**Smart Restaurant**

**A PROJECT REPORT**

**Submitted By**

Dhruvam Jhaveri(170770107057)

Vivek Thakkar(170770107239)

Swapnil Shah(170770107216)

Rohan Amin(170770107003)

***In fulfillment for the award of the degree of***

**BACHELOR OF ENGINEERING**

**In**

**Information Technology**



**SILVER OAK COLLEGE OF ENGINEERING AND TECHNOLOGY**

**OPP. BHAGWAT VIDYAPITH, NEAR GOTA CROSS ROAD, AHMEDABAD- 382481.**

**Candidate’s Declaration**

We hereby declare that project report titled “Smart Restaurant” submitted towards the completion of project in 8th semester of bachelor of Information Technology in Silver Oak College Of Engineering & Technology, Ahmedabad is an authenticate record of our work carried out.

We further declare that to the best of our knowledge the report of C.E. 8th semester.

First candidates name **: Jhaveri Dhruvam**

Branch **: CE**

Enrollment Number **: 170770107057**

Second candidates name **: Thakkar Vivek**

Branch **: CE**

Enrollment Number **: 170770107239**

Third candidates name **: Shah Swapnil**

Branch **: CE**

Enrollment Number **: 170770107216**

Fourth candidates name **: Amin Rohan**

Branch **: CE**

Enrollment Number **: 170770107003**

Submitted to:

**Silver Oak College Of Engineering & Technology**, Ahmedabad Affiliated to:**Gujarat Technological University**

**Abstract**

Smart restaurant is a python based web-application in which we are going to make restaurant as contactless as possible.

By using this web-application user can select tables according his choice depending on the number of persons and availability, user can also give order can see the menu of restaurant and can also pay the bill of restaurant. User can see the ingredients of the particular item. User can also add notes, update order and also can delete the order.

**ACKNOWLEDGEMENT**

We are heartily thankful to my supervisor, Mr. Haresh Parmar, whose encouragement, supervision and support from the preliminary to the concluding level enabled me to develop an understanding of the subject. At the end, we offer my regards and blessings to all of those who supported us in any respect during the completion of the project and to our college for providing a resources and materials.

|  |  |  |
| --- | --- | --- |
| **2.1** | **Incremental model** | **10** |
| **2.2** | **Pie chart** | **11** |
| **2.3** | **Roles & Responsibilities** | **12** |
| **4.1** | **Use case-Staff** | **25** |
| **4.2** | **Use case-System** | **26** |
| **4.3** | **Use case-User** | **27** |
| **4.4** | **Activity diagram-Admin** | **28** |
| **4.5** | **Activity diagram-Staff** | **29** |
| **4.6** | **Activity diagram-User** | **30** |
| **4.7** | **Sequence diagram-Admin** | **32** |
| **4.8** | **Sequence diagram-Staff** | **33** |
| **4.9** | **Sequence diagram-User** | **34** |
| **4.10** | **E-R diagram** | **36** |
| **4.11** | **Class diagram** | **38** |
| **5.1** | **Flow chart-admin** | **54** |
| **5.2** | **Flow chart-Staff** | **55** |
| **5.3** | **Flow chart-User** | **56** |
| **5.4** | **State transaction diagram** | **57** |

**Chapter No.**

1. **Introduction**

**Title Page**

**No.**

* 1. Project Summary 2
  2. Project Scope 2
  3. Objective 2
  4. Technologies and Literature Review **4**

1. **Software Project management**
   1. **Project Planning and Scheduling 8**
      1. Project Planning 8
      2. Project Scheduling 8
   2. **Project Development Approach 8**
   3. **Project Plan 10**
      1. Milestone and Deliverables 11
      2. Roles and Responsibilities 12
      3. Cost Estimation 12
      4. The COCOMO Model 13
2. **SYSTEM REQUIREMENT STUDY**
   1. User Characteristics 19
   2. Hardware and Software Characteristics 19
   3. Constraints **20**
      1. Regulatory Policies 20
      2. Hardware Limitation 20
3. **System Analysis**
   1. **Study of Current System 21**
   2. **Requirement of this System 21**
      1. Functional Requirement 21
      2. Non functional requirement 22
   3. **Feasibility Study 22**
   4. **Requirement Validation 23**
   5. **Function of the System 24**
      1. Use-Case 24
      2. Sequence Diagram 28
      3. Activity Diagram 31
   6. **Data Modeling 35**
      1. E-R Diagram 35
      2. Class Diagram 37
      3. Data Dictionary 38
   7. **Main modules of new system 46**
   8. **Justification 47**
4. **System Design**
   1. **Database design/database structure design 49**
      1. Mapping Object/Classes to Object 49
   2. **System procedural design 52**
      1. Design Pseudo code or algorithm for method or 52

operation

* + 1. Flow chart diagram 53
  1. **Data modeling 56**

5.3.2 State Transition Diagrams 56

6.0 **Implementation Planning and details**

6.1 Implementation Environment

(Single vs Multiuser, GUI vs Non GUI) 54

6.2 Program/Modules Specification 54

6.3 Security Features 55

6.4 Coding Standards 55

6.5 Sample Coding

**7.0 Testing (choose appropriate testing strategy or techniques suitable to your system)**

7.1 Testing Plan 57

7.2 Testing Strategy 58

7.3 Testing Methods 58

7.4 Test Cases (Purpose, Required output, Expected Result) 59

**CHAPTER: 1 INTRODUCTION**

**CHAPTER 1 INTRODUCTION**

* 1. **Project Summary**
     + In this project we are going to develop the python based web application which involves smart restaurant in Ahmadabad.
     + There are mainly three modules Customers , users and restaurants.
     + It includes various functionalities such as table selection, placing order, paying bills , adding notes , give feedback through our web-application.
     + There is much functionality like security, online payment integration etc.
  2. **Purpose:**
     + This application can make restaurant contactless with less staff and more safety
     + The task of restaurant and customers become easy by using this portal because they can manage in proper manner.
  3. **Scope:**

**The scopes are mention below:**

* + - **Manage Restaurant Table:** this is managed by restaurant manager according to the area of restaurant.
    - **Manage Menu:** Restaurant will manage it. They will also give information about particular food.
    - **Manage users:** customers are managed by completing their requirements. By giving response to them. They can login on the portal and be a part of the portal.
    - **Manage Employee :** admin will manage this.
  1. **Technology and Literature Review:**
     + The front end used in our project is HTML, JavaScript, CSS and the back end used is PYTHON, Django
     + Tools used are postgres, Pycharm.
     + We will follow the Iterative model for developing this Project.

**HTML**

* + - HTML an initialize of Hyper Text Markup Language for web pages.
    - It provides a means to describe the structure of text based information in document by denoting text as headings, paragraphs, lists and so on and to supplement that text with interactive forms, embedded images and other objects.

**JavaScript**

* + - JavaScript supports the development of both client and server components of web based applications.
    - On the client side, it can be used to write programs that are executed by a web browser within the context of the web page.
    - On the server side, it can be used to write web server programs that can be process information submitted by a web browser and then update the web browser display accordingly.

**Advantages:**

* + - It can be used for server side and client side scripting.
    - It is more flexible then JavaScript.

**Python technology:**

* + - Python is an interpreted, high-level and general-purpose programming language. Created by Guido van Rossum and first released in 1991.
    - Python is developer friendly and consistent technology.

**Benefit of the DJANGO framework:**

* + - DJANGO provide the following benefits:
      * Reduce programming effort.
      * Increase program speed and quality.

**Features of Django**

Rapid Development .

**Secure**.

Scalable.

Fully loaded.

Versatile.

Open Source.

Vast and Supported Community.

**SQL:**

* **SQL** (**Structured Query Language**) is a [special-purpose programming language](http://en.wikipedia.org/wiki/Special-purpose_programming_language) designed for managing data held in a [relational database management system](http://en.wikipedia.org/wiki/Relational_database_management_system) (RDBMS).
* Originally based upon [relational algebra](http://en.wikipedia.org/wiki/Relational_algebra) and [tuple relational calculus](http://en.wikipedia.org/wiki/Tuple_relational_calculus), SQL consists of a [data definition language](http://en.wikipedia.org/wiki/Data_definition_language) and a [data manipulation language.](http://en.wikipedia.org/wiki/Data_manipulation_language)
* The scope of SQL includes data insert, query, update and delete, [schema](http://en.wikipedia.org/wiki/Database_schema) creation and modification, and data **access control**. Although SQL is often described as, and to a great extent is, a [declarative language](http://en.wikipedia.org/wiki/Declarative_programming) ([4GL](http://en.wikipedia.org/wiki/4GL)), it also includes [procedural](http://en.wikipedia.org/wiki/Procedural_programming) elements.
* **Data Definition**: Defining tales and structure in the database.
* **Data manipulation:** Used to manipulate the data within those schema objects.

**CHAPTER: 2**

**SOFTWARE PROJECT MANAGEMENT**

**CHAPTER 2 SOFTWARE PROJECT MANAGEMENT**

* 1. **Project planning and scheduling**
     1. **Project Planning**
        + Project planning is part of project management, which relates to the use of schedules such as Gantt charts to plan and subsequently report progress within the project environment.
        + Initially, the project scope is defined and the appropriate methods for completing the project are determined. Following this step, the durations for the various tasks necessary to complete the work are listed and grouped into a work breakdown structure.
        + Project planning is often used to organize different areas of a project, including project plans, workloads and the management of teams and individuals.
     2. **Project Scheduling**
        + Project Scheduling is the culmination of a planning activity that is primary component of software project management.
        + When combined with estimation methods and risk analysis, scheduling, establishes a road map for the project management.
        + Scheduling begins with the process composition. The characteristics of the project are used to adapt an appropriate task set for the work to be done.
        + The task network is used to compute the critical project path, a time line chart and a variety of project information.
  2. **Project Development Approach**

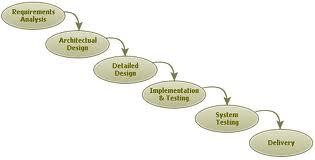
The activities we followed for this project is listed below:

* Planning the work or objectives
* Analysis & Design of objectives
* Assessing and controlling risk
* Allocation of resources
* Organizing the work
* Database Designing
* Form Design

The Process Paradigm we used for our project is Incremental Model.

