

**DEPT. Of Computer Science Engineering**

**SRM IST, Kattankulathur – 603 203**

**Course Code: 18CSC206J**

**Course Name: Software Engineering and Project Management**

|  |  |
| --- | --- |
| **Experiment No** | 1a |
| **Title of Experiment** | To identify the Software Project |
| **Name of the candidate** | K. BHANU PRAKASH |
| **Team Members** | DASU SARAT, YASWANTH SAI, BHANU PRAKASH |
| **Register Number** | RA1911003010467, RA1911003010465, RA1911003010471 |
| **Date of Experiment** | 22/01/2021 |

**Aim**

To Frame a project team, analyze and identify a Software project

**Team Members:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl No** | **Register No** | **Name** | **Role** |
| **1** | **RA1911003010471** | **K. BHANU PRAKASH** | **Lead** |
| **2** | **RA1911003010465** | **S. YASWANTH SAI** | **Member** |
| **3** | **RA1911003010467** | **DASU SARAT** | **Member** |

**Project Title: RESTAURANT MENU MANAGEMENT SYSTEM**

**1)INTRODUCTION**

Project Description:A restaurant’s menu is its heart and soul. It is something that every customer interacts with and makes decisions based on what looks or sounds appealing. However, a menu needs to be more than just attractive. A [smartly engineered restaurant](https://www.posist.com/restaurant-times/resources/restaurant-menu-design-maximizes-sales-increases-profits.html) menu is high on profits and also well-designed to tempt customers to order more. This is where the role of restaurant menu management software comes into the picture.

You can book a table and cancel at any time before a few hours (As per the restaurant)

Before we move ahead and apprehend the features you should be looking out in a restaurant menu management software, we need to understand what this software is all about.

A restaurant menu management software is a feature of the [POS software](https://www.posist.com/) that is used to create and manage the restaurant menu. It is integrated with the [billing](https://www.posist.com/features/restaurant-billing-software/) and the [inventory management](https://www.posist.com/features/restaurant-inventory-software/) module and helps you curate and manage the restaurant menu.

* 1. SCOPE OF THIS APPLICATION:

1. Any restaurant can use this menu management software.
2. It is user friendly software, it can be use by an age group people.
   1. Focus of the project: The system is completed under the guidance of the theory and methods of management information systems. This paper first discusses the background, purpose and significance of the topics. Then describes the development platform and the advantages of each, followed by more devoted system requirements analysis, design, implementation, and the implementation of the tasks, techniques and tools. End system to complete the information input, output, data modification, query and statistics, as well as print statements, make operation simple and quick.

In this project, we try to build up sound software which can operate any challenging situation in the modern time. The decisions are more accurate, relevant and timely the information stored or process is more effective.

* 1. **Features of the project:**
* This is very easy to use for each user.
* Increase Efficiencies and Reduce Costs
* Secure All Data
* Easy Account Maintenance
* Transaction History
* It is user friendly software

**Result:** Thus, the project team formed and the project is described

**1B:**

AIM: To create a business case and arrive at a Problem Statement for e-Menu.

# BUSINESS CASE:

# EXECUTIVE SUMMARY: It is a software which can be used for different restaurants, in which customers can select their favorite dishes from the e-menu, at the same time the customers can know the bill for their orders. Also, the customers can book a table and cancel whenever they want few hours before.

1. Strategic Business Context:

2.1) Business need:

Our main agenda is to develop a software that can be used in all restaurants for showing their menu so customers can select different types of dishes from the menu and book a table for the customers.

2.2) Business outcomes:

We are expecting profits from the 2nd financial year from the date of release of our e-Menu software to different restaurants.

We will develop our software by following latest trends in software industry, and we will work hard for increasing the efficiency of the software.

3) Detailed Business problem:

# 3.1) Problem/Opportunity Statement:

Nowadays the restaurants business going so well, and the restaurants owners are thinking differently so that in which way they can further develop their restaurant business. In that 1st way is showing menu to the customers in a different way like customers easily select their menu at last automatically the Bill will generate and display.

So, we are mainly focusing on the e-menu so that customers can select their favorite dishes very easily by the software and they can know how much bill they have to pay at last.

