiante total a confiado Djangodesarrollador. Proceda paso a paso a través de cuatro redes progresivamente más complejasaplicaciones: desde una aplicación 'Hello World' hasta una robusta aplicación Blog conformularios y cuentas de usuario. Aprenda las mejores prácticas actuales en función de las clasesvistas, plantillas, URL, autenticación de usuarios, pruebas e implementación.

*Fig 5.7 Homepage and URL after Spanish is selected.*

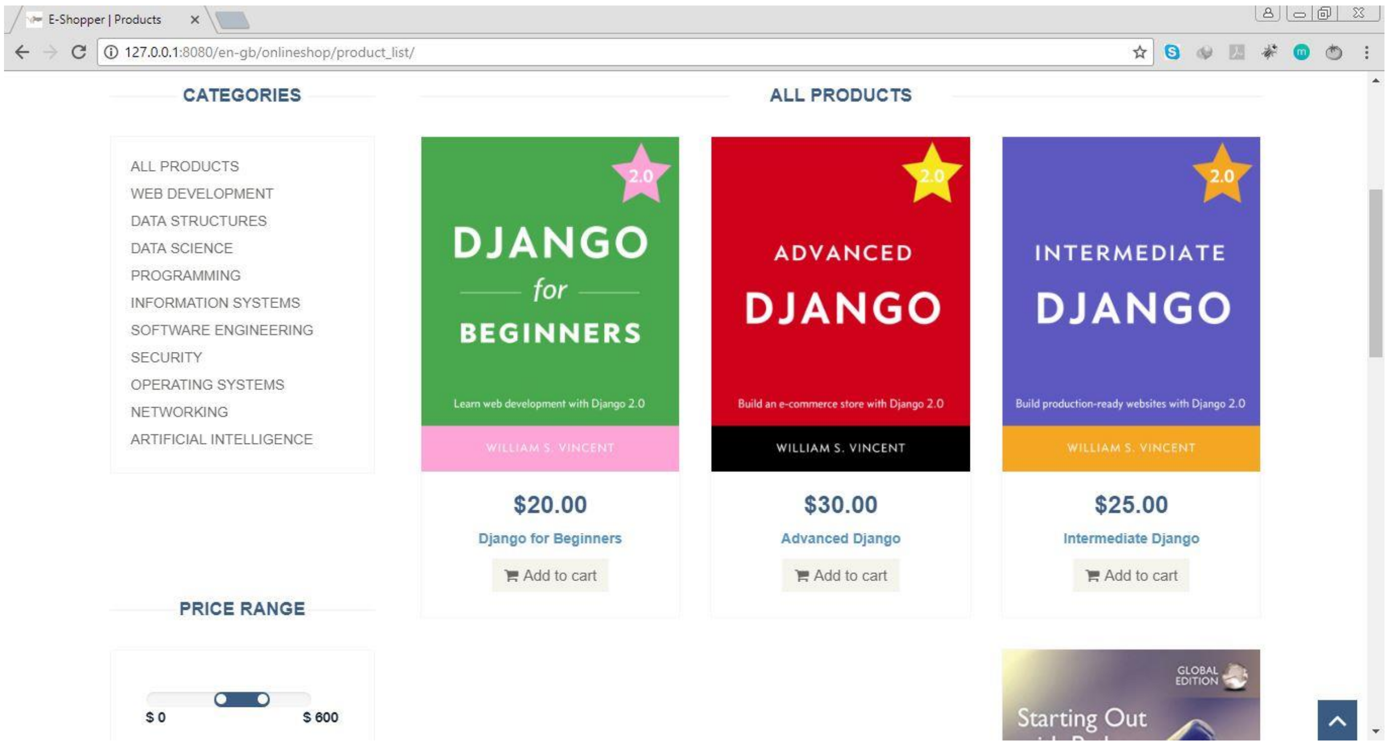
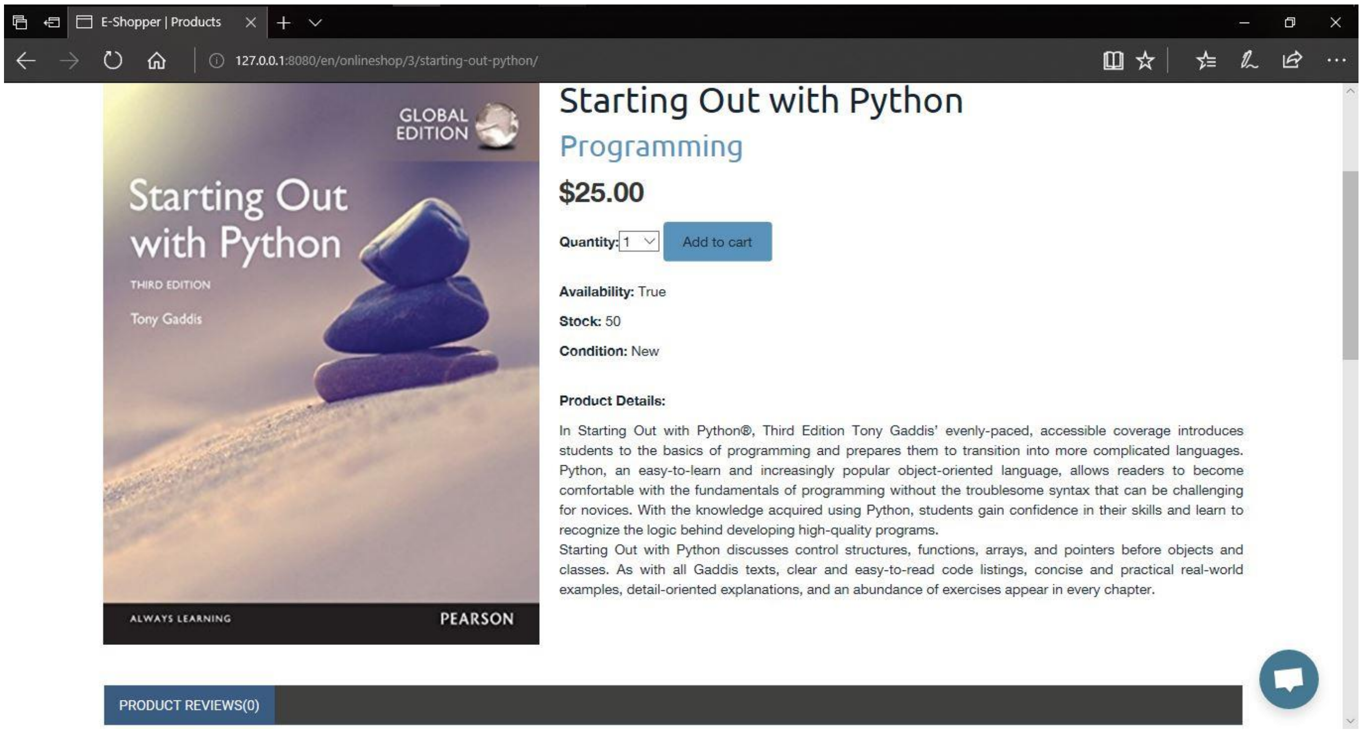


*Fig 5.8 Shop or Product list page with Categories*



Starting Out with Python

Programming

$25.00

**Quantity:** 1 Add to cart

**Availability:** True

**Stock:** 50

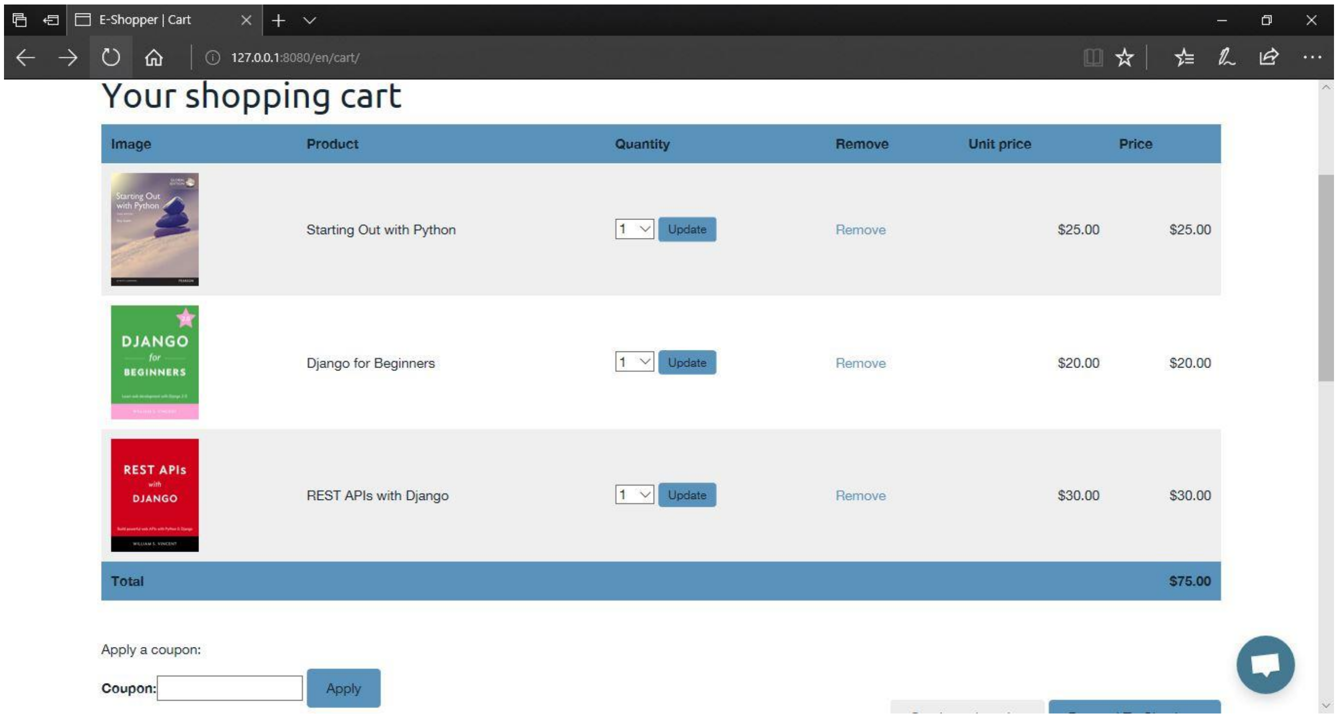
**Condition:** New

**Product Details:**

In Starting Out with Python®, Third Edition Tony Gaddis' evenly-paced, accessible coverage introduces students to the basics of programming and prepares them to transition into more complicated languages. Python, an easy-to-learn and increasingly popular object-oriented language, allows readers to become comfortable with the fundamentals of programming without the troublesome syntax that can be challenging for novices. With the knowledge acquired using Python, students gain confidence in their skills and learn to recognize the logic behind developing high-quality programs.
Starting Out with Python discusses control structures, functions, arrays, and pointers before objects and classes. As with all Gaddis texts, clear and easy-to-read code listings, concise and practical real-world examples, detail-oriented explanations, and an abundance of exercises appear in every chapter.

PRODUCT REVIEWS(0)

*Fig 5.9 Product Detail Page*



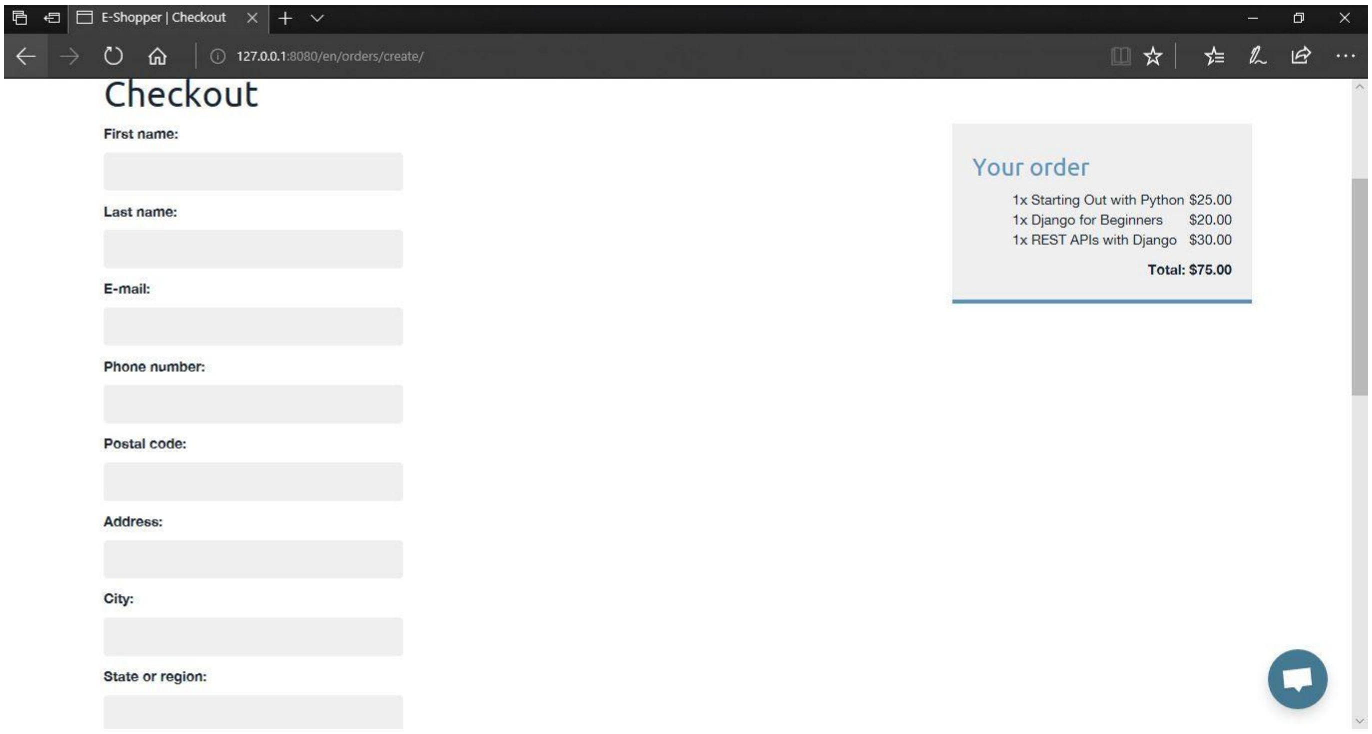
Your shopping cart

| Image | Product | Quantity | Remove | Unit price | Price |
|---|---|---|---|---|---|
| | Starting Out with Python | 1 Update | Remove | $25.00 | $25.00 |
| | Django for Beginners | 1 Update | Remove | $20.00 | $20.00 |
| | REST APIs with Django | 1 Update | Remove | $30.00 | $30.00 |
| Total | | | | | $75.00 |