**The Incremental Software Process Model**

* + The Incremental Model combines elements of the linear sequential model with the iterative philosophy of prototyping. The incremental model applies linear sequences in a staged fashion as calendar time progresses.
  + Each linear sequence produces a deliverable “increment” of the software. For example, word processing software developed using the incremental paradigm might deliver basic file management, editing and document production functions in the first increment; more sophisticated editing and document production capabilities in the second increment; spelling and grammar checking in the third increment; and advanced page layout capability in the fourth increment.
  + It should be noted that the process flow for any increment can incorporate the prototyping paradigm.
  + When an incremental model is used, the first increment is often a core product. That is, basic requirements are addressed, but many supplementary features remain undelivered.
  + The core product is used by the customer. As a result of use and/or evaluation, a plan is developed for the next increment. The plan addresses the modification of the core product to better meet the needs of the customer and the delivery of additional features and functionality.
  + This process is repeated following the delivery of each increment, until the complete product is produced.
  + The Incremental process model, like prototyping and other evolution approaches, is iterative in nature.
  + But unlike prototyping, the incremental model focuses on the delivery of an operational product with each increment.
  + Early increments are stripped down versions of the final product, but they do provide capability that serves the user and also provide a platform for evaluation by the user.

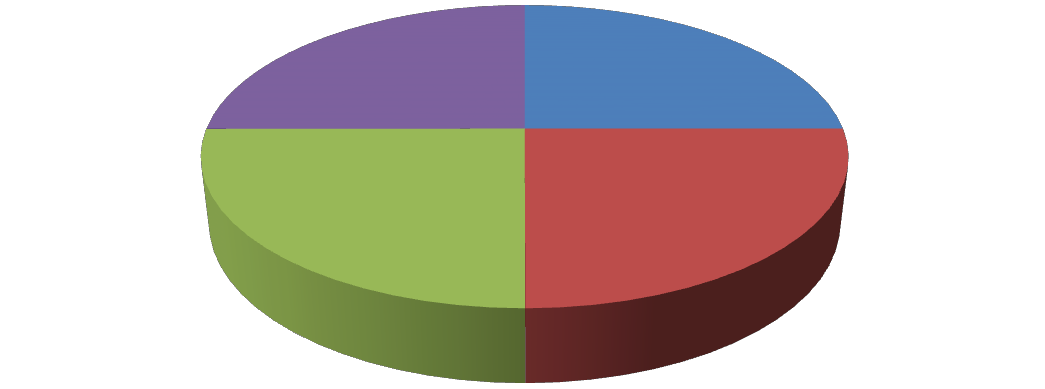


* 1. **Project Plan**

Figure 2.1 Incremental Model

|  |  |  |
| --- | --- | --- |
|  | **From Date** | **To Date** |
| **1.Preliminary Investigation** | 2/7/2020 | 4/8/2020 |
| **2.Requirment Analysis** | 5/8/2020 | 15/9/2020 |
| **3.Designing** | TBD | TBD |
| **4.Implementation** | TBD | TBD |
| **5.Testing** | TBD | TBD |

[Table 2. 1 Project Plan]



**Start Date**

Testing

25%

Analysis

25%

Coding

25%

design

25%

[ Figure 2.2 Pie Chart]

* + 1. **Milestone and Deliverables**
       - In this project, we went through Module Wise Completion. First we did analysis of first module; we went through all the requirements for first module that is Admin Module.
       - By this analysis we decided field of all the tables of Admin Module. Then we started Database Design.
       - After competing it we started with the design of all forms of this module. Then we did coding and finally validations and testing of forms that we made.
       - After completing the first module we started the same procedure for the Member Module.
       - After that we completed User Module. Between this we did settings forms e.g. Change theme, change background color of the main screen etc.
       - There was continuously interaction with the client that was very beneficial for us.
       - When we completed whole project we started testing the whole project for final verification.
       - Then we started documentation of our project. Finally, we completed the project with client’s satisfaction.
       - A milestone is an end point of software process activity.

|  |  |
| --- | --- |
| **Software Process Activity** | **Milestones** |
| Project Plan | Project Schedule |
| Requirement Collection | User requirements |
| Data Flow Analysis | System Flow |
| Design   1. Database Design 2. User Interface Design 3. System Design | System Design Document |
| Implementation   1. Code For giving security 2. Code for reports | Access Reports Reports |
| Testing | Setting validation and error message |

[Table 2. 2 Milestones and Deliverables]

* + 1. **Roles and Responsibilities**

**SMART RESTAURANT**

Dhruvam ,Swapnil

1. Investigation
2. Requirement Analysis
3. Coding
4. Testing

Vivek,Rohan

1. Investigation
2. Requirement Analysis
3. DB Design

[Figure 2.3 Roles and Responsibilities]

* + 1. **Gantt Chart**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *ID* | *Task Name* | *Start* | *Finish* | *Duration* | *2020* | | | | | | *2021* | | | | |
| *Jul* | *Aug* | *Sep* | *Oct* | *Nov* | *Dec* | *Jan* | *Feb* | *Mar* | *Apr* |  |
| 1 | Preliminary Investigation | 02/07/2020 | 04/08/2020 | 4.8w |  |  | | | | | | | | | |
| 2 | Requirement Analysis | 05/08/2020 | 16/09/2020 | 6.2w |  | | | | | | | | | | |
| 3 | Design | 22/10/2020 | 28/12/2020 | 9.4w |  | | | | | | | | | | |
| 4 | Imlimentation | 05/01/2021 | 10/02/2021 | 4.5w |  | | | | | | | | | | |
| 5 | Testing | TBD | TBD | ------ |  | | | | | | | | | | |

* + 1. **Cost Estimation: The COCOMO Model**
       - Like all estimation models for software, the COCOMO models require sizing information.
       - Three different sizing options are available as part of the model hierarchy: object points, function points, and lines of source code.
       - Like function points, the object point is indirect software that is computed using counts of the number of

1. Screens (at the user interface),
2. Reports,
3. Components likely to be required to build the application.
   * Once complexity is determined, the number of screens, reports, and components are weighted according to Table above.
   * The object point count is then determined by multiplying the original number of object instances by the weighting factor in table above and summing to obtain a total object point count.
   * When component based development or general software reuse is to be applied, the percent of reuse (%reuse) is estimated and the object point count is adjusted:
   * **NOP = (object points) X [(100 - %reuse) / 100].**
   * Where NOP is defined as new object points. To derive an estimate of effort based on the computed NOP value, a “productivity rate” must be derived.
   * **PROD=NOP / person-month**
   * For different levels of developer experience and development environment maturity. Once the productivity rate has been determined, an estimate of project effort can be derived as Estimated effort = NOP/PROD.
   * There are three types of software project:
     + Organic project
     + Semi-deteched project\
     + Embedded project

**Cost required to develop project=effort\*rs/month**

* + **Effort Estimation (E):**

In Organic=2.4 (KLOC)1.05 PM

In semidetached=3.0(KLOC)1.12 PM In Embedded=3.6(KLOC)1.20PM

* + **Duration Estimation (D):**

In Organic=2.5(effort)0.38months

In semidetached=2.5(effort)0.35 months In Embedded=2.5((effort)0.32months

* + **Person Estimation:**

P=E/D

**KLOC=Kilo Line of Code**

**In Or Project,**

|  |  |  |
| --- | --- | --- |
| **Modules** | **Page** | **Estimated line of code** |
| **Admin** | **-** | **4535** |
| Login | 100 |
| Manage tenders | 600 |
| Manage tanks | 800 |
| Manage area | 400 |
| Manage complain | 200 |
| Send mail/sms | 300 |
| Show/manage feedback | 300 |
| Manage notifications | 300 |
| Time schedule | 700 |
| Show report | 150 |
| **Vendor** | **-** | **4350** |
| Registration | 200 |
| Login | 100 |
| Show tender list | 500 |
| MANAGE LABOR | 500 |
| Manage complain | 900 |
| Send sms/mail | 300 |
| Show feedback | 300 |
| Show notification | 300 |
| Show tank list | 300 |
| Show time schedule | 600 |
| Manage report | 350 |
| **User** |  | **3650** |
| Registration | 200 |
| Home | 900 |
| About us | 150 |
| Contact us | 250 |

|  |  |  |
| --- | --- | --- |
|  | Login | 100 |
| List of tender | 100 |
| Complain | 200 |
| Feedback | 300 |
| Notification | 500 |
| Tank list | 350 |
| Time schedule | 600 |

[Table 2. 3 Cost Estimation]

* + Total line of code=12,535

 KLOC=12.54

We are using Organic Project Type,

**Effort Estimation (E):**

=2.4 (KLOC)1.05 PM

=33.30 PM

**Duration Estimation (D):**

= 2.5(effort)0.38 month

=2.5(33.30)0.38 months

=9.5 months

**Project Cost:**

=effort\*RS/month

=33.30\*12000

=3, 99,600 RS

**Advantages of COCOMO:**

* + - COCOMO is factual and easy to interpret.
    - One can clearly understand how it works.
* Accounts for various factors that affect cost of the project.
* Works on historical data and hence is **more** predictable and accurate.

**Disadvantages:**

* COCOMO model ignores requirements and all documentation.
* It ignores **customer skills**, cooperation, knowledge and other parameters.
* It oversimplifies the impact of safety/security aspects.
* It ignores hardware issues
* It ignores personnel turnover levels
* It is dependent on the amount of time spent in each phase.