The 2nd way is booking a table for the customers and letting them know if the tables are available.

# 3.2. High Level Requirements:

The software estimation cost is about $10,000 and team of 3 members goes around $30,000 and total estimated cost of the project is $40,000.

1. User should be able to navigate the system without any difficulty.
2. System supports native language of the country and other commonly spoken languages.
3. User should be able to book table and place order according to his choices.
4. The system should be able to take any amount of order and display it when finished.
5. Bringing regular updates to the software for smooth functioning.

# 3.3) Assumptions:

|  |  |
| --- | --- |
| **S.no** | **Assumptions** |
| **1** | **All users are eligible to use the service in terms of the device and network availability** |
| **2** | **The production cost will be taken into count and reduced as much as possible.** |
| **3** | **All the necessary resources will be notified and accessible to the team.** |

**3.4) Constraints:**

|  |  |  |
| --- | --- | --- |
| **S.no** | **Category** | **Constraints** |
| **1** | **Time Constraints** | **Finishing the project within the stipulated time; 3 months** |
| **2** | **Budget Constraints** | **According to the situations budget constraint is important to be notified and cannot be limited for a project under development.** |

## 3.5. Dependencies:

**Hardware dependencies:** Databases to maintain user data, servers.

**Software dependencies:**

**Language used: C++**

**3.6) Stakeholder Analysis:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Designation** | **Role in Project** |
| Mr. Ramraj | Corporate Head for Sales & Marketing | Executive Sponsor |
| K. BHANU PRAKASH | Chief Information Officer or Regional Head of Sales & Marketing | Project Sponsor |
| S. YASHWANTH SAI | Finance Head | Cost Approver |
| DASU SARAT CHANDRA SRIDHAR | Department Head(s) | Scope / Requirement Approver |
|  |  |  |

Result: Thus, the business case was prepared and the problem statement was arrived.

**EXP 2:**

**Aim**

To identify the appropriate Process Model for the project and prepare Stakeholder and User Description.

# 1)Executive Summary

Incremental model methodology is used to understand proper requirements for this project. And for analysis and identification of stakeholder for this project we took observation Elicitation technique.

**Observation** method was used to gather requirements/ perform stakeholder analysis for the project. In this method the user behaviour is monitored directly or user reviews of existing services are studied to gather requirements. This would help to understand user requirements in a better manner and deliver more user-friendly service.

**Most of the users felt that the following features would help to make software user friendly:**

1. UI should be user friendly.
2. Bill generating should be convenient.
3. Menu should be customizable.
4. Booking of table should be first come first serve basis.

# Selection of Methodology

Incremental Model is a process of software development where requirements are broken down into multiple standalone modules of software development cycle. Incremental development is done in steps from analysis design, implementation, testing/verification, maintenance.



The product is decomposed into a number of components, each of which are designed and built separately (termed as builds). Each component is delivered to the client when it is complete. This allows partial utilization of product and avoids a long development time. It also creates a large initial capital outlay with the subsequent long wait avoided. This model of development also helps ease the traumatic effect of introducing completely new system all at once.

There are some problems with this model. One is that each new build must be integrated with previous builds and any existing systems. The task of decomposing product into builds not trivial either. If there are too few builds and each build degenerates this turns into Build-And-Fix model. However, if there are too many builds then there is little added utility from each build.

* System development is broken down into many mini development projects
* Partial systems are successively built to produce a final total system
* Highest priority requirement is tackled first
* Once the requirement is developed, requirement for that increment are frozen
* Generates working software quickly and early during the software life cycle.
* More flexible – less costly to change scope and requirements.
* Easier to test and debug during a smaller iteration.
* Easier to manage risk because risky pieces are identified and handled during its iteration.
* Each iteration is an easily managed milestone.

## Roles and Methods

We used observation method for identification of stakeholders.

1. Project Manager
2. Customers
3. Tester
4. Project Team members.
5. Executives.
6. Resource managers.
7. Sponsor
8. Public
9. Evaluator.

# Stakeholder Management

## Identification of Stakeholders:

**Project Manager**: makes sure the project is ready and delivered on time, manages the whole team and makes sure smooth execution takes place at each phase.