Apply a coupon:

**Coupon:** Apply

*Fig 5.10 Cart Page*


E-Shopper | Checkout

127.0.0.1:8080/en/orders/create/

Checkout

First name:

Last name:

E-mail:

Phone number:

Postal code:

Address:

City:

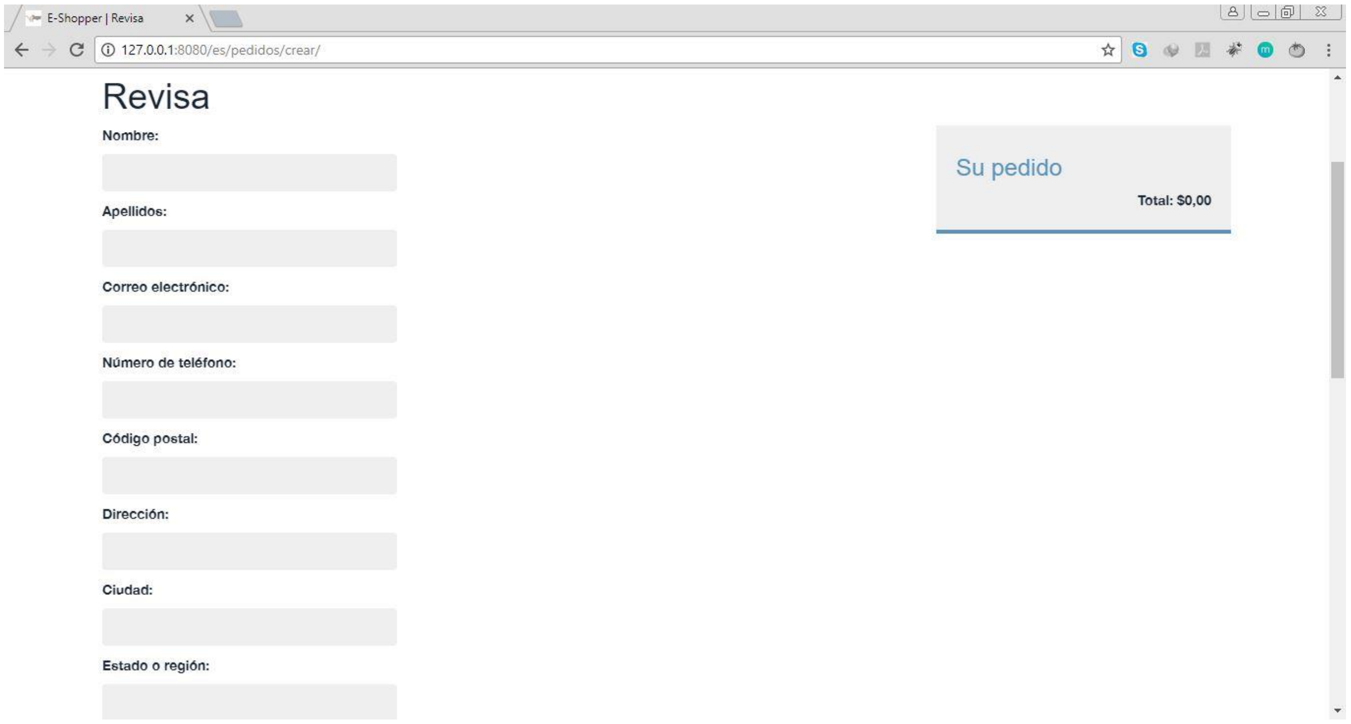
State or region:

Your order

1x Starting Out with Python $25.00
1x Django for Beginners $20.00
1x REST APIs with Django $30.00

Total: $75.00

*Fig 5.11 Checkout Page with cart.*


E-Shopper | Revisa

127.0.0.1:8080/es/pedidos/crear/

Revisa

Nombre:

Apellidos:

Correo electrónico:

Número de teléfono:

Código postal:

Dirección:

Ciudad:

Estado o región:

Su pedido

Total: $0,00

*Fig 5.12 Checkout Page with empty cart in Spanish.*

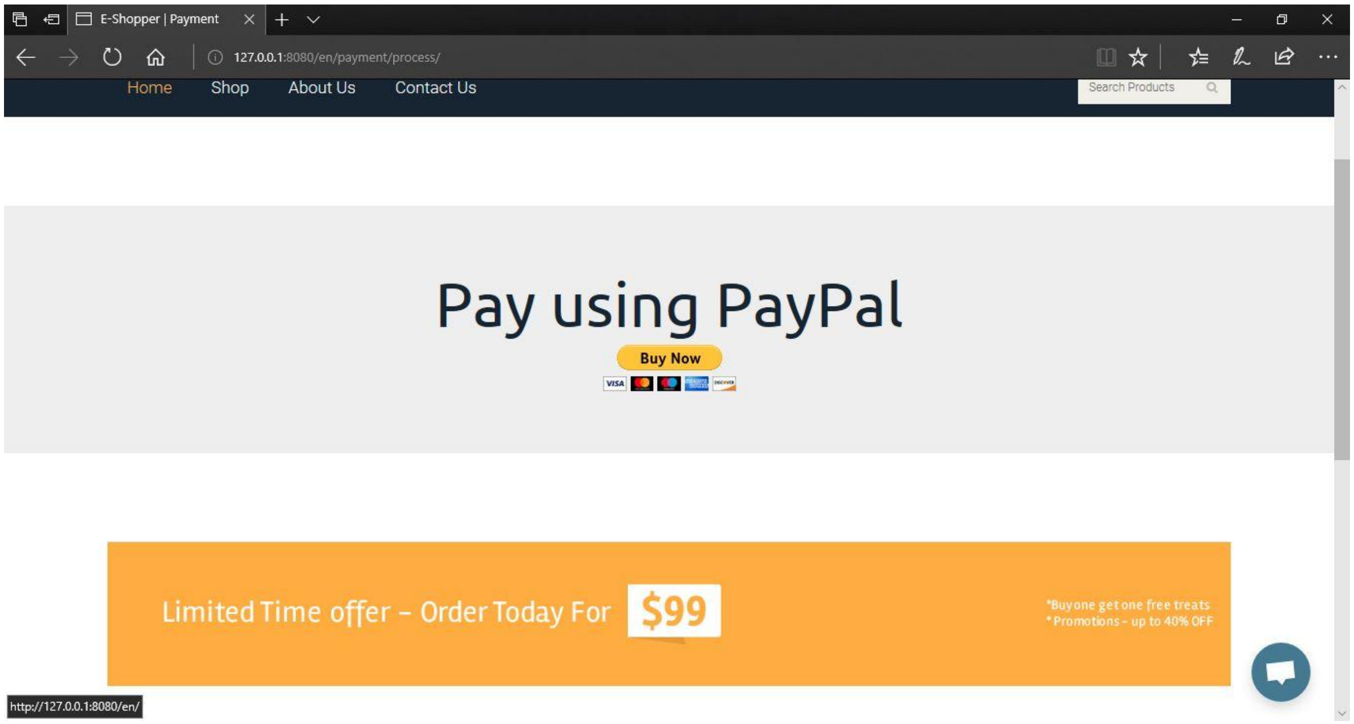

E-Shopper | Payment
127.0.0.1:8080/en/payment/process/
Home
Shop
About Us
Contact Us
Search Products
Pay using PayPal
Buy Now
Limited Time offer – Order Today For $99
*Buy one get one free treats
*Promotions – up to 40% OFF
http://127.0.0.1:8080/en/

*Fig 5.13 Proceed to Payment Page*

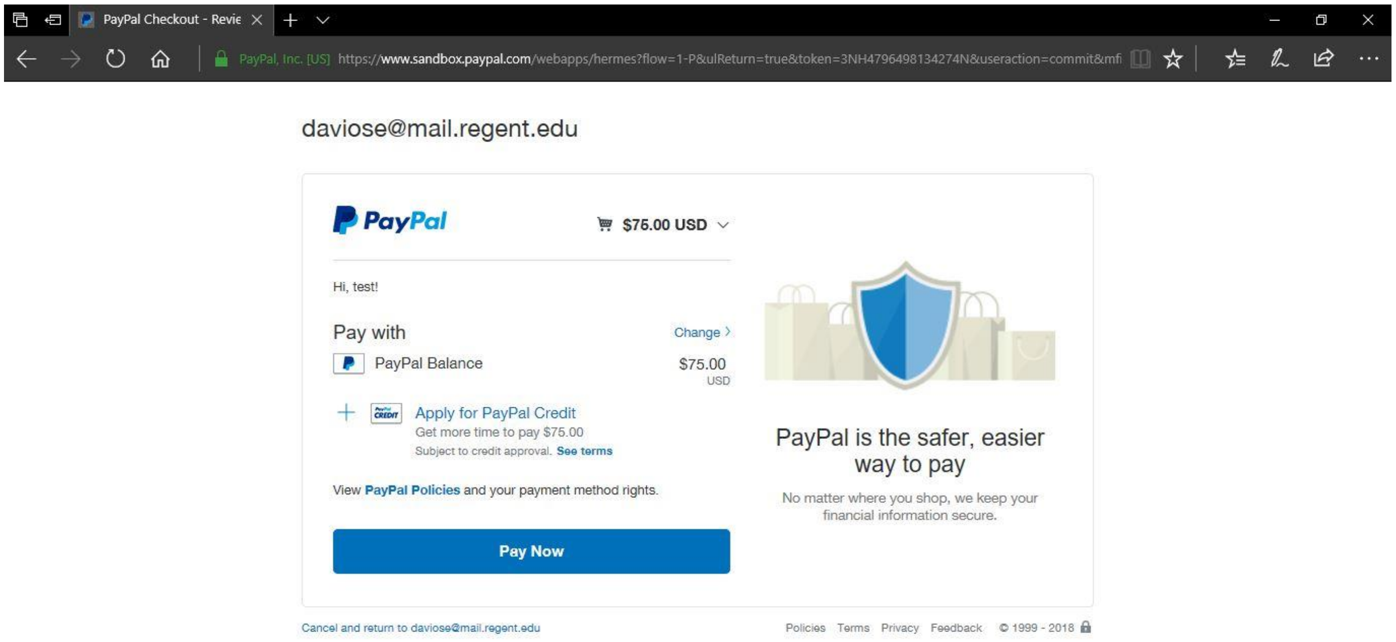
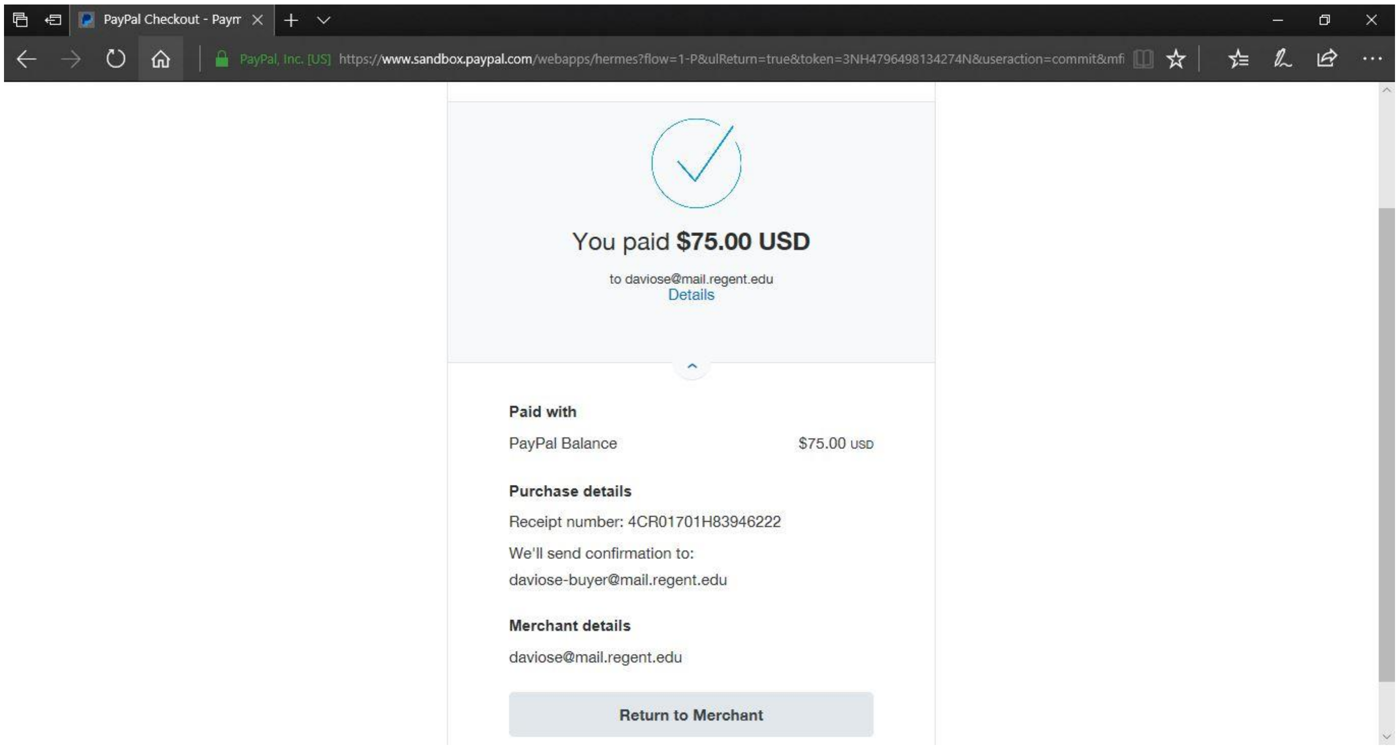