**CHAPTER 3**

**SYSTEM REQUIREMENT**

**STUDY**

**CHAPTER 3 SYSTEM REQUIREMENT STUDY**

* 1. **SYSTEM REQUIRMENT STUDY**
  2. **User Characteristics:**
     + It describes the type of user which deals with the applications. Basically, this application has three types of users as given below:

1. Administrator
2. Employee
3. Customer(User)
4. **Administrator:**
   * Responsibility of administrator is to manage the application database and update the data in database regularly. For e.g. manage data of restaurant and customers
5. **Employee:**
   * Responsibility of employee is to complete tasks which are assigned to him by administrators and manage the some part of database.
6. **User**
   * Once the portal being tested and being uploaded then after customer (or end user) will use this application. User in the main reason for which this application is being built. The end user can easily interact with the system.
   1. **Hardware and Software Characteristics:**

**Hardware Requirements:**

* + - Minimum 2.27Ghz processor
    - RAM: 2GB minimum

**Software Requirements:**

* + - * Python compatible system

**Hardware requirement for python Application Development**

* + - * Minimum 2.0 GHz processor
      * Minimum 2GB of RAM
      * 100GB free space in Hard Disk storage

**Software requirement for Python Application Development**

* + - * Pycharm editor
      * Postgresql Database
      * Django
  1. **Constraints:**
     1. **Regulatory Policies:**
        + Regulatory Policies or mandates, limit the discretion of individual and agencies, or otherwise compel certain types of behavior.
        + These policies are generally thought to be best applied when good behavior can be easily defined and bad behavior can be easily regulated and punished through fine and sanctions.
     2. **Hardware Limitation**
        + The smooth functionality of the portal mainly depends on the speed of hardware and then on speed of the internet.
        + It is always advisable to be update as far as hardware is concerned. The hardware limitation occurs if the user is still using a very low MHz processor or a RAM or less than 128Mb.
        + This will generally reduce the portal and also the use will waste a lot of useful time, energy and resource.

**CHAPTER 4 SYSTEM ANALYSIS**

**CHAPTER 4 SYSTEM ANALYSIS**

* 1. **Study of Current System**
     + Currently there is a portal of similar kind but we are planning to update the system with some new features
  2. **Requirement of this System**
     + We are developing restaurant portal. By using this portal, the customer can easily interact with the restaurant system easily and efficiently and with less effort and maximum safety and contactless.
     1. **Functional Requirement:**

**User Requirement:**

* + - * User who uses this portal should know how to operate the Web Portal. Because the software has the same look and features like whole software is menu driven.

**Identification of functional requirement:**

* + - * The high level functional requirement often needs to be identified from an informal problem description document or from a conceptual understanding of the problem.
      * Each high level requirement characterizes away of system usage by some users to perform some meaningful piece of work.

**Documentation of functional requirement:**

* + - * For documenting the functional requirement we need to specify the set of functionalities supported by the system.

* + - * A function can be identified the state at which the data to input to the system, its input data domain, the output data domain, and the type of possessing to be carried out on the input data to obtain the output data.
    1. **Non functional requirement:**

**Usability:**

* + - * The interface should use terms and concepts, which are drawn from the experience of people who will make most of the system.\

**Efficiency:**

* + - * The portal must provide easy and fast access without consuming more cost.

**Readability:**

* + - * User should never be surprised by the behavior of the system and it should also provide meaningful feedback when error occurs so that user can recover from the error.

**Accuracy:**

* + - * The user should require that data are obtained from database and stored in database must be accurate.

**Security:**

* + - * The user wants the data stored in database must be secured and cannot be accessed by unauthorized user.

**Maintainability:**

* + - * User wants that the system should be maintained easily means that if there are some changes required in the system that can be done easily.
  1. **Feasibility Study**
* Feasibility is the measure of how beneficial the development of information system will be to an organization.
* The feasibility analysis is categorized under four different types.

1. Operational Feasibility
2. Technical Feasibility
3. Schedule Feasibility
4. Economic Feasibility
5. **Operational Feasibility:**
   * The System is to be developed for any user who wants to use it. We want our system user friendly and easy to use.
   * The administrator also may be non-technical, so the user interface will be designed in such a way that it gets comfortable for non-technical person to operate easily.
6. **Technical Feasibility:**
   * It is a partially measurement of specific technical solution and the availability of technical resorts and expertise.
   * The analyst must find out whether the current technical resources, which are available in the system is capable of handling the job.
   * If not, then the analyst with the help of developer should confirm whether the technology is available and capable or not.

**Better Considering:**

* + Here we have to consider those tools which are required for developing the project.
  + As far as basic knowledge concerned we have studied basic of Python, Javascript, Django and SQL.

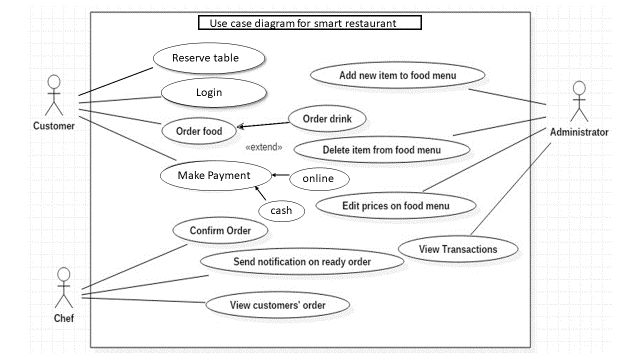
1. **Schedule Feasibility:**
   * Schedule feasibility corresponds to whether sufficient time is available to complete the project.

**Factor considered:**

* + Schedule of the project
  + Time by which project has to be completed
  + Reporting period

1. **Economic feasibility:**
   * Economic feasibility is a measure of cost effectiveness of a project or solution.
   * For declaring that the system is economically feasible, the benefits from the project should exceed or at least to the equal to the cost of development.
   1. **Requirement Validation:**

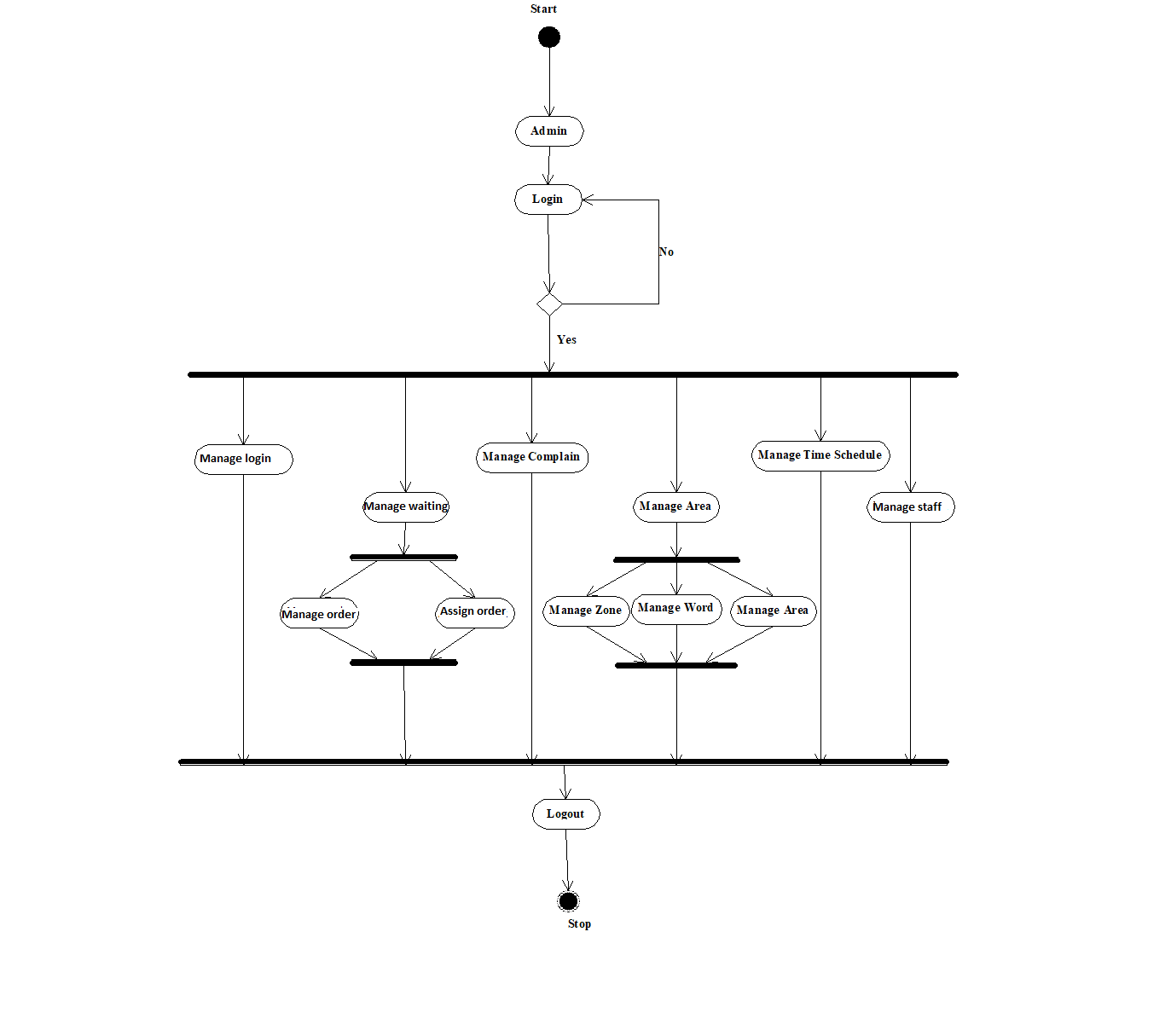
* Requirement validation examines this specification to ensure that all the system requirements have been stated unambiguously.
* These inconsistent, error have been detected and corrected and the work products confirmed to the standard.
* Source of the requirement are identified, final Statement of requirement has been examined by original source.
* Requirements related to main requirements are founds.
* Requirements are clarifying stated and are not misinterpreted.
* All sources of requirements are covered to get a maximum requirement.
* All method of finding requirements is applied.
  1. **Function of the System:**
     1. **Use-Case:**
        + In software and systems engineering, a **use case** is a list of steps, typically defining interactions between actor and a system, to achieve a goal.
        + The actor can be a human, an external system, or time.
        + In systems engineering, use cases are used at a higher level than within software engineering, often representing missions or stakeholder goals.
        + The detailed requirements may then be captured in Systems Modeling Language or as contractual statements.
        + As an important requirement technique, use cases have been widely used in modern software engineering over the last two decades.
        + Use case driven development is a key characteristic of process models and frameworks.
        + With its iterative and evolutionary nature, use case is also a good fit for agile development.