**Tester:** testing and debugging the code for optimum performance of the application

**Customers**: Gathering data on the customer's needs, culture, business pains and then documenting it all before the project execution starts will always help when conflicts arise.

**Project team members**: Project team members key manages team expectations is constant communication. Schedule informal one-on-one meetings with each member to get insight into the group's temperature. Let them know you can be approached any time.

**Executives**: The key here is presenting them with a comprehensive plan with milestones. Then continually update them on project risks based on their information needs.

**Resource managers**: The key here is establishing a good relationship. If you are in good standing with a resource manager, your requests for equipment or manpower will never be roadblocked.

**Sponsor:** Sponsors the project to seek reputation.

**Public**: Public gives the feedback of the product by giving rating and reviews.

**Evaluator:** Evaluates the final output of this project.

**Project stakeholders** usually include the **project manager**, the customer, team members within the performing organization, and the **project** sponsor.

## Interest and Influence matrix

|  |  |
| --- | --- |
| **Interest** | **Influence** |
| High | High |
| Low | Low |
| Low | High |
| High | Low |

|  |  |
| --- | --- |
| **Low Interest, High Influence**  Keep them satisfied as they can be ‘defenders’  Help them engage more | **High Interest, High Influence**  Engage them closely as they are key ‘drivers’ |
| **Low Interest, Low Influence**  Low Priority as they are ‘spectators’ | **High Interest, Low Influence**  Keep them informed as they can be ‘blockers’ |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Stakeholder Name** | **Activity / Area / Phase** | **Interest** | **Influence** | **Priority (High 1/ Medium 2/Low 3)** |
| Project manager | Manages the team and ensures smooth execution of the phases involved in the project. | High | High | 1 |
| Developer | Works on the implementation of the project, writing front end code and deploying API. | High | High | 1 |
| Tester | Tests, debugs and validates the code required for the project. | high | high | 1 |
| Customer | The end-users who buy the product | high | low | 1 |
| Investor | Invests on the project | high | high | 1 |
| Sponsor | Sponsors our project | low | high | 2 |
| Public | Gives feedback | low | low | 3 |
| Evaluator | Evaluating the project | low | high | 2 |

## Communication Plan for Stakeholders:

1. Feedback from the institution to be taken twice in a month, through in-person meetings, voice calls, forms, mails.
2. Online chat rooms and video calls for the team with Project owner.

3)Walk through of progress to High Interest and High Influence stakeholders in meeting.

4)Bi-Weekly report for High Interest and Medium Influence.

**Result**: Thus, the Project Methodology was identified and stakeholders were described.

**EXP 3:**

**Aim:**

To Identify and document the Requirements of a software system designed for Hotel Management System.

**Table of Contents**

[***1.***](#_30j0zll) ***Executive Summary 2***

[***2.***](#_1fob9te) ***Project Scope 2***

[**2.1.**](#_3znysh7) **In Scope 2**

[**2.2.**](#_2et92p0) **Out of Scope 2**

[***3.***](#_tyjcwt) ***Epics [Major Functions] 2***

[***4.***](#_3dy6vkm) ***Requirements 2***

[**4.1.**](#_1t3h5sf) **Functional Requirements 2**

[**4.2.**](#_4d34og8) **Non-Functional Requirements 3**

[**4.3.**](#_2s8eyo1) **Infrastructure Requirements 3**

[**4.4.**](#_17dp8vu) **Requirement definition in Agile [Optional … Use according to methodology chosen by student] 4**

[***Reference 4***](#_3rdcrjn)

# Executive Summary

A restaurant menu management software is a feature of the [software](https://www.posist.com/) that is used to create and manage the restaurant menu. It is integrated with the [billing](https://www.posist.com/features/restaurant-billing-software/) and the [inventory management](https://www.posist.com/features/restaurant-inventory-software/) module and helps you curate and manage the restaurant menu easily. Customer can book a table and cancel at any time before a few hours of allocate time.