PayPal Checkout - Revie
PayPal, Inc. [US] https://www.sandbox.paypal.com/webapps/hermes?flow=1-P&ulReturn=true&token=3NH4796498134274N&useraction=commit&mfi
daviose@mail.regent.edu
PayPal
$75.00 USD
Hi, test!
Pay with
Change
PayPal Balance
$75.00 USD
Apply for PayPal Credit
Get more time to pay $75.00
Subject to credit approval. See terms
View PayPal Policies and your payment method rights.
Pay Now
PayPal is the safer, easier way to pay
No matter where you shop, we keep your financial information secure.
Cancel and return to daviose@mail.regent.edu
Policies Terms Privacy Feedback © 1999 - 2018

*Fig 5.14 PayPal Payment page*


PayPal Checkout - Paym
PayPal, Inc. [US] https://www.sandbox.paypal.com/webapps/hermes?flow=1-P&ulReturn=true&token=3NH4796498134274N&useraction=commit&mfi

You paid **$75.00 USD**

to daviose@mail.regent.edu
Details

**Paid with**

PayPal Balance $75.00 USD

**Purchase details**

Receipt number: 4CR01701H83946222

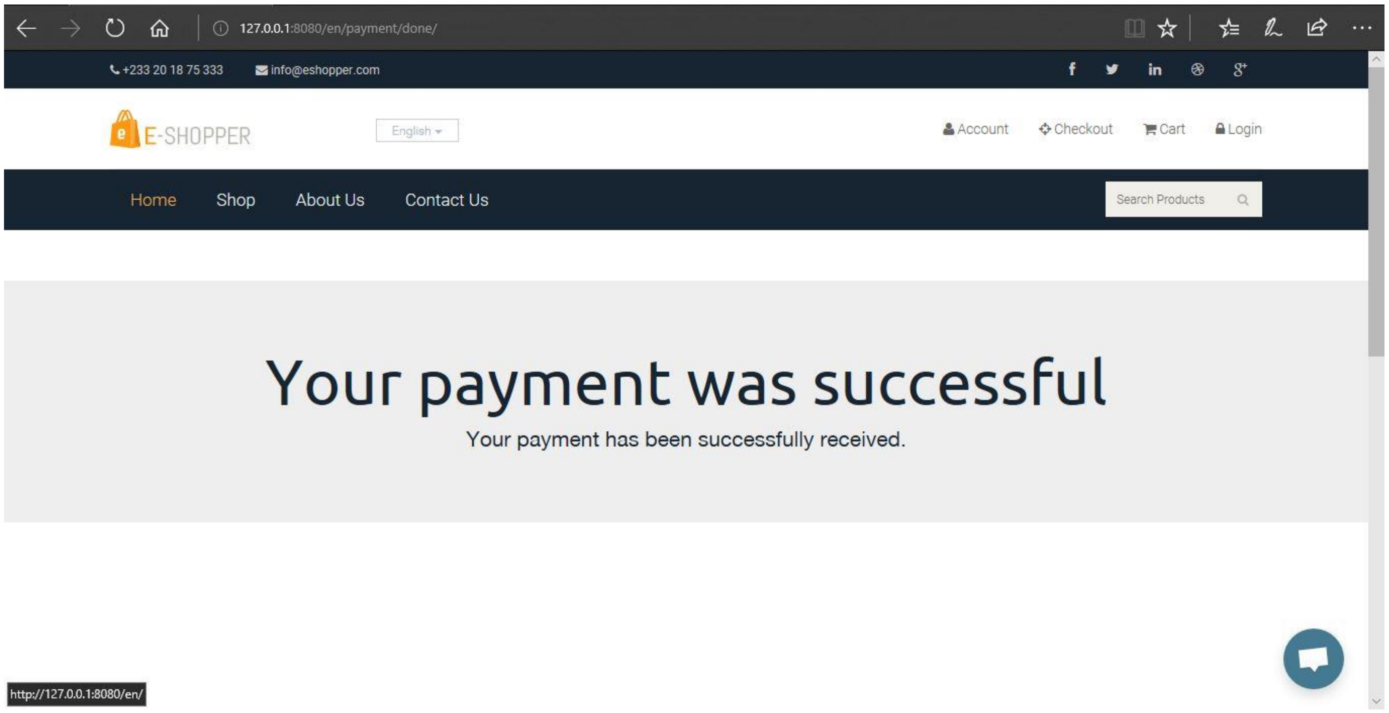
We'll send confirmation to:
daviose-buyer@mail.regent.edu

**Merchant details**

daviose@mail.regent.edu

Return to Merchant

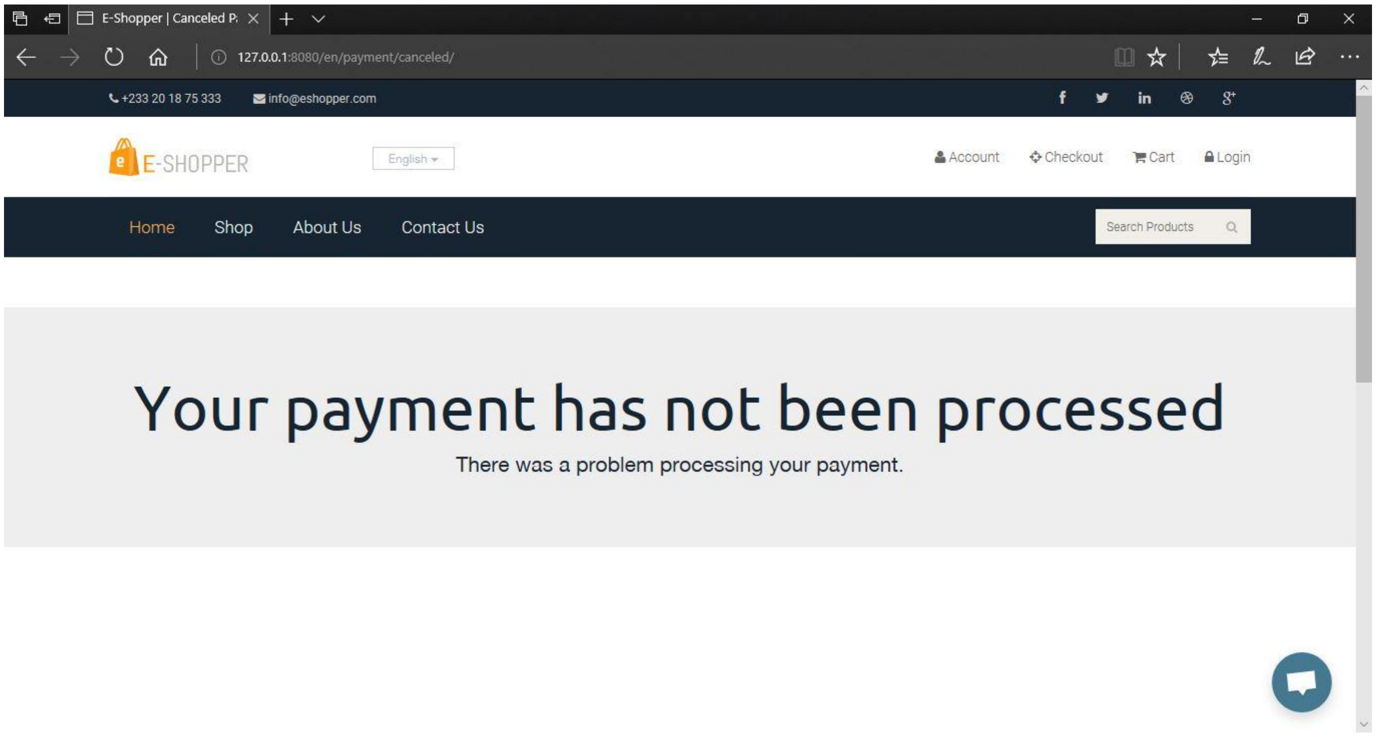
*Fig 5.15 PayPal Successful Payment page*



Your payment was successful

Your payment has been successfully received.

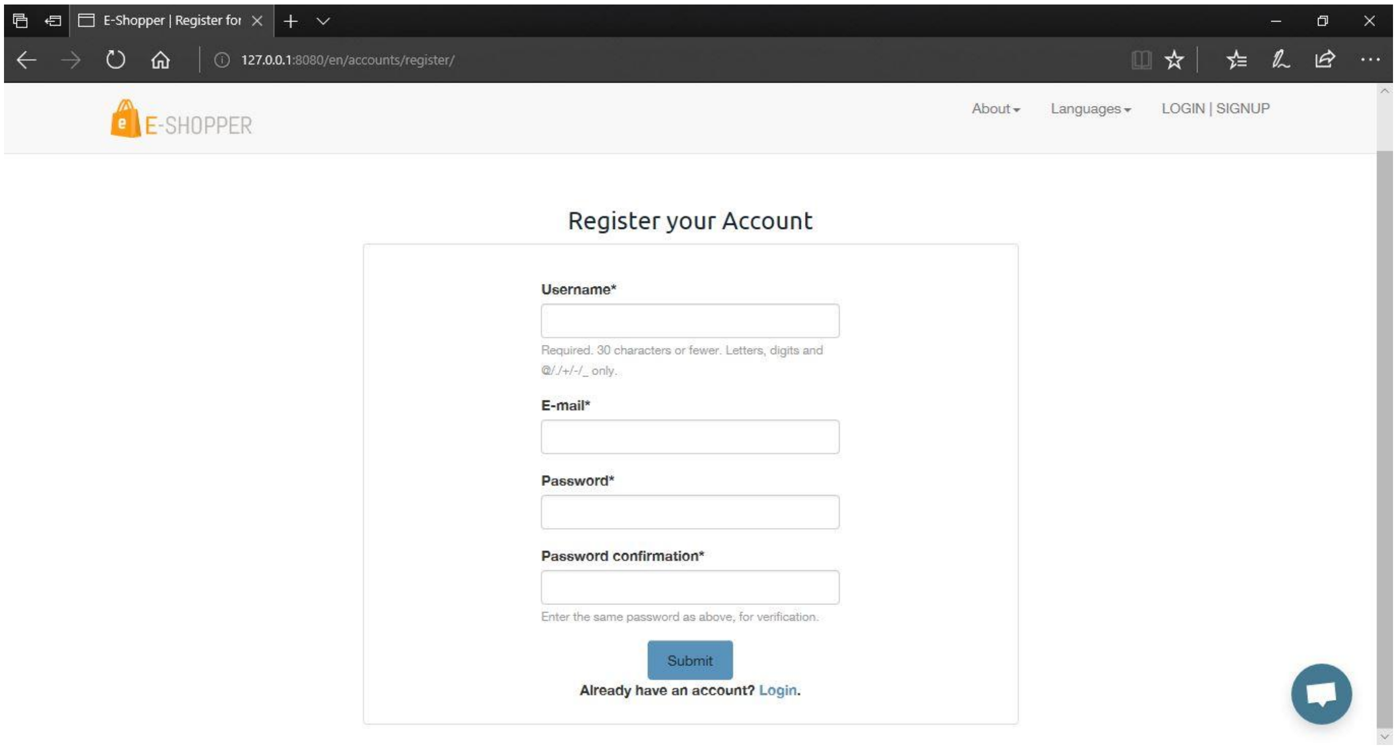
*Fig 5.16 Successful Payment Page*



Your payment has not been processed

There was a problem processing your payment.