[Figure 4.1 Use-Case Vender]

* + 1. **Activity Diagram:**

**Admin:**

****

[Figure 4.4 Activity Diagram –Admin]

**Employee :**

**Start**



**Vender**

**Login**

**No**

**Yes**

**View Complain**

**View Notification**

**View Information**

**View Feedback**

**View Area**

**View Time Schedule**

**Logout**

**Stop**

[Figure 4.5 Activity Diagram-Vender]

**User:**

**Start**



**User**

**See waiting time**

**See Menu**

**Place Order**

**Pay Bills**

**Send Feedback**

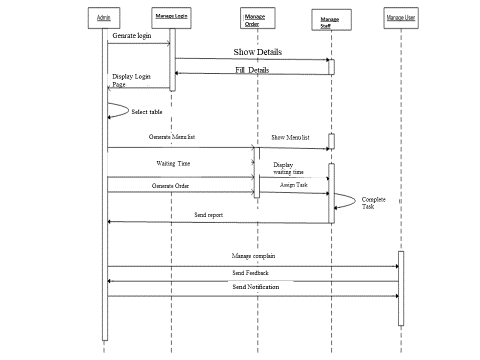
**Logout**

**Stop**

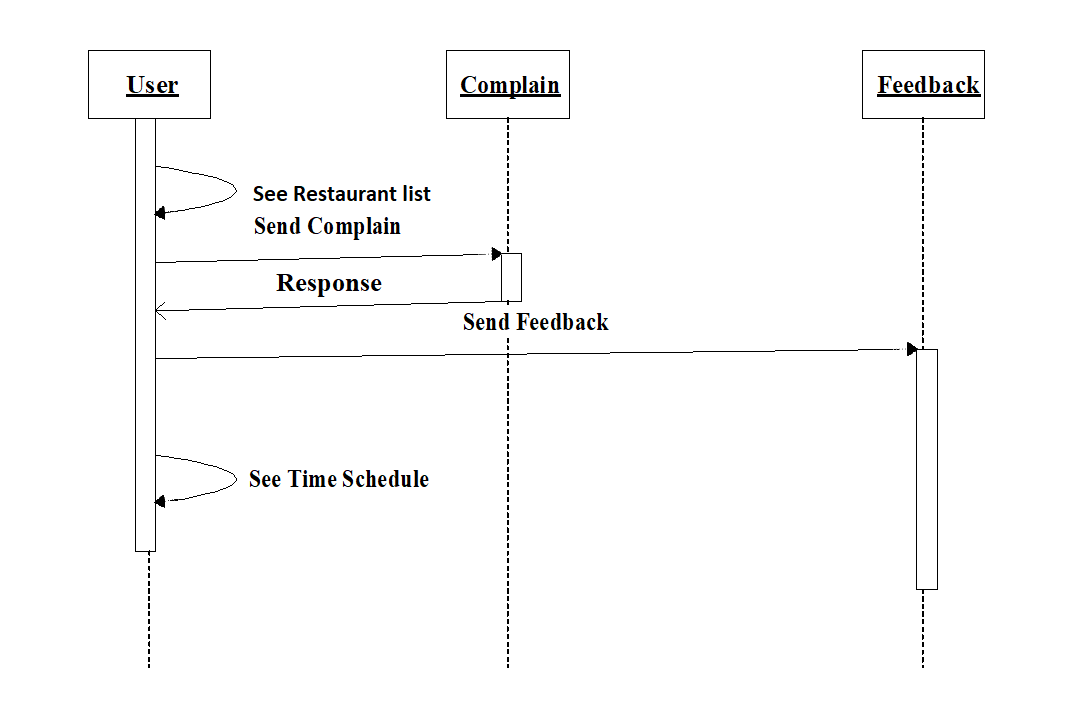
[Figure 4.6 Activity Diagram-User]

* + 1. **Sequence Diagram:**
       - The well-known [Message Sequence Chart](http://en.wikipedia.org/wiki/Message_Sequence_Chart) technique has been incorporated into the [Unified Modeling Language (UML) diagram](http://en.wikipedia.org/wiki/Unified_Modeling_Language) under the name of **Sequence Diagram**.
       - A sequence diagram shows, as parallel vertical lines, different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur.
       - This allows the specification of simple runtime scenarios in a graphical manner.
* The well-known [Message Sequence Chart](http://en.wikipedia.org/wiki/Message_Sequence_Chart) technique has been incorporated into the [Unified Modeling Language (UML) diagram](http://en.wikipedia.org/wiki/Unified_Modeling_Language) under the name of **Sequence Diagram**. A sequence diagram shows, as parallel vertical lines, different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

**Admin**

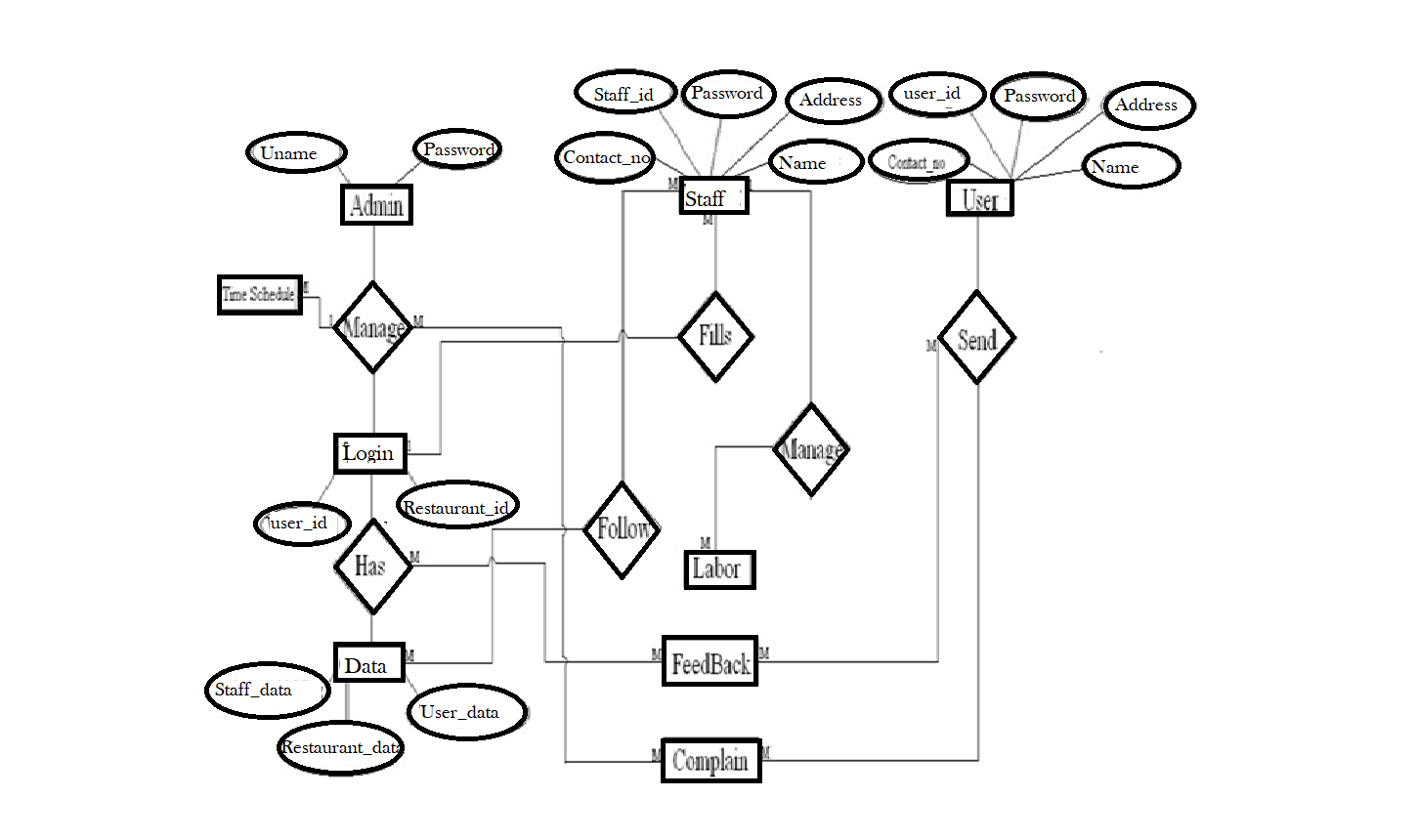
****

[Figure 4.7 Sequence Diagram –Admin]



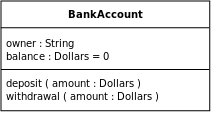
[Figure 4.8 Sequence Diagram-Vender]

* 1. **Data Modeling:**
     1. **E-R Diagram:**
        + In software engineering, an **entity–relationship model** (**ER model**) is a data model for describing the data or information aspects of a business domain or its process requirements, in an abstract way that lends itself to ultimately being implemented in a database such as a relational database.
        + The main components of ER models are entities and the relationships that can exist among them, and databases.
        + An entity-relationship model is a systematic way of describing and defining a business process.
        + The process is modeled as components (entities) that are linked with each other by relationships that express the dependencies and requirements between them, such as: one building may be divided into zero or more apartments, but one apartment can only be located in one building. Entities may have various properties (attributes) that characterize them.
        + Diagrams created to represent these entities, attributes, and relationships graphically are called entity–relationship diagrams.

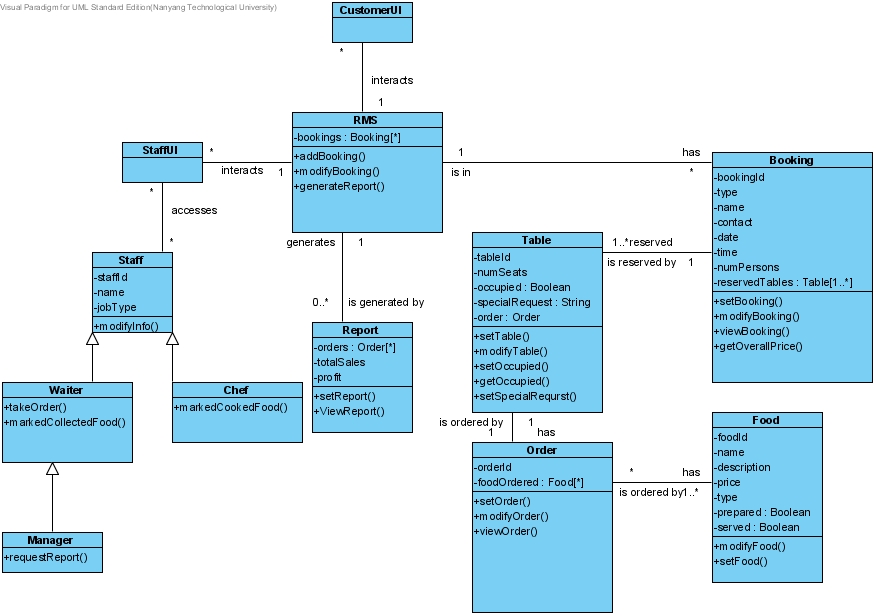


[Figure 4.10 E-R Diagram]

* + 1. **Class Diagram:**
       - In software engineering, a **class diagram** in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.
       - The class diagram is the main building block of object oriented modeling.
       - It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code.
       - Class diagrams can also be used for data modeling.
       - The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed.