# Project Scope

This software is aimed at facilitating the prompt, accurate and easy management of a menu and table booking in restaurants. It will be very helpful to the owners to run the restaurants easily and profitability and on the other hand and to the customers as they would be able to get good and fast service.

|  |  |  |
| --- | --- | --- |
| **S. No** | **Activities in Scope** | **Activities Out of Scope** |
| 1. | To deliver the software, so that customers coming to restaurants could find the menu user friendly. | The scope does not include the internal operations of the restaurant. |
| 2. | Booking tables by customers by their own. | Taste and cost of the food items |
| 3. | UI of the software | Behavior of the staff |

## In Scope

1. To deliver the software, so that customers coming to restaurants could find the menu user friendly.

2)Booking tables by customers by their own.

3)UI of the software

## Out of Scope

The activities which are not going to be delivered are said to be out of scope.

1. The scope does not include the internal operations of the restaurant.
2. Taste and cost of the food items
3. Behavior of the staff

# Epics [Major Functions]

|  |  |
| --- | --- |
| **Epic (#)** | **Epic Description** |
| E1 | Menu management |
| E2 | Table booking |

# Requirements

## Functional Requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Requirement (#)** | **Requirement Specification** | **Department** | **Name of Business User** | **Status** |
| E1FR1 | Ability to alter the price of food items |  |  |  |
| E1FR2 | Updating the items in the menu |  |  |  |
| E1FR3 | **Table Reservation:**  The main function of this application is online table reservation user can reserve any table through this application any  time and from anywhere |  |  |  |
| E1FR4 | Bill will be generated automatically |  |  |  |
| E1FR5 | Ability add/remove offers. |  |  |  |
| E1FR6 | Ability to print customer receipt on order completion |  |  |  |
| E1FR7 | Ability to add sub order to the existing order. |  |  |  |

## Non-Functional Requirements

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requirement (#)** | **Category of NFR** | **Requirement Specification** | **Department** | **Name of Business User** | **Status** |
| NFR1 | Performance | Minimum clicks from the beginning to the end of an order. |  |  |  |
| NFR2 | Performance | Ability to calculate order price and discounted price. |  |  |  |
|  | Availability | Software should be available for 24x7. |  |  |  |
|  | Scalability | Booking service should accept upto 30 requests per minute. |  |  |  |
|  | Confidentiality | Booking of the table and details of the customer must be encrypted and stored securely |  |  |  |
| E1NFR2 | Usability | E-menu should be simple for customers to understand and quick to use.  The interface of the software must be user-friendly. |  |  |  |
|  |  |  |  |  |  |
|  | Flexibility | Should be able to access all types of food items in the menu. |  |  |  |
| E1NFR1 | Extensibility | Ordering Service should be extensible within the state. |  |  |  |
|  | Reliability | Restaurant should rely on us without any problems. |  |  |  |
|  | Rapidity | Integrate New Payment method. |  |  |  |

## Infrastructure Requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Requirement (#)** | **Requirement Specification** | **Department** | **Name of Business User / Project Team Member** | **Status** |
| IR1 | Tablet |  |  |  |
| IR2 | Laptop and computer |  |  |  |
|  |  |  |  |  |

## Requirement definition in Agile [Optional … Use according to methodology chosen by student]

|  |  |  |
| --- | --- | --- |
| **User Story** | **Acceptance Criteria** | **Size of User Story** |
| As a customer I can book any table in advance and browse different type of food items available in the menu and get bill at a time of ordering the food itself. | Look at all the food items in the menu and order my preferred food item and Book table of my wish (seating). | Medium |

# Reference

1. <https://www.pmi.org/>
2. <https://www.atlassian.com/agile/project-management/user-stories>

**Result:**

Thus, the requirements are identified, collected and documented.

**EXP 4:**

**Aim**

To Prepare Project Plan based on scope, Find Job roles and responsibilities, Calculate Project effort based on resources.