*Fig 5.17 Canceled or Unsuccessful Payment Page.*



Register your Account

Username*

Required. 30 characters or fewer. Letters, digits and @/./+/-/_ only.

E-mail*

Password*

Password confirmation*

Enter the same password as above, for verification.

Already have an account? Login.

*Fig 5.18 Customer Register Account Page*

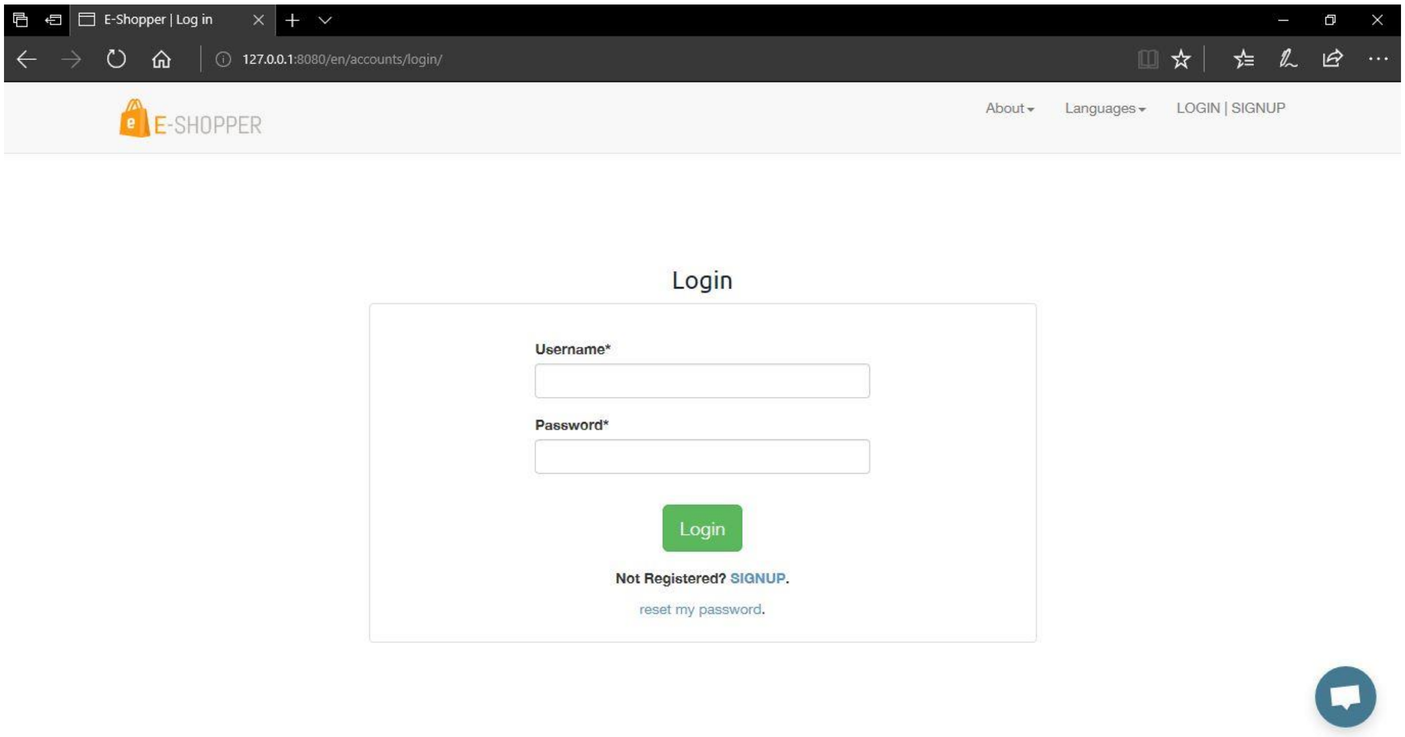
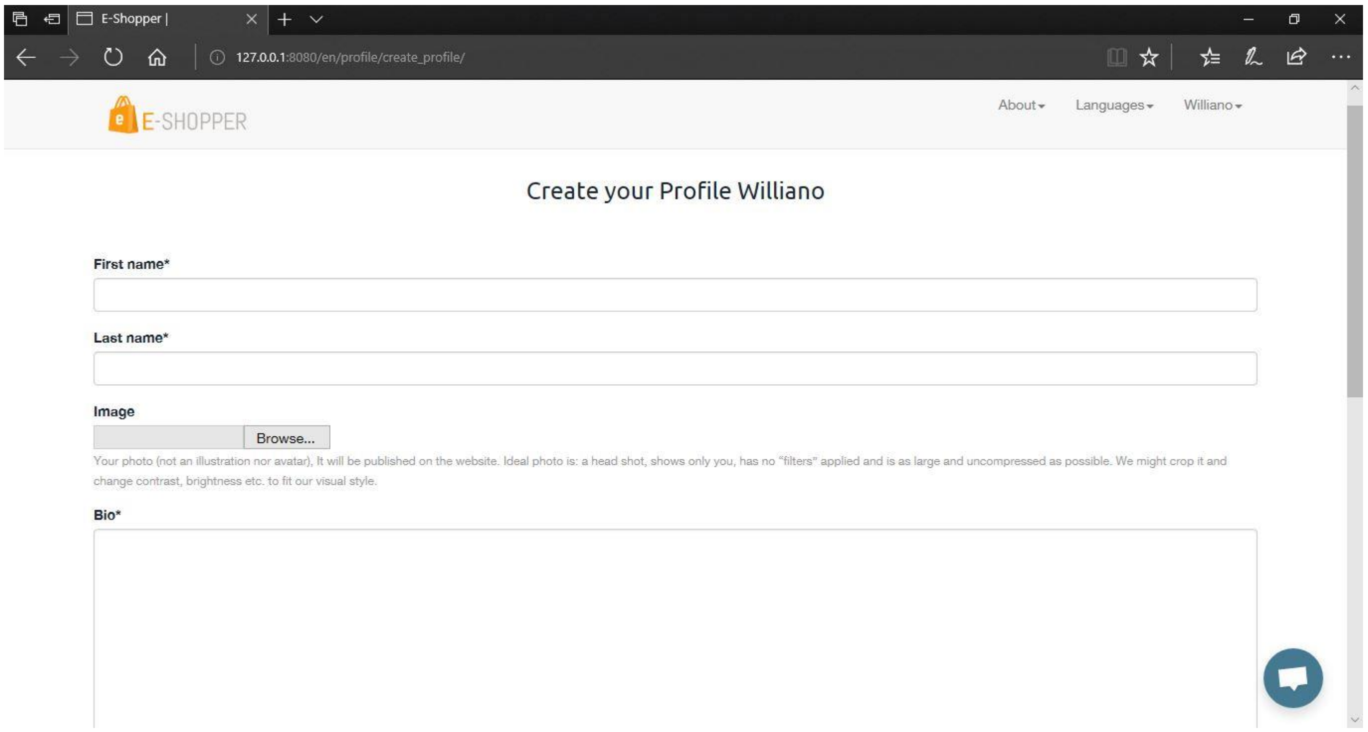
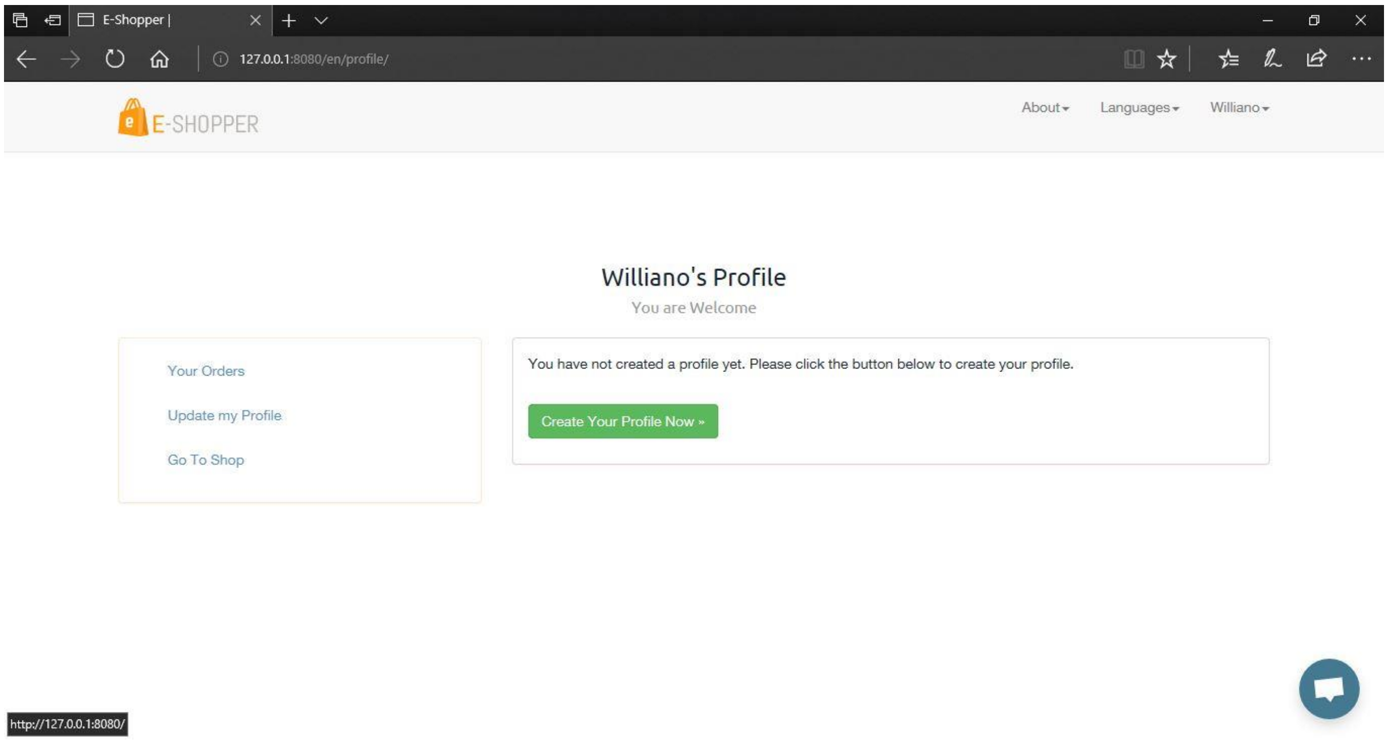


*Fig 5.19 Customer Login Page*



Your photo (not an illustration nor avatar). It will be published on the website. Ideal photo is: a head shot, shows only you, has no "filters" applied and is as large and uncompressed as possible. We might crop it and change contrast, brightness etc. to fit our visual style.