* + - * In the diagram, classes are represented with boxes which contain three parts:
      * The top part contains the name of the class. It is printed in Bold, centered and the first letter capitalized.
      * The middle part contains the attributes of the class. They are left aligned and the first letter is lower case.
      * The bottom part gives the methods or operations the class can take or undertake. They are also left aligned and the first letter is lower case.



* + 1. **Data Dictionary**

[Figure 4.11 Class Diagram]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name:** Smart Restaurant | | | **Table Name:** Employee\_master | | |
| **Primary key:**Employee\_id | | | **Foreign key:** - | | |
| **Table Description:** This table is for description about tender | | | | | |
| **Table Structure** | | | | | |
| **Field Name** | **Data Type** | **Size** | | **Constrains** | **Description** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Employee\_id | Int | 20 | Primary key | This field contains Employee identity. |
| Employee\_first\_name | Varchar | 50 | Not null | This field contains Employee’s first Name. |
| Employee\_middle\_name | Varchar | 50 | Not null | This field contains Employee’s middle Name. |
| Employee\_last\_name | Varchar | 50 | Not null | This field contains Employee’s last Name. |
| Pofile pic | Imagefield | - | Not null | This field contains profile picture. |
| Id\_proof | Imagefield | - | Not null | This field contains id proof |
| Mobile\_number | Varchar | 10 | Not null | This field contains mobile number. |
| Email\_id | Email\_id | 50 | Not null | This field contains Email\_id. |
| Type | Varchar | 50 | Not null | This field contains  Type of employee. |
| Address | Textfield | - | Not null | This field contains  Address of employee |

[Table 4. 1 employee\_master]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name:** Smart restaurant | | | **Table Name:** Customer | | |
| **Primary key:** customer\_id | | | **Foreign key:** - | | |
| **Table Description:** This table is for description about customer. | | | | | |
| **Table Structure** | | | | | |
| **Field Name** | **Data Type** | **Size** | | **Constrains** | **Description** |
| customer\_id | Int | 20 | | Primary key | This field contains customer identity. |
| Customer\_first\_name | Varchar | 40 | | Not null | This field contains customer Name. |
| Customer\_middle\_name | Varchar | 40 | | Not null | This field contains customer Name. |
| Customer\_last\_name | Varchar | 40 | | Not null | This field contains customer Name. |
| Customer\_Email | Email | 40 | | Not null | This field contains customer’s Email. |
| Mobile\_number | Varchar | 15 | | Not Null | This field contains customer’s Mobile number |

[Table 4. 2 customer\_regular]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name:** Smart restaurant | | | **Table Name:** Menu | | |
| **Primary key:** Item\_id | | | **Foreign key:** table\_id | | |
| **Table Description:** This table is for description about menu. | | | | | |
| **Table Structure** | | | | | |
| **Field Name** | **Data Type** | **Size** | | **Constrains** | **Description** |
| item\_id | Int | 20 | | Primary key | This field contains item id. |
| Item\_name | Varchar | 40 | | Not null | This field contains item Name. |
| Description | Textfield | - | | Not null | This field contains description of item. |
| Type | Varchar | 40 | | Notnull | This field contains food type. |
| Price | Integer | 5 | | Notnull | This field contains price of the item. |
| Image | Imagefield |  | | Notnull | This field contains image of food. |
| Table\_id | integer |  | | Notnull | This field contains table id. |
| Quantity | Integer | 10 | | Notnull | This field contains quantity of food. |
| Status | Varchar | 10 | | Notnull | This field contains  Status of table |

[Table 4. 3 menu\_table]

|  |  |
| --- | --- |
| **Project Name:** Smart restaurant | **Table Name:** Table |
| **Primary key:** table\_id | **Foreign key:** - |
| **Table Description:** This table is for description about area. | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table Structure** | | | | |
| **Field Name** | **Data Type** | **Size** | **Constrains** | **Description** |
| table\_id | Int | 20 | Primary key | This field contains table identity. |
| Capacity | Int | 5 | Not null | This field contains capacity of the table |

[Table 4. 4 table\_regular]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name:** Smart restaurant | | | **Table Name:** bill\_master | | |
| **Primary key:** bill\_id | | | **Foreign key:** table\_id | | |
| **Table Description:** This table is for description about bill. | | | | | |
| **Table Structure** | | | | | |
| **Field Name** | **Data Type** | **Size** | | **Constrains** | **Description** |
| bill\_id | Int | 10 | | Primary key | This field contains bill identity. |
| Bill\_number | Varchar | 40 | | Not null | This field contains bill Number. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Type | varchar | 90 | Not null | This field contains type of bill |
| Bill\_amouunt | Integer | 10 | Not null | This field contains amount of bill |
| Date | Datefield |  | Not null | This field contains date of bill, |

[Table 4. 5 bill\_master]

* 1. **Main Modules of New System**
     + The water tank purification can be used to manage water tanks by AMC. It will support both standalone and also networking environment.
     + The main modules involved in this system are: 1Login

2Admin access 3Employee

4User interaction

* + - **Module Wise description**

1. **Login and contact us:**
   * Login module is used to check whether the user id authorized person or not.
   * For this user should give correct user name and password. The different types of user are: admin, employee, users.
2. **Admin access**
   * The menu are managed by admin and also admin is responsible for assigning tasks to employee. He is responsible for managing different tasks such as table assigning, receive payment etc.
3. **Employee**
   * Employee is responsible for completing its assigned task. The task is assigned by admin. Employee manages its area.
4. **User interaction**
   * User can see the list of menu items. Also gives suggestion about food. They can give feedbacks about services. Also send complains.
   1. **Justification:**
      * Python is best open source programming language to develop web application for small and large enterprise with portable implementation.
      * Postgresql is also best open source database engine to work.
      * Combination of both will give great web application as output.

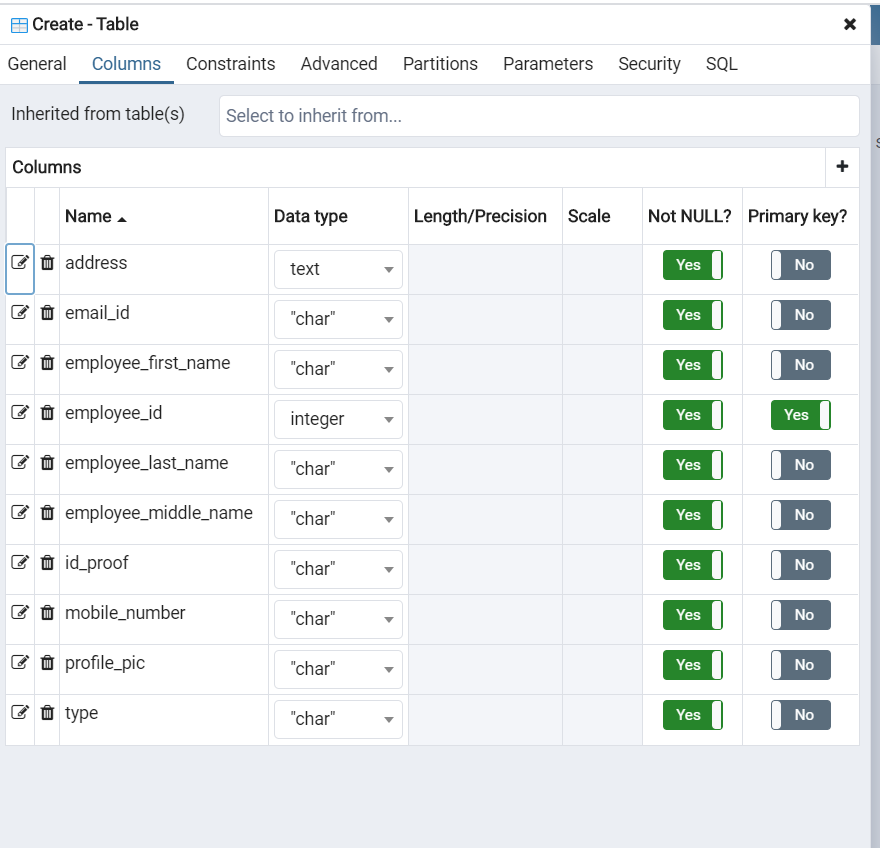
**CHAPTER 5 SYSTEM DESIGN**

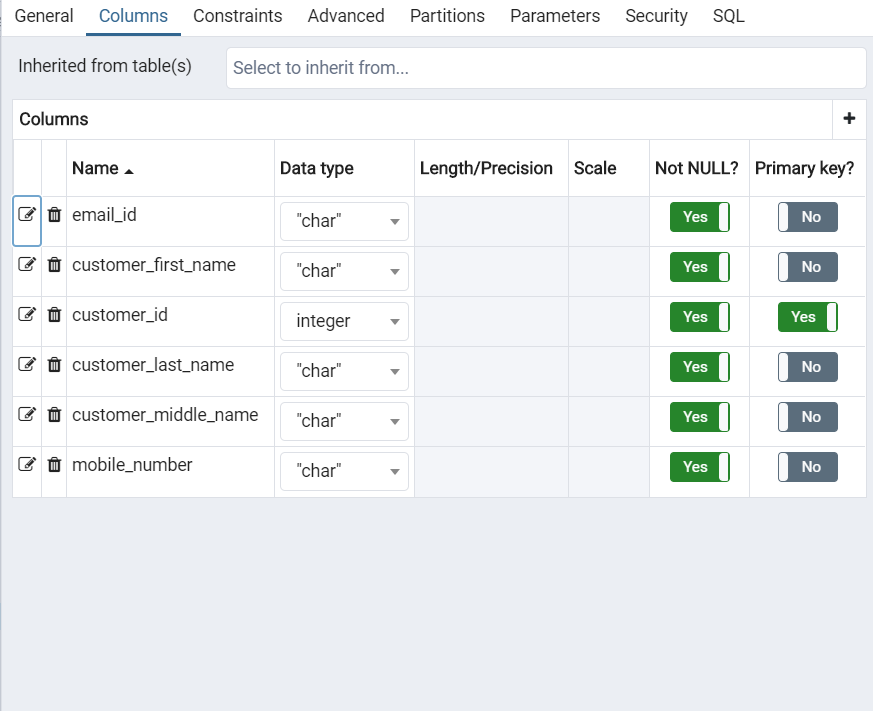
**CHAPTER 5 SYSTEM DESIGN**

Systems design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. The System Design Description report provides summary or detailed information about a system design represented by a model. Systems design is therefore the process of defining and developing systems to satisfy specified requirements of the user.