**Table of Contents**

[***1.***](#_heading=h.gjdgxs) ***Executive Summary 2***

[***2.***](#_heading=h.30j0zll) ***Project Management Plan 2***

[***3.***](#_heading=h.3znysh7) ***Estimation 3***

[***3.1.***](#_heading=h.2et92p0) ***Effort and Cost Estimation 3***

[***3.2.***](#_heading=h.tyjcwt) ***Infrastructure/Resource Cost [CapEx] 3***

[***4.***](#_heading=h.3dy6vkm) ***Maintenance and Support Cost [OpEx] 4***

[***5.***](#_heading=h.1t3h5sf) ***Project Team Formation 4***

[***5.1.***](#_heading=h.4d34og8) ***Identification Team members 4***

[***5.2.***](#_heading=h.2s8eyo1) ***Responsibility Assignment Matrix 4***

[***Reference 5***](#_heading=h.17dp8vu)

# Executive Summary

It is a software which can be used for different restaurants, in which customers can select their favourite dishes from the e-menu, at the same time the customers can know the bill for their orders. Also, the customers can book a table and cancel whenever they want few hours before. We will be using COCOMO to estimate the cost. The

project management plan will be drafted keeping in mind various aspects like scope

management, schedule management, cost management etc.

# Project Management Plan

This helps to keep the project on track and outlines the scope , schedule, cost, resource,risk

|  |  |
| --- | --- |
| **Focus Area** | **Details** |
| Integration Management | Project Head (1) > Team Manager (2) > Developers (3) > Tester (4)  (1) Controls activities and makes sure standards are met  (2) Supervises and manages the team  (3) Creates the software using technical skills  (4) Analyse the project for flaws |
| Scope Management | This project aims to create a software for the Restaurants to book a table and for selecting different types of food items from E-menu |
| Schedule Management | Software interface development  Testing and improvising |
| Cost Management | Using cocomo model  Effort estimation is 5 per month |
| Quality Management | Quality Assurance: Quality assurance will be managed including governance, roles and responsibilities, tools and techniques and reporting  Quality Control: Quality can be maintained by fixing the bugs and updating the application. |
| Resource Management | Estimated by the COCOMO model.  People: Team, Backend developers, UI/UX designers, testers.  Finance: Budget- ₹ 200000  Physical: Laptops |
| Stakeholder | Stakeholder analysis is done by incremental model methodology.  Stakeholders include:1) Project Manager  2)Customers  3)Tester  4)Project Team members.  5)Executives.  6)Resource managers.  7)Sponsor  8)Public  9)Evaluator. |
| Communication Management | Weekly scheduled meetings between the team leader and the  developers. The project head and other stakeholders have to be  updated on the progress of the project. Communication should be  open and two-way |
| Risk Management | The privacy of the users and the information stored in the database  has to be secured to prevent a security breach.  The inability to complete the project on time and within budget is also  a major risk. |

# Estimation

# Effort and Cost Estimation

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **WBS** | **Activity** | **Activity Description** | **Sub-Task** | **Sub-Task Description** | **Effort (in hours)** | **Cost in INR** |
| E1FR1 | E1R1A1 | Designing the software | E1R1A1T1 | Writing the code for the software | 36 | 12000 |
| E1R1A1T2 | Designing the UI | 20 | 6000 |
| E1R1A1T3 | Testing the software | 8 | 2000 |
|  |  | Developing the  Database | E1R2A1T1 | Designing the MS SQL  Server | 40 | 16000 |
| E1R2A1T2 | Connecting the software interface  to the database | 30 | 12000 |
|  |  |  | E1R2A1T3 | Testing the database | 8 | 2000 |

|  |  |
| --- | --- |
| **Effort (hr)** | **Cost (INR)** |
| 1 | 400 |

**Using COCOMO:**

No. of lines of code= 2KLOC

For organic project, the limits are 2-50KLOC, hence this falls under the organic category

This is an organic project because:

● The size of the team is small

● The program to be developed is well understood

● The lines of code fall in the 2-50 LOC bracket

Effort (E)=ab (KLOC)bb

=2.4(2)1.05

=4.969PM

Hence, the effort is 4.969 Person Months

Deployment Time(D)= cb(E)db

=2.5\*(4.969)0.38

=4.597 months

# Infrastructure/Resource Cost [CapEx]

|  |  |  |  |
| --- | --- | --- | --- |
| **Infrastructure Requirement** | **Qty** | **Cost per qty** | **Cost per item** |
| Office rent