*Fig 5.20 Customer Create Profile Page*

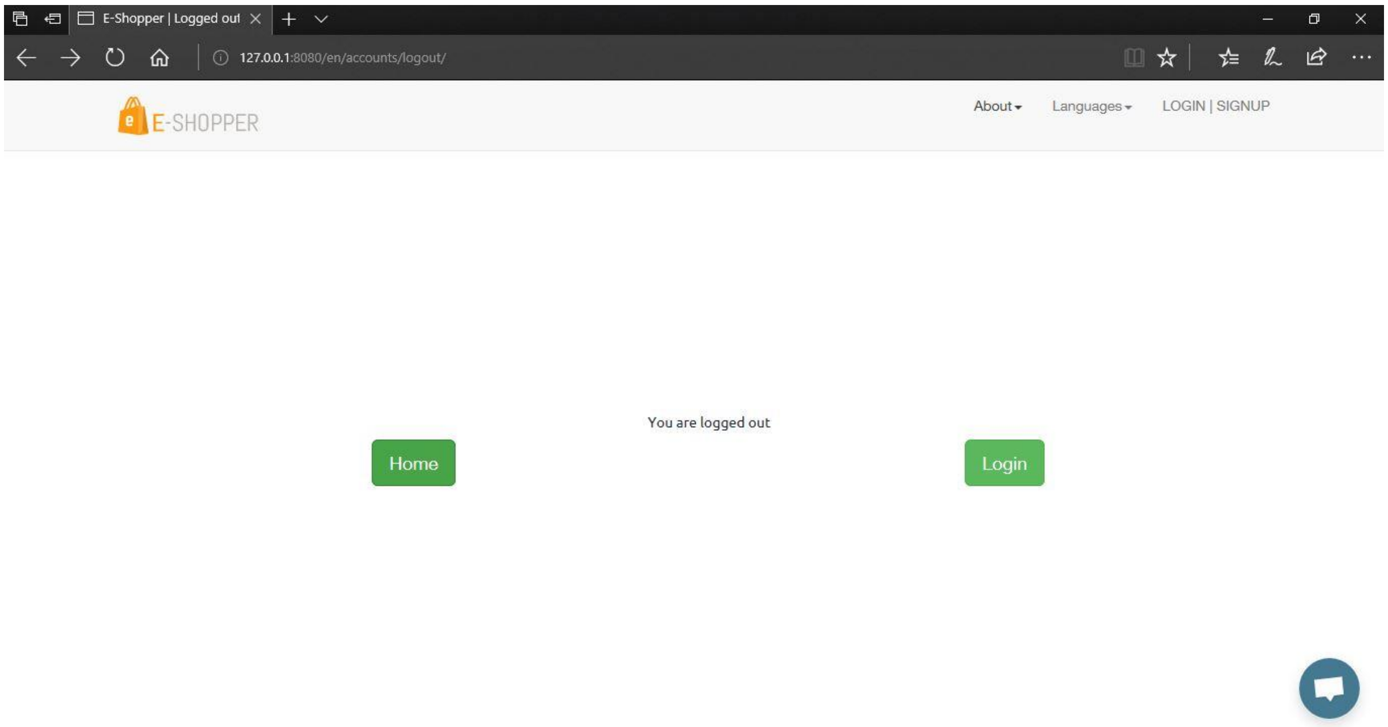


Williano's Profile

You are Welcome

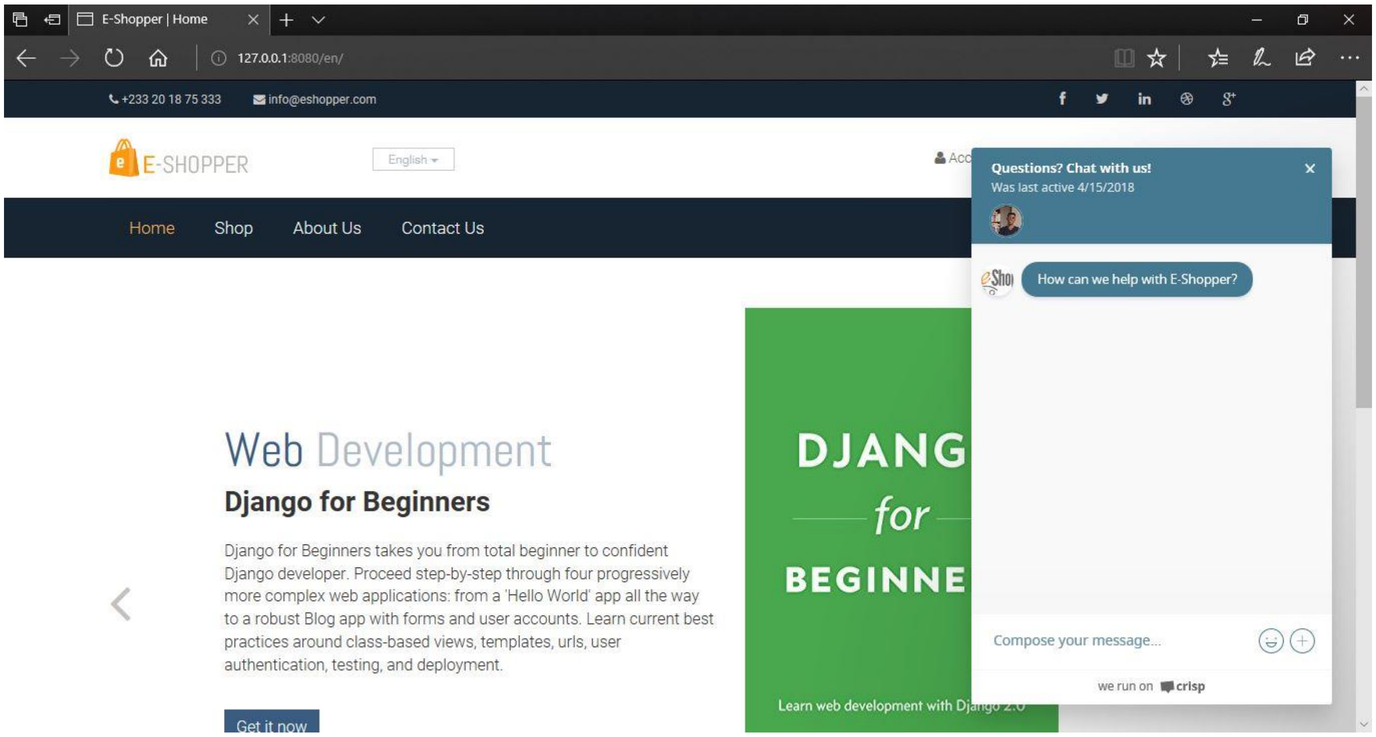
You have not created a profile yet. Please click the button below to create your profile.

*Fig 5.21 Customer Profile Dashboard*



You are logged out

*Fig 5.22 Customer Logout Page*



E-Shopper | Home

127.0.0.1:8080/en/

+233 20 18 75 333 info@eshopper.com

E-SHOPPER English

Home Shop About Us Contact Us

Questions? Chat with us!
Was last active 4/15/2018

How can we help with E-Shopper?

Compose your message...

we run on crisp

Web Development

Django for Beginners

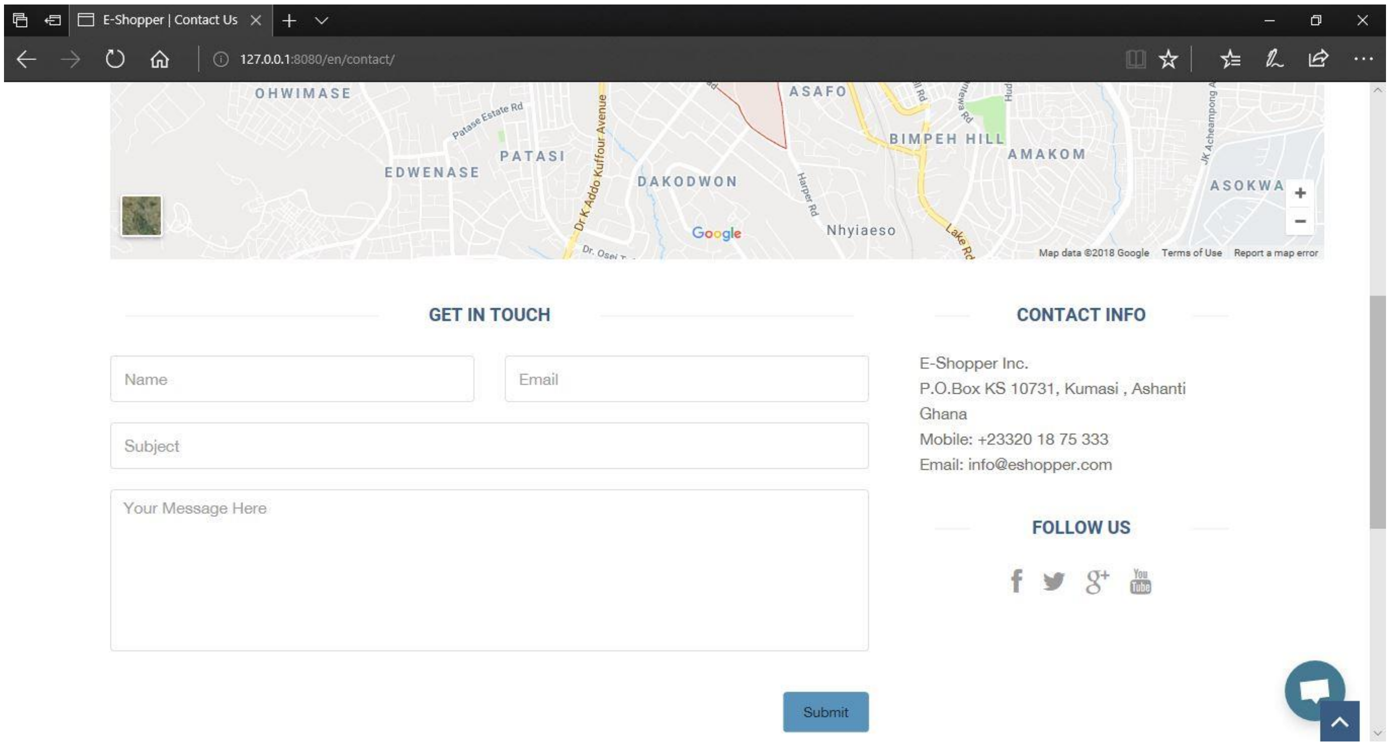
Django for Beginners takes you from total beginner to confident Django developer. Proceed step-by-step through four progressively more complex web applications: from a 'Hello World' app all the way to a robust Blog app with forms and user accounts. Learn current best practices around class-based views, templates, urls, user authentication, testing, and deployment.

Get it now

DJANGO for BEGINNERS

Learn web development with Django 2.0

*Fig 5.23 Live Chat Support*



E-Shopper | Contact Us

127.0.0.1:8080/en/contact/

GET IN TOUCH

Name Email

Subject

Your Message Here

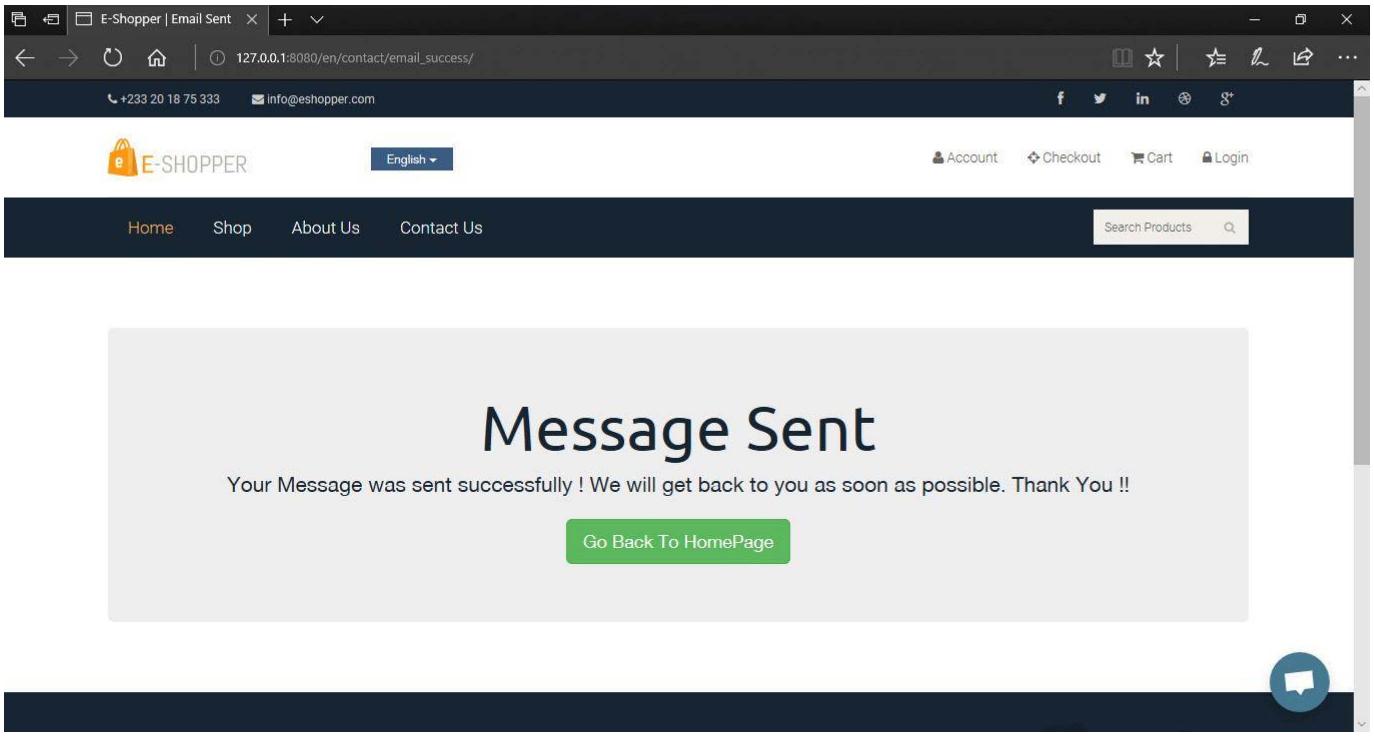
Submit

CONTACT INFO

E-Shopper Inc.
P.O.Box KS 10731, Kumasi , Ashanti
Ghana
Mobile: +23320 18 75 333
Email: info@eshopper.com

FOLLOW US

*Fig 5.24 Contact Us Page*



Message Sent

Your Message was sent successfully ! We will get back to you as soon as possible. Thank You !!