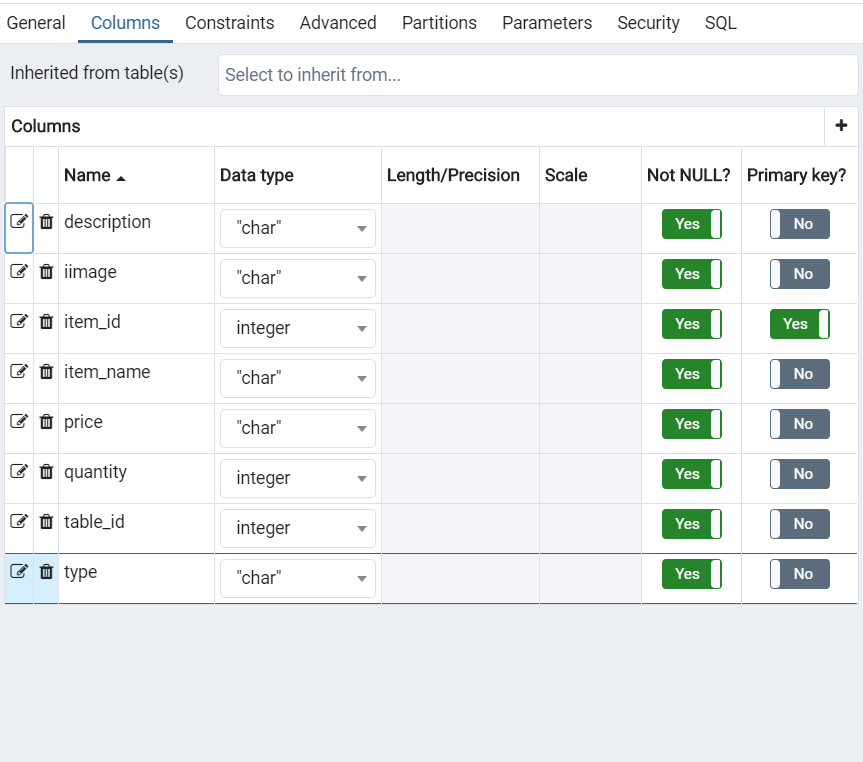
* 1. **database design/database structure design:**

Database design is the process of producing a detailed data model of a database. This logical data model contains all the needed logical and physical design choices and physical storage parameters needed to generate a design in a Data Definition Language, which can then be used to create a database. A fully attributed data model contains detailed attributes for each entity.

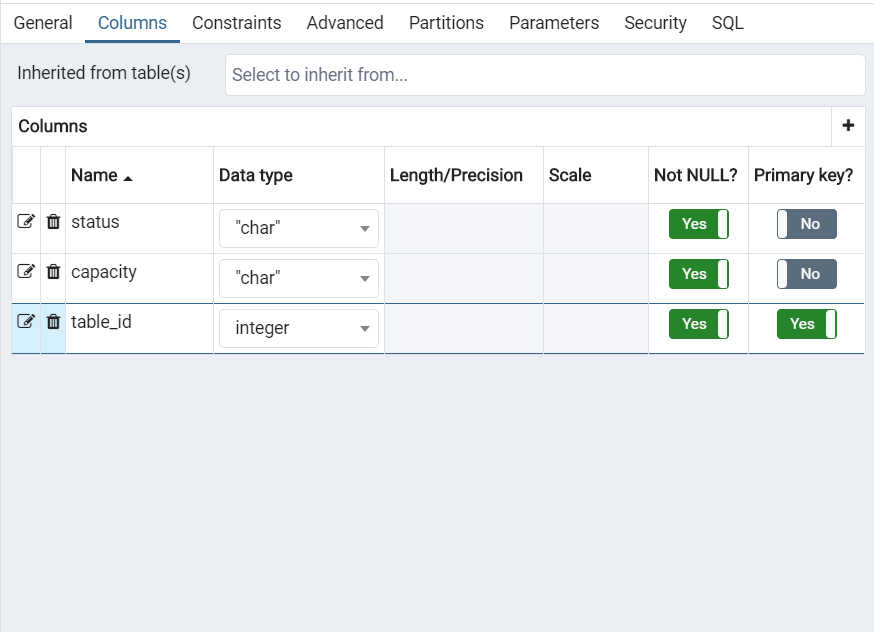




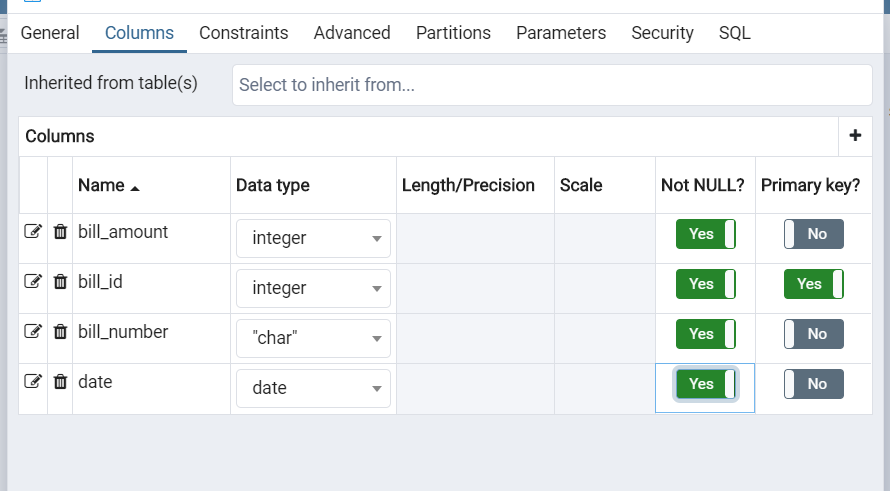
[Table 5. 2 customer\_master]



**[**Table 5.Menu Master]



[Table 5. 4 Table Master]



[Table 5. 5 Bill Master]

* 1. **System procedural design**
     1. **Design Pseudo code or algorithm for method or operation**

**Admin side**

Step 1: enter the URL to open the system Step 2: provide user name and password

Step 3: if username and password both is correct then it will login successfully. Step 4: it shows home page

Step 5: admin can able to perform many operations. Step 6: there is also one facility to change password.

**User side**

Step 1: enter the URL to open the system

Step 2: home page contains many function of the system. Step 3: this page contains events function.

Step 4: event function contains placing orders, making payments and selecting tables.

Step 5: feedback and complain a system contains feedback and complains of users.

Step 6: close the systems.

* + 1. **Flow Chart**

**Admin:**

**No**

**Admin**

**Login**

**Logout**

**Start**

**Is**

**Successful??**

**Yes**

End

**Manage Employee**

**Manage Menu**

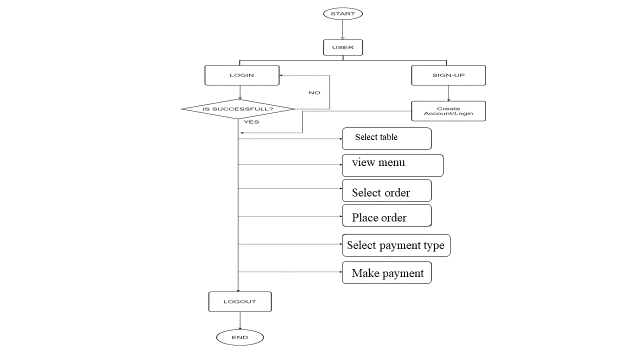
**Manage Tables**

**Manage Payment**

**Manage Time Schedule**

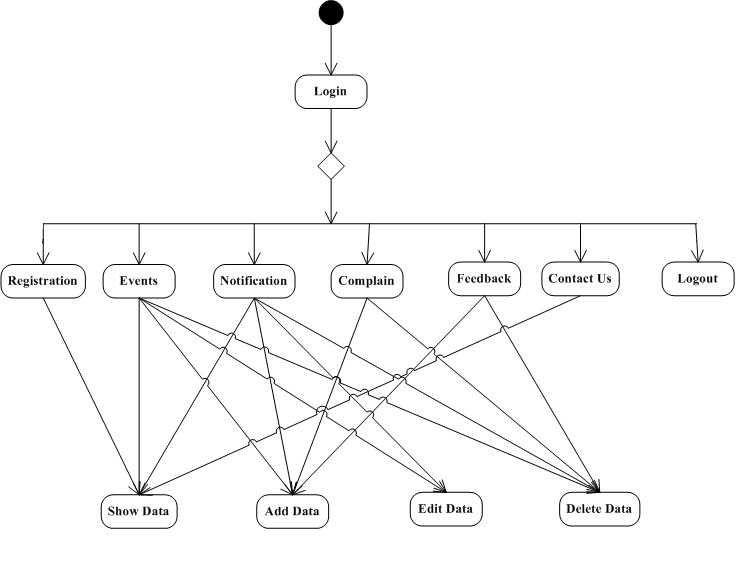
[Figure 5.1 Flow Chart -Admin]

**User:**

****

[Figure 5.3 Flow Chart –User]

* 1. **Data Modeling:**
     1. **State chart diagram**



[Figure 5.4 State Chart Diagram]

**CHAPTER 6 IMPLEMENTATION**

#### Implementation planning & details:-

* + - Implementation planning is the act of developing a tactical plan to complete a strategic initiative. Strategy is the overarching plan to move the organization, department, or project forward. Implementation is the act of putting the strategy into place utilizing resources within an organization or department. Execution is completing the tasks as part of the implementation plan to complete the strategic initiative through resources of the organized team.
    - Adding more to this, as we have defined in chapter 2 we are using web development to create an dynamic web application. We wanted to make our project in dynamic because it is the most growing technology nowadays.
    - So, our planning was very clean that first of all we will collect every detail related to our project from nearby aspects. And starting building our application on basis of our knowledge and based on the knowledge we can get from our guide.
    - And finally, when all information seems to be perfect from our perspective, we started developing the website. From basic design to hard databaseand user interaction activities..
    - We are facing many problems on the road of developing this website. But, with the help of team members and guide we are solving those problems and getting close to building application.

#### Implementation Environment (Single vs Multi-user) :-

* + - Each user is intended to be used by a different physical person. Each user has distinct application data and some unique perspective which makes them unique, as well as a user interface to explicitly switch between users. A user can run in the background when another user is active. the system manages shutting down users to conserve resources when appropriate.
    - A layout defines the structure for a user interface in your app, such as in activity. All elements in the layout are built using a hierarchy of View and View Group objects. A View usually draws something the user can see and interact with. Whereas a View Group is an invisible container that defines the layout structure for View and other ViewGroup objects..

#### Program/Modules Specification:-

* + - A module is a collection of source files and build settings that allow you to divide your project into discrete units of functionality. Our project can have one or many modules and one module may use another module as a dependency. Each module can be independently built, tested, and debugged.
    - Additional modules are often useful when creating code libraries within your own project or when you want to create different sets of code and resources for different device types, such as phones and wearables, but keep all the files scoped within the same project and share some code.
    - Within each module, files are shown in the following groups.

.

* + - **PYTHON: -** Contains the Python source code files, separated by package names, including index test code.
    - **Postgres :**- It is a open source relational data base management system emphasizing extensibility and SQL compliance.

#### Security Features:-

* + - A robust securitymodel is essential to enable a vigorous ecosystem of applications and devices built on and around the Android platform and supported by cloud services. As a result, through its entire development lifecycle.
    - We should minimize the number of permissions that your app requests. Restricting access to sensitive permissions reduces the risk of inadvertently misusing those permissions, improves user adoption, and makes your app less vulnerable for attackers.
    - Generally, you should strive to define as few permissions as possible while satisfying your security requirements.
    - By following standard coding style and guidelines, we can build a better website
    - If you follow standard coding style in android, it will be easier for you and also for others to understand your code easily.
    - Coding Style for Contributors**: -**From here you can learn the coding style that you should follow while contributing to python source code.
    - Project Guidelines**: -** here you will get to know about file naming, parameter ordering in methods, etc

.

**CHAPTER 7: TESTING**

CHAPTER 7

TESTING

#### Testing Plan:-

* + - Web application testing is a process by which application software developed for handheld website is tested for its functionality, usability and consistency. Website application testing can be an automated or manual type of testing.
    - A Test Plan is a document which describes a scope of testing, test strategy, objectives, effort, schedule and resources required. Its main purpose is to guide the whole testing process and used mostly by Project Managers or Tests Engineers.

#### Testing Strategy:-

* + - Test strategy :- A test strategy is an outline that describes the testing approach of the software development cycle. It is created to inform project managers, testers, and developers about some key issues of the testing process.
    - A Test Strategy document is a high-level document and normally developed by project manager. This document defines “Software Testing Approach” to achieve testing objectives. ... Some companies include the “Test Approach” or “Strategy” inside the Test Plan, which is fine and it is usually the case for small projects.

#### Testing Methods:-

* + - These testing methods are usually conducted in order and include : Unit testing. Integration testing. System testing
    - Unit tests often mirror the structure of the code under test. For example, a unit test project would be created for each code project in the product. The test project can be in the same solution as the production code, or it can be in a separate solution. You can have multiple unit test projects in a solution.
    - Integration testing is a component of Extreme Programming (XP), a pragmatic method of software development that takes a meticulous approach to building a product by means of continual testing and revision. There are two major ways of carrying out an integration test, called the bottom-up method and the top-down method
    - System Testing (ST) is a black box testing technique performed to evaluate the complete system the system's compliance against specified requirements. In System testing, the functionalities of the system are tested from an end-to-end perspective It

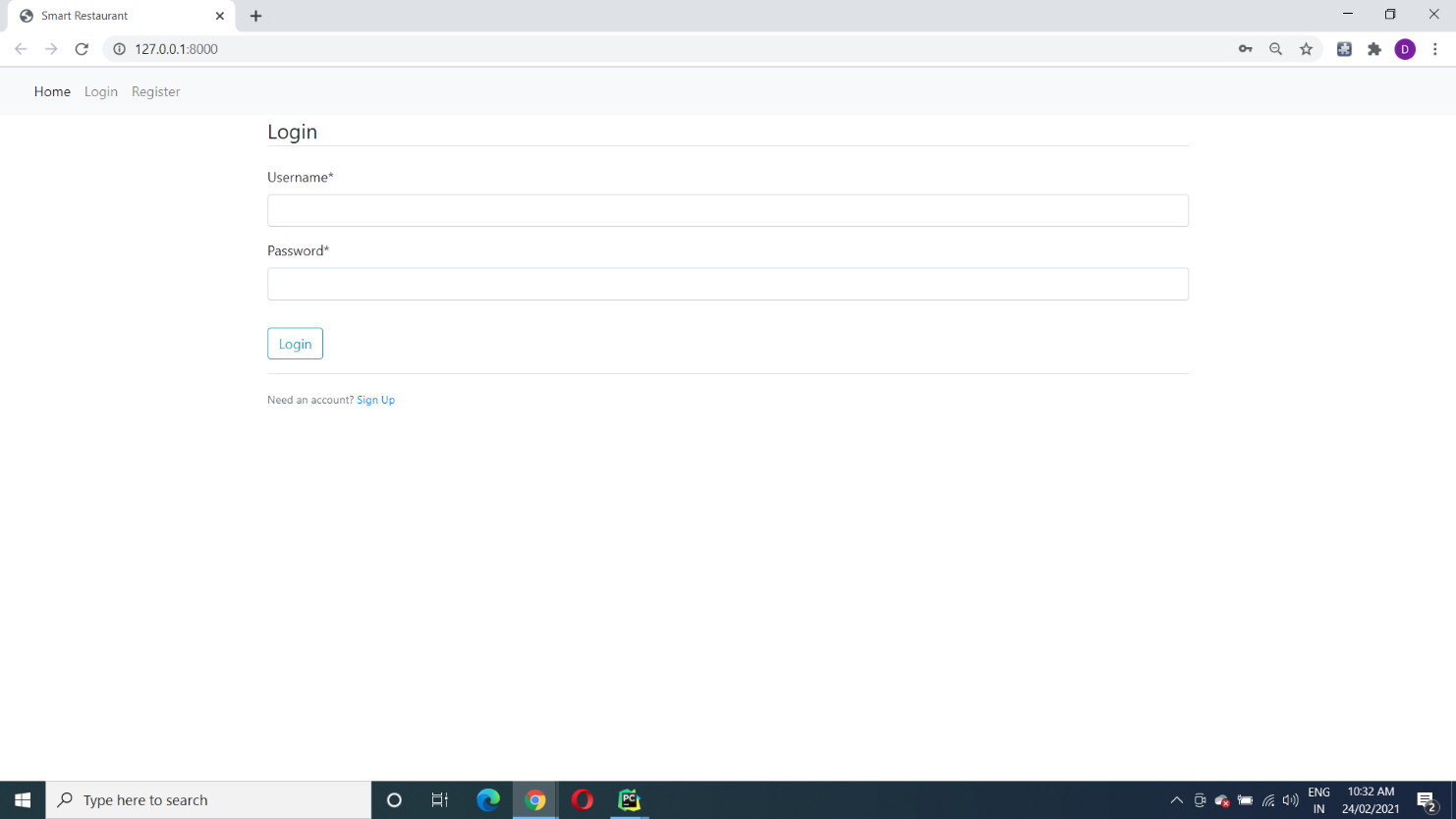
includes both functional and Non-Functional testing.