Go Back To HomePage

*Fig 5.25 Successful Contact Us Message sent page*

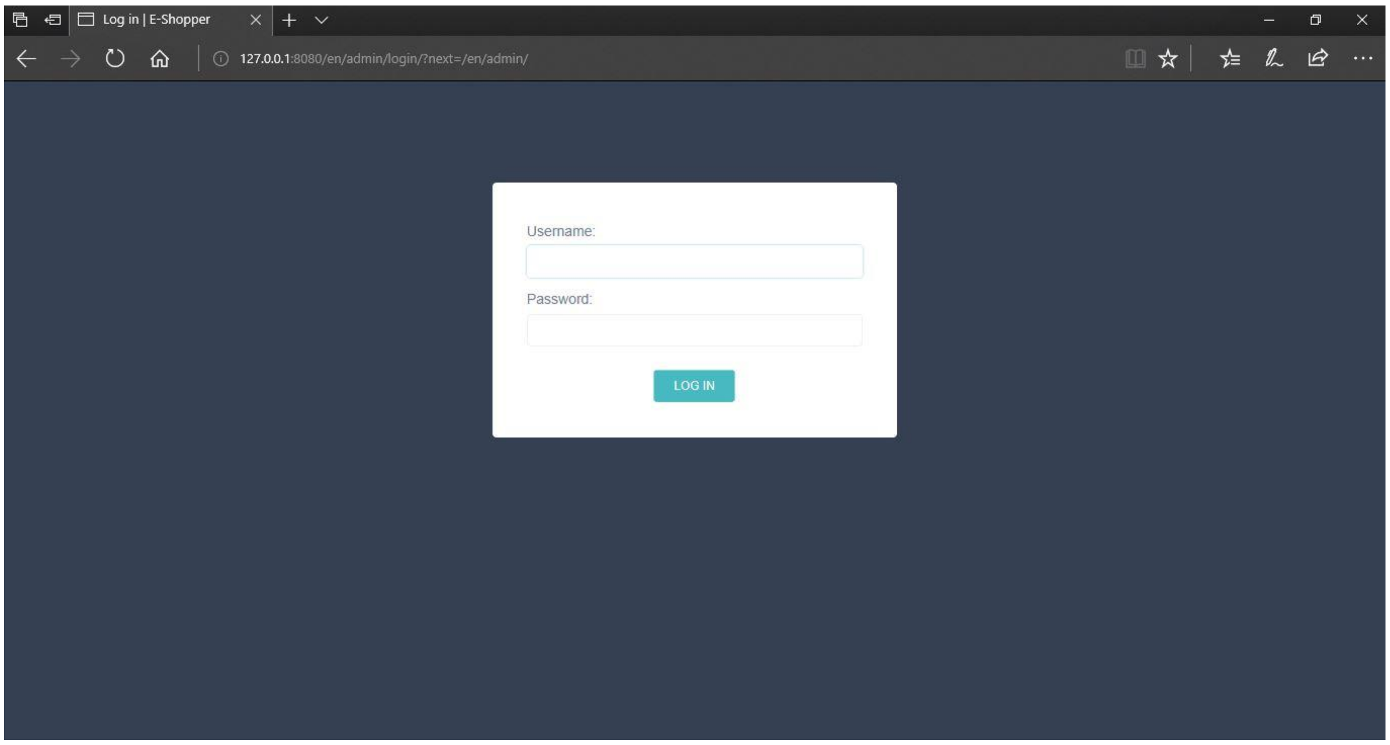
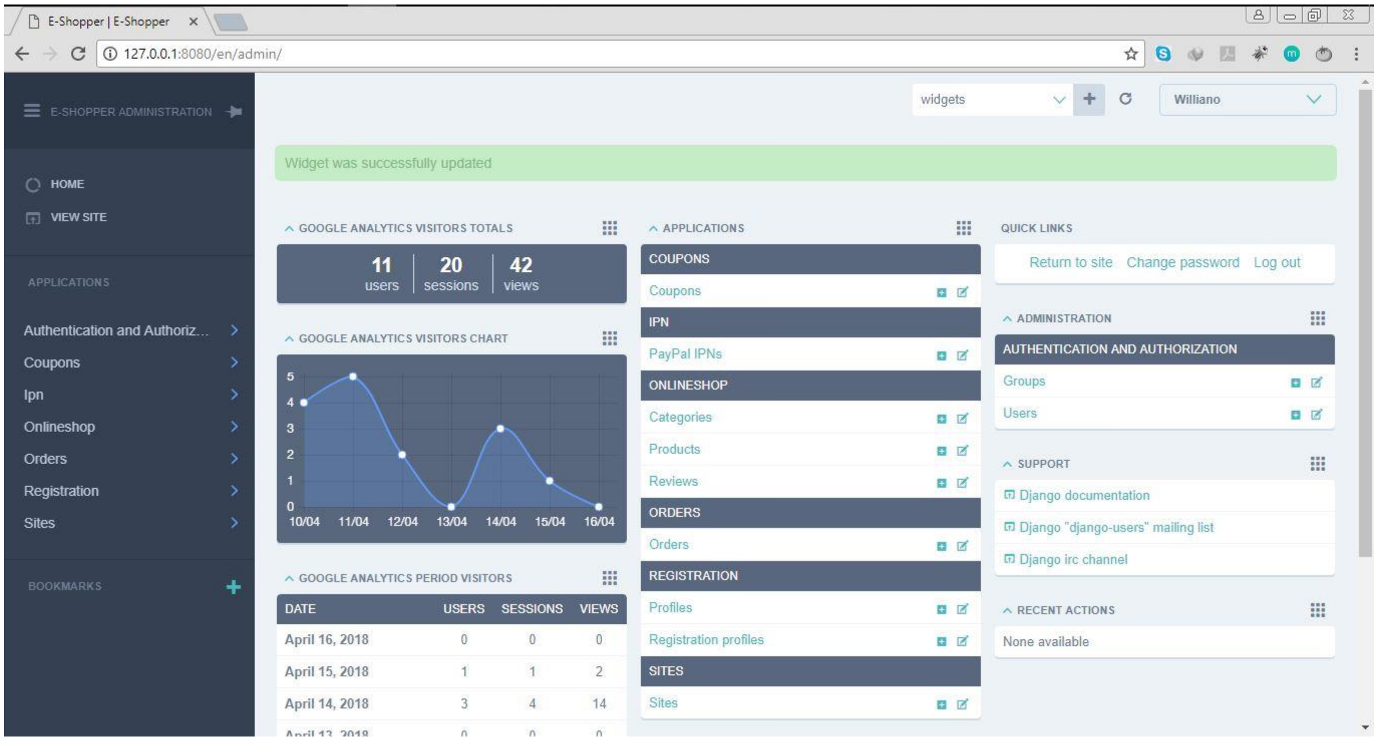


*Fig 5.26 Shop Owner or Manager Login Page*


E-Shopper | E-Shopper

127.0.0.1:8080/en/admin/

E-SHOPPER ADMINISTRATION

HOME

VIEW SITE

APPLICATIONS

Authentication and Authoriz...

Coupons

Ipn

Onlineshop

Orders

Registration

Sites

BOOKMARKS

widgets

Williano

Widget was successfully updated

GOOGLE ANALYTICS VISITORS TOTALS

11 users | 20 sessions | 42 views

GOOGLE ANALYTICS VISITORS CHART

GOOGLE ANALYTICS PERIOD VISITORS

| DATE | USERS | SESSIONS | VIEWS |
|---|---|---|---|
| April 16, 2018 | 0 | 0 | 0 |
| April 15, 2018 | 1 | 1 | 2 |
| April 14, 2018 | 3 | 4 | 14 |
| April 13, 2018 | 0 | 0 | 0 |

APPLICATIONS

COUPONS
- Coupons

IPN
- PayPal IPNs

ONLINESHOP
- Categories
- Products
- Reviews

ORDERS
- Orders

REGISTRATION
- Profiles
- Registration profiles

SITES
- Sites

QUICK LINKS

Return to site Change password Log out

ADMINISTRATION

AUTHENTICATION AND AUTHORIZATION
- Groups
- Users

SUPPORT
- Django documentation
- Django "django-users" mailing list
- Django irc channel

RECENT ACTIONS

None available

*Fig 5.27 Shop Manager or Owner Dashboard to accept orders, add products, add categories, add coupons with Analysis of people visiting the website*

## 5.3 TESTING

After the system developed, process of system testing must be carry on in order to test if the system is free of bugs. If during the system testing, there are bugs or errors detected, the developer may need to correct and fix the bugs immediately.

Testing is the process of evaluating a software or system to detect differences between given input and expected output. Testing is conducted to assess the quality of a system hence can be said to be a validation and verification process. This ensures that system meets the technical requirements that guided its design and development, works as expected and can be implemented with the same characteristics.

There are few types of system testing that must be performed which include the unit testing, integration testing, system testing, and acceptance testing. System testing is not a testing that is limited only to the development team but it also requires the help from specific outsider (beta-tester) to test on the system acceptance.

### 5.3.1 UNIT TESTING

Unit testing is a testing which requires the developer to test on every single part or component in the system. It is the practice of testing certain functions and areas known as units of a source code. This is important because it helps to verify whether the various units are functioning as expected, hence are returning the proper values, therefore it helps in identifying failures or errors in a source code. Every single step of unit testing will be recorded to the test plan for later testing review purposes. In the unit testing, the testing only involves members from the development team which mean beta-tester is not required.

With this system, various units were tested such as the product order unit, customer registration unit, payment unit among others to ensure they were returning the expected values. This system has two main units; one for the customer and one for the shop owner (manager). The two were tested to ensure that they functioned independently without errors in either of the two units affecting each other. This is relevant because it also helps in ensuring that the various units were error free and working in the most efficient manner.

### 5.3.2 INTEGRATION TESTING

Integration testing is a testing that must be conduct in order to test the integration between multiple pages of the system. After the unit testing, the entire units are integrated to form a complete system. The purpose of the integration testing is to make sure that there are no defects during the integration of multiple pages or modules. This is to detect inconsistencies between the various units after which system testing is performed on the system as a whole.

It is usually conducted after the unit testing. During the unit testing, the tester might not find any of the errors but it does not mean that the system will totally free of bugs since the system might not properly integrated which causes errors.

### 5.3.3 SYSTEM TESTING

System testing is a testing that must be conduct in order to test the complete system as a whole. The purpose of system testing is to test the whole application after it is considered completed. System testing is a very important testing since it requires the system to meets the requirements and quality set by the users.

This testing is conducted to evaluate the systems compliance with its specified requirements that is both functional and non- functional requirements.

In conducting system testing, some invalid input was entered into the system such as negative order quantity to ensure the system does not accept such values. This was successful because the system responded by alerting that the input was invalid. Also, the same was done with the date on which the order is to be delivered by entering dates that are past the default date. The system also alerted that the date was invalid. This enforced the fact that the system is validated and will not accept invalid input.

Also, the system's payment process was tested to ensure that an order is only successful when payment is successful. This also was successful; hence the system performs in harmony with the functional requirements.

With the non-functional requirements, the system is very interactive in that it gives feedback to the user for every required function. The system is also easy to use since it has an easy and friendly looking interface. The system was also tested on other platforms apart from the laptop such as a mobile phone and a tablet, and the system fit onto those devices and functioned well. This proved that the system is also scalable as required.

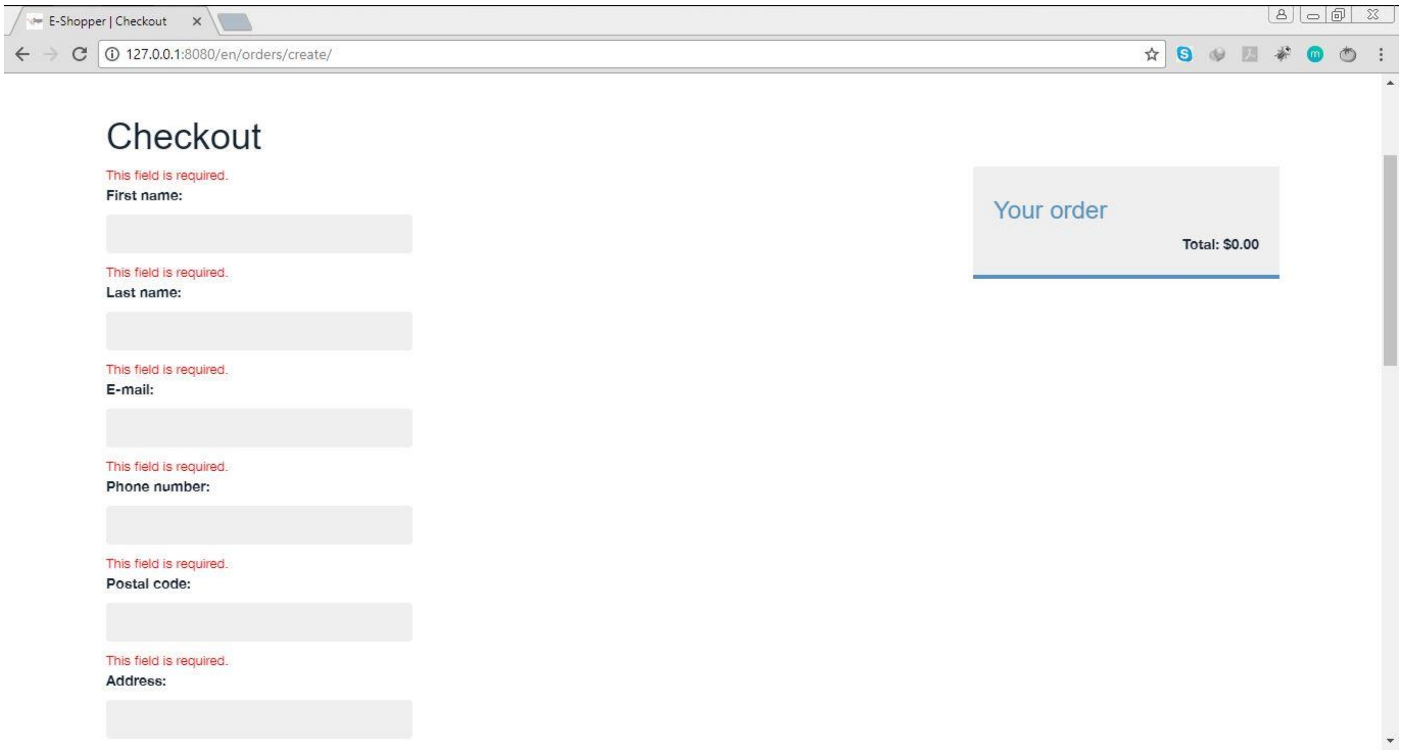
### 5.3.4 ACCEPTANCE TESTING

Acceptance testing which will involve the outsider (my supervisor) to test the system in order to find out if the system meets their requirements from all perspectives. Acceptance testing is usually the final testing conducted to ensure the system to be delivered meets the specifications and purpose.
Once the system successfully goes through all the testing, the system will more likely to be delivered to the real world for use.

### 5.3.3 RESULTS OF TESTING

Below are some screenshots from the various testing we carried out to ensure that the system was functioning in line with the requirements analysis we gathered from users.

As stated above, during the system testing, the application did not accept negative inputs.



E-Shopper | Checkout

127.0.0.1:8080/en/orders/create/

Checkout

This field is required.
First name:

This field is required.
Last name:

This field is required.
E-mail:

This field is required.
Phone number:

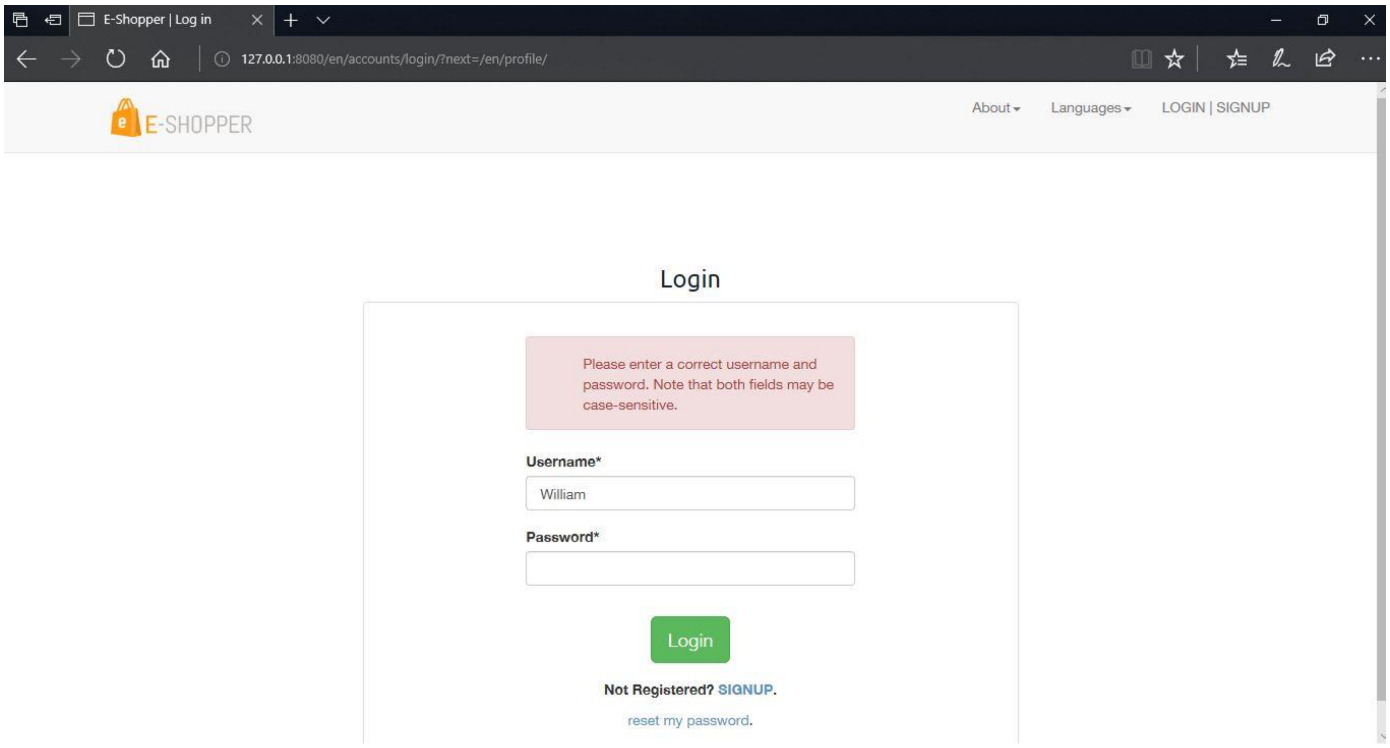
This field is required.
Postal code:

This field is required.
Address:

Your order

Total: $0.00

*Fig 5.28 Testing for a Customer checkout details.*



E-Shopper | Log in

127.0.0.1:8080/en/accounts/login/?next=/en/profile/

E-SHOPPER

About Languages LOGIN | SIGNUP

Login

Please enter a correct username and password. Note that both fields may be case-sensitive.

Username*: William

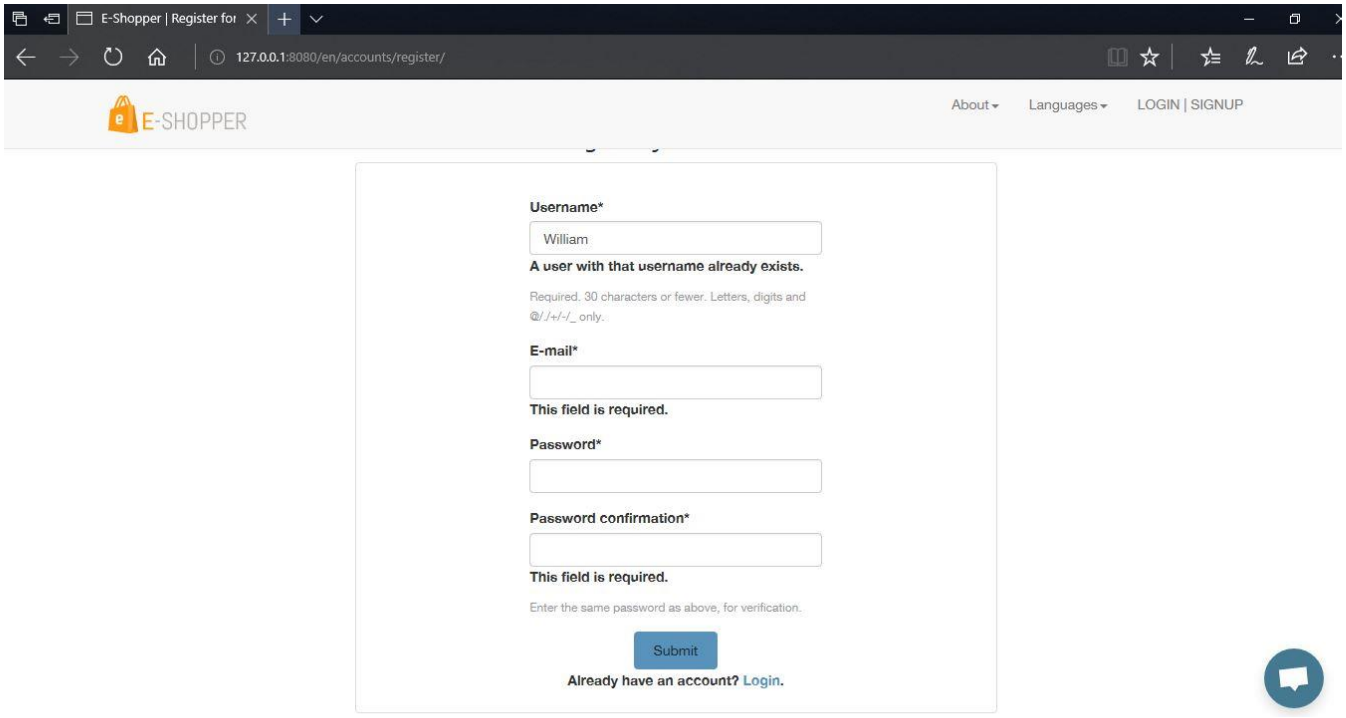
Password*:

Login

Not Registered? SIGNUP.

reset my password.

*Fig 5.29 Testing for a Customer login details.*



E-Shopper | Register for

127.0.0.1:8080/en/accounts/register/

E-SHOPPER

About Languages LOGIN | SIGNUP

Username*

William

A user with that username already exists.

Required. 30 characters or fewer. Letters, digits and @/./+/-/_ only.

E-mail*

This field is required.

Password*

Password confirmation*

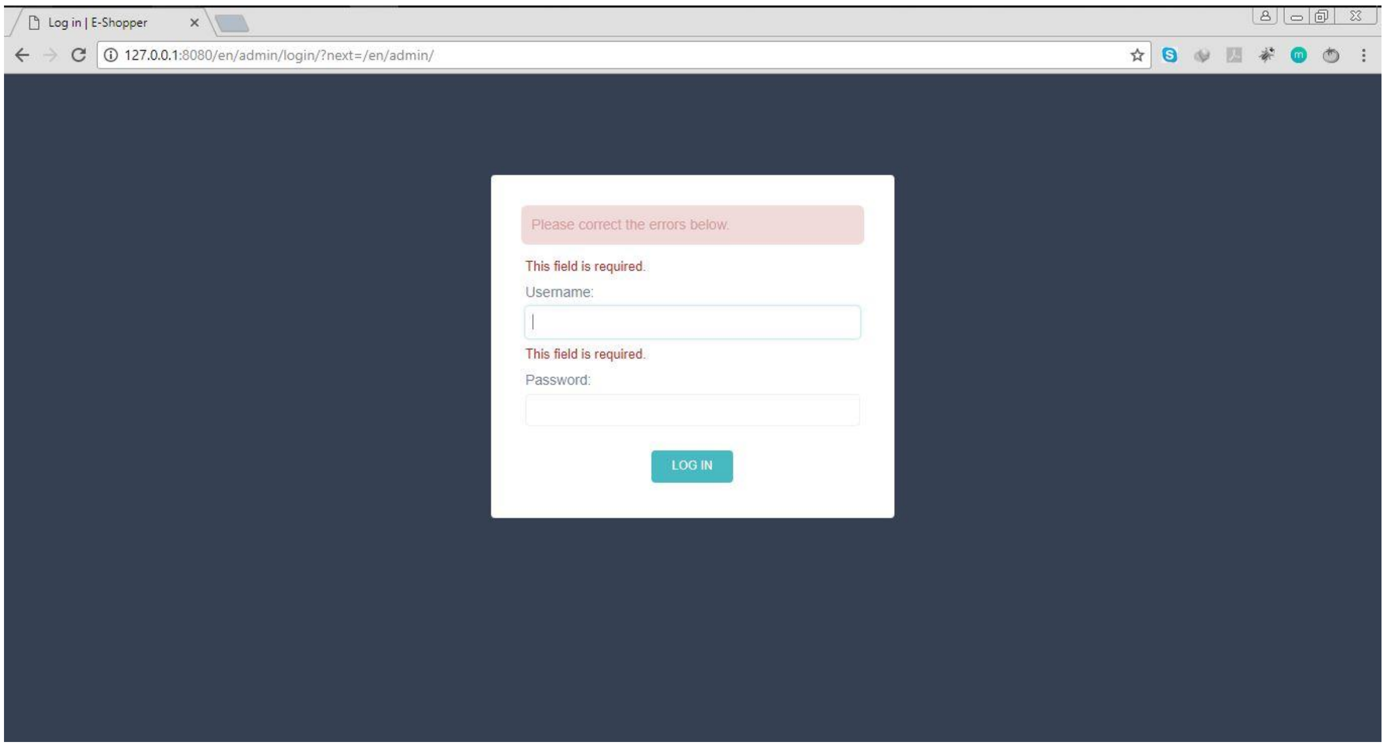
This field is required.

Enter the same password as above, for verification.

Submit

Already have an account? Login.

*Fig 5.30 Testing for a Customer Register details.*



Log in | E-Shopper

127.0.0.1:8080/en/admin/login/?next=/en/admin/

Please correct the errors below.

This field is required.

Username:

This field is required.

Password:

LOG IN

*Fig 5.31 Testing to ensure that the admin enters the right login in details.*

## 5.4 EVALUATION OF THE PROJECT

The purpose of evaluation is to assess the system as to whether it does what it is expected to do and if it is working properly. From the testing phase, the system was analyzed to ensure is functioning well according to the specified requirements.

One of the aims for designing and developing the online shop included easing customers of the stress they go through when shopping online. This system provides the solution to that, in that customers in their comfort can choose the language of their choice and place their orders. They also receive email confirming their order has been received and being processed which is very assuring.