#### Test Cases(Purpose, Required Output, Expected Result):-

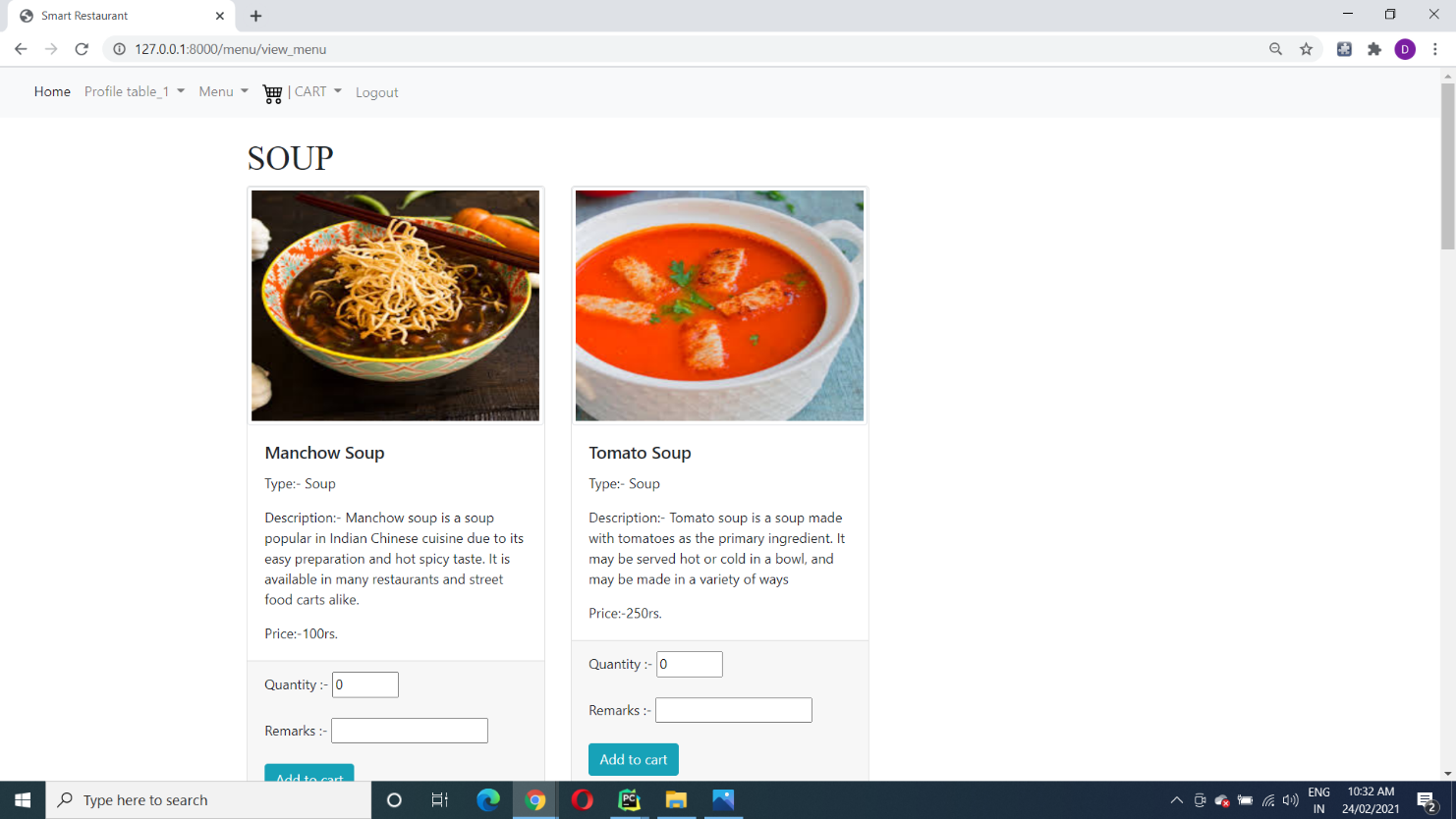
* + - A Test Case is defined as a set of actions executed to verify a particular feature or functionality of the software application. A test case is an indispensable component of the Software Testing Lifecycle that helps validate the AUT (Application Under Test).
    - A user guide or user's guide, also commonly known as a manual, is a technical communication document intended to give assistance to people using a particular system.User guides are most commonly associated with electronic goods, computer hardware and software, although they can be written for any product

1. **Screen shots and User manual**

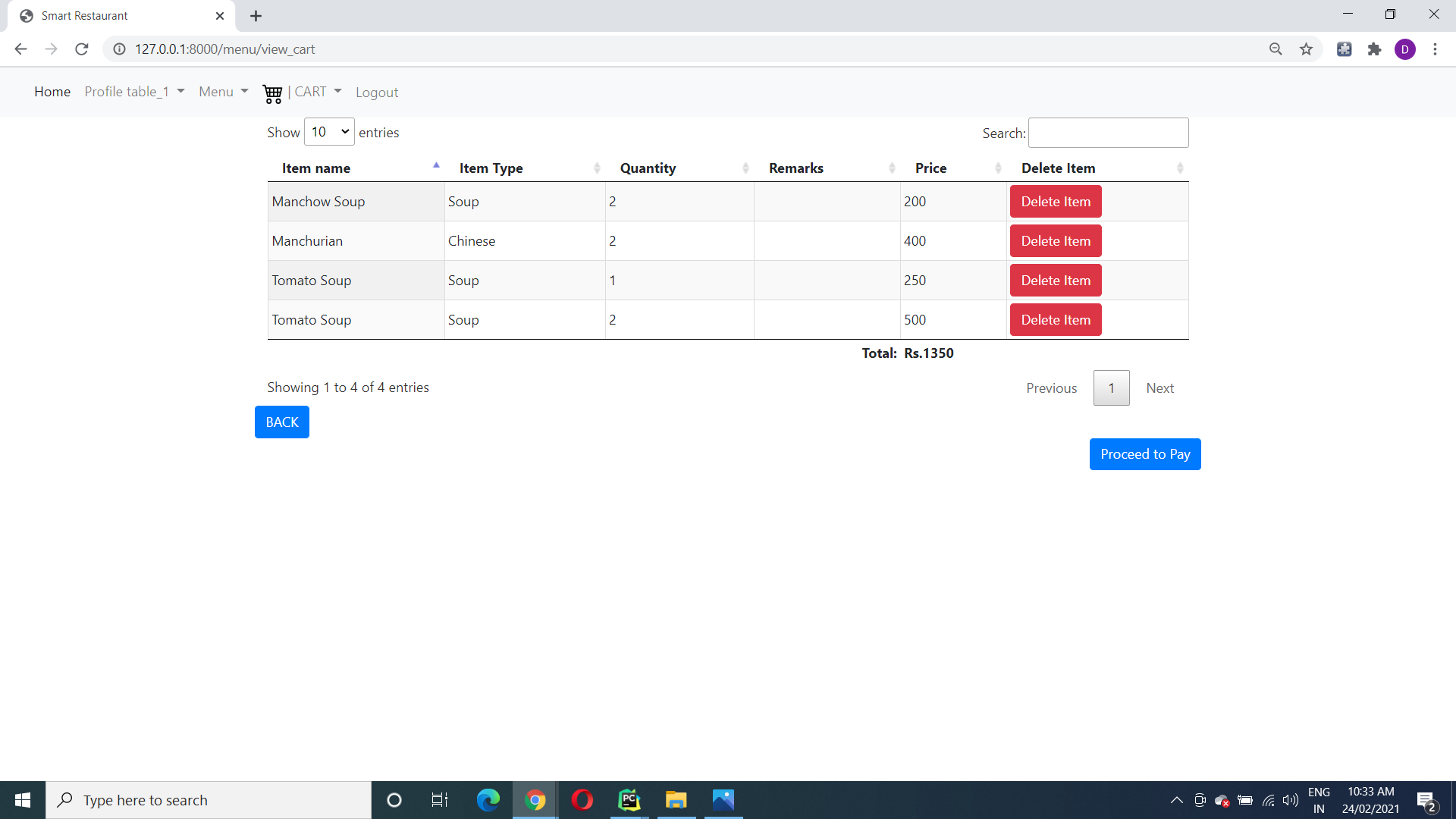
**Login Page**



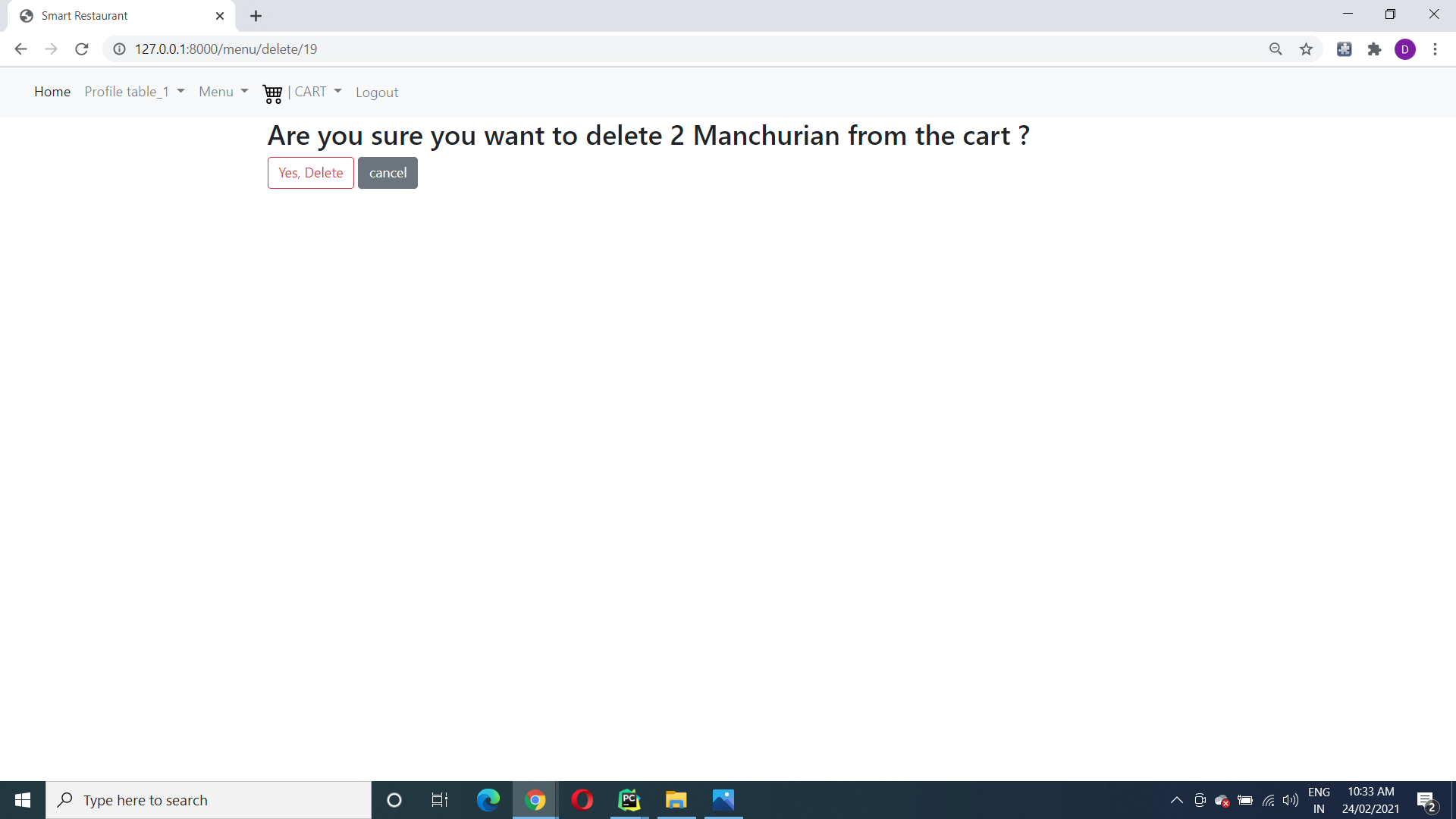
Manu Page



Cart Page



Edit Cart



Logut Page