The system is said to have an interactive interface which is also easy to use because the various products available are displayed on the shop or product listing page for customers to view and select their choice. The various products (books) have also been sorted into categories such as Web development, Software Engineering, Programming, Data Structures for easy searching and selection. Customers can therefore use the system as first timers without difficulties. These therefore will increase customer satisfaction as perceived in chapter one of this documentation.

Also, a payment system was implement as stated in the project scope. Customers can therefore only have a successful product order after payment has been done.

On the side of shop owners or managers, they can perform some managerial duties with the system such as edit products by adding books available or deleting books unavailable. With Google Analytics added, they can also see the number of people visiting their website. They can also reply to feedbacks from customers by sending them a direct email. They can generate reports in the form of PDF or Excel of the various orders received. This will help them access their performance over a specified period of time.

From the above, a conclusion can be made that from the evaluation of our project, the project meets its overall aim as stated in chapter one which is to design and implement a multi-lingual e-commerce system to enable online shoppers order products.

### 5.4.1 EVALUATION OF SOLUTION

Testing and evaluating the system has helped to realized that the system serves as a solution to the problem statement in chapter one. All that customers have to do is choose the language of their choice and make their orders without any language barrier. The problem of e-commerce having poor records is also solved. This is because managers can use the application to access reports and status of orders and can also download them in PDF or Excel format for record keeping.

From the above, a conclusion can be made that the system is a solution to the problem aimed to address and solve.

### 5.4.2 EVALUATION OF METHODOLOGY

The agile method with the incremental software process model was adopted to develop the application as stated in chapter three. This approach best suits E-commerce website development because the application was developed in increments and based on the suggestions of my supervisor, the necessary changes were made until the final phase was reached. This made the development much easier and helped to meet the various user and system requirements.

## 5.5 CONCLUSION

In this chapter, mapping the logical design onto the physical platform was looked at where the logical database schema was transformed into executable codes. A look at constructions where we the various user interfaces displayed were discussed in the previous chapter. Screen shots of the various forms, reports, etc. were captured under construction. Testing of the whole system was examined. Both unit, integration, system and acceptance testing were performed to ensure the software was working as required. Finally, evaluation of the project where highlights of the various efforts put in place to make this project a success was looked at.

# CHAPTER SIX

# FINDINGS AND CONCLUSION

## 6.0 INTRODUCTION

Findings and conclusion concerns itself with all the information gathered including facts and figures compiled to fulfil the objectives of a research or the development of a project as well as the illations drawn from the findings. Also included is the principal results of a research project as well as what the project suggested or indicated and the interpretations of the relevance of the findings of a project.

This final chapter will consist of findings and conclusion. This will include summary of various problems faced in the development of the system, achievements and challenges, recommendations as well as enhancements that can be made to the system in future.

## 6.1 SUMMARY OF PROBLEMS

Every research or project has challenges and this project is no exception. A number of problems were encountered during the development of the project. Mobile Money payment is a major payment platform in Africa and my goal was to integrate it into this project to easy payment for users. These major mobile money networks don’t provide an API for developers to interface other apps to use these services. There are third party companies that provide API but in only few programming languages like PHP and Java. I used the Python programming language to develop this project but all the effort to use this API’s was not successful because there was no API for the language. I made an effort to contact them but all of them said there was no API for the Python programming language. Because of this, I had to use PayPal as the only payment gateway to accept payments for both debit and credit cards.

The above were basically the problems encountered while working on this project.

## 6.2 ACHIEVEMENTS AND CHALLENGES

### 6.2.1 ACHIEVEMENTS

Through dedication, determination, effort, skill and hard work a number of successes and achievements were achieved these are stated below:

- Working alone has helped me to develop new skills like completing a project within a specific time. I used to do a lot of side projects but never finished any of them but this project has taught me to break-up projects into smaller pieces and accomplished them within a specific time.

- The development of the project required the use of a Python framework known as Django and also a database known as Postgresql which my first time of was using them but I was able to learn and use it for the development of the project. This have therefore helped me to enhance my knowledge and programming skills.

- Implementing the online shop in twenty different languages was the ultimate goal for this project. This is an achievement because the main motivations for undertaking this project included making online shopping easy for shoppers and shop owners. By means of internationalization and localization that problem is tackled.

### 6.2.2 CHALLENGES

Although the development of the project was a success, a number of challenges were encountered as well because every research or project has challenges and this project is no exception.

These include:

- One of the challenges faced was to complete the project on time. This is as a result of the fact that I had to make time for classes, mid-semester exams, final examinations and also for developing the project.
- Implementing the Mobile Money Payment system was the biggest challenge of this project. This was very difficult from the onset because there was no API's for the Python programming language.

## 6.3 RECOMMENDATIONS

Recommendations involve additions, suggestions or courses of actions that could be added to the project in the future.

Some recommendations for the system include:

- The project is purely web-based hence we recommend that a mobile version could be developed to make it more convenient for users to use especially now that mobile devices are abundant.
- Another recommendation is that more payment options should be added especially for the mobile money section. We have options only for either debit or credit cards hence limiting those who can use the application.
- Adding multiple currencies for user to choose their currency of choice before they start shopping is one recommendation for future works.

## 6.4 FUTURE WORK

In future, I hope to delve more into the integration of Mobile Money payments. This is because the payment aspect of the system does not offer this to the user. The hope is to expand the choices available to the shoppers. Also, I hope to add multiple currency to make users choose the currency of their choice before they start shopping.

## 6.5 CONCLUSION

The Internet has become a major resource in modern business, thus electronic shopping has gained significance not only from the entrepreneur's but also from the customer's point of view. For the entrepreneur, electronic shopping generates new business opportunities and for the customer, it makes comparative shopping possible.

As per a survey, most consumers of online stores are impulsive and usually make a decision to stay on a site within the first few seconds. "Website design is like a shop interior. If the shop looks poor or like hundreds of other shops the customer is most likely to skip to the other site.

Hence, the project has been designed to provide the user with easy navigation, retrieval of data and necessary feedback as much as possible. In this project, the user is provided with a multi-lingual ecommerce web site that can be used to buy books online. To implement this as a web application Django was used as the Technology. Django has several advantages such as enhanced performance, scalability, built-in security and simplicity.

To build any web application using the Django framework, the programming language needed is Python. Python was the language used to build this application. For the client browser to connect to the Django framework, Web Server Interface Gateway (WSGI) was used as the Web Server.

Django uses Object-Relational Mapping (ORM) to interact with the database as it provides in-memory caching that eliminates the need to contact the database server frequently and helps protect against SQL injection attack. PostgreSQL was used as back-end database since it is one of the most popular and open source databases, and it provides fast data access, easy installation and simplicity.

A good shopping cart design must be accompanied with user-friendly shopping cart application logic. It should be convenient for the customer to view the contents of their cart and to be able to remove or add items to their cart. The shopping cart application described in this project provides a number of features that are designed to make the customer more comfortable. This project helps in understanding the creation of an interactive web page and the technologies used to implement it. The design of the project which includes Data Model and Process Model illustrates how the database is built with different tables, how the data is accessed and processed from the tables.

The building of the project has given me a precise knowledge about how Django is used to develop a website, how it connects to the database to access the data and how the data and web pages are modified to provide the user with a shopping cart application.

# REFERENCES

10ecommercetrends. (2017, November 20). *10 Ecommerce Trends 2017*. Retrieved from 10 Ecommerce Trends : http://10ecommercetrends.com/

Ahiabenu, K. (2017, November 15). *How can e-commerce transform your business?* Retrieved from Graphic Online: https://www.graphic.com.gh/features/opinion/how-can-e-commerce-transform-your-business.html

ARM Worldwide . (2017, November 17). *What are the E-Commerce Tech Trends in 2018*. Retrieved from ARM Worldwide: https://armworldwide.com/e-commerce-tech-trends/

Boampong, P. B. (2017, November 13). *Breakthrough of E-commerce in Ghana*. Retrieved from Ghana Web: https://www.ghanaweb.com/GhanaHomePage/NewsArchive/Breakthrough-of-E-commerce-in-Ghana-347247

Codecademy. (2018, April 05). *MVC: Model, View, Controller*. Retrieved from Codecademy: https://www.codecademy.com/articles/mvc

Django Stars. (2018, January 15). *Why We Use Django Framework*. Retrieved from Django Stars: https://djangostars.com/blog/why-we-use-django-framework/

*eCommerce Report: Ghana's Top 20 eCommerce Websites*. (2017, November 22). Retrieved from Modern Ghana: https://www.modernghana.com/news/640481/ecommerce-report-ghanas-top-20-ecommerce-websites.html

Frederick, J. (2017, November 22). *4 Emerging Ecommerce Technology Innovations*. Retrieved from PFS Web: http://www.pfsweb.com/blog/4-emerging-ecommerce-technology-innovations/

Gary, H. (2017, November 22). *15 Must-Have Features for E-commerce Sites*. Retrieved from Search Engine Journal: https://www.searchenginejournal.com/15-must-have-features-for-e-commerce-sites/181974/

Gaur, R. (2018, April 05). *Software Architecture: One-Tier, Two-Tier, Three Tier, N Tier*. Retrieved from TECHYNEWS4U: https://techynews4u.com/software-architecture-one-tier-two-tier-three-tier-n-tier/

Inflectra. (2018, April 06). *What are System Requirements Specifications/Software (SRS)?* Retrieved from Inflectra: https://www.inflectra.com/ideas/topic/requirements-definition.aspx

Investopedia. (2017, November 20). *Business To Consumer - B To C*. Retrieved from Investopedia: https://www.investopedia.com/terms/b/btoc.asp

Kramek, A. (2018, April 05). *Introduction to Client Server Architecture*. Retrieved from Foxite: http://weblogs.foxite.com/andykramek/2008/09/29/introduction-to-client-server-architecture/

Miva. (2017, November 26). *The History Of Ecommerce: How Did It All Begin?* Retrieved from Miva: https://www.miva.com/blog/the-history-of-ecommerce-how-did-it-all-begin/

Nicole, F. (2016, April 5). *4 Issues Your E-Commerce Business Needs to Address*. Retrieved from Business News Daily: https://www.businessnewsdaily.com/8947-research-roundup-ecommerce-issues.html

Nige, B. (2018, January 15). *Why Django?* Retrieved from Django Book: https://djangobook.com/tutorials/why-django/

Openhatch. (2018, April 05). *Django for Designers/Basic views*. Retrieved from Openhatch: http://wiki.openhatch.org/Django_for_Designers/Basic_views

Opoku, J. (2017, November 25). *Setting up an e-commerce business in Ghana*. Retrieved from Sidekick Gh: https://www.sidekickgh.com/setting-up-an-e-commerce-business-in-ghana/

Opoku, J. (2017, October 17). *Why Responsive Websites are Awesome*. Retrieved from Sidekick Gh: https://www.sidekickgh.com/why-responsive-websites-are-awesome/

Post, J. (2017, December 19). *How to Beat 4 Big Challenges for Small E-Commerce Retailers*. Retrieved December 19, 2017, from Business News Daily: https://www.businessnewsdaily.com/6028-small-ecommerce-challenges.html

Rajkumar. (2018, April 05). *Software Architecture: One-Tier, Two-Tier, Three Tier, N Tier*. Retrieved from SoftwareTestingMaterial: https://www.softwaretestingmaterial.com/software-architecture/

Sheppy. (2018, January 17). *CSS: Cascading Style Sheets*. Retrieved from MDN web docs: https://developer.mozilla.org/en-US/docs/Web/CSS

SmartDraw. (2018, April 09). *Entity Relationship Diagram*. Retrieved from SmartDraw: https://www.smartdraw.com/entity-relationship-diagram/

Smith, L. (2018, January 25). *What PostgreSQL has over other open source SQL databases: Part I*. Retrieved from Compose: https://www.compose.com/articles/what-postgresql-has-over-other-open-source-sql-databases/

studytonight. (2018, April 09). *Normalization of Database*. Retrieved from studytonight: https://www.studytonight.com/dbms/database-normalization.php

Tawiah, A. (2017, November 22). *Ghana's Top 12 eCommerce Websites*. Retrieved from IT News Africa : http://www.itnewsafrica.com/2015/09/ghanas-top-12-ecommerce-websites/

Team, P. C. (2017, November 23). *7 Key Features Online Shoppers Demand From an Online Store*. Retrieved from PinnacleCart: https://www.pinnaclecart.com/blog/7-key-features-online-shoppers-demand-from-an-online-store/

The Editors of Encyclopaedia Britannica. (2018, April 06). *Database*. Retrieved from Britannica: https://www.britannica.com/technology/database

Visual Paradigm. (2018, April 06). *What is Unified Modeling Language (UML)?* Retrieved from Visual Paradigm: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/

Wikibooks. (2017, December 21). *Introduction to Computer Information Systems/E-Commerce*. Retrieved from Wikibooks: https://en.wikibooks.org/wiki/Introduction_to_Computer_Information_Systems/E-Commerce#Manufacturer_and_E-Tailer_Sites

Wikipedia. (2018, April 09). *User interface design*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/User_interface_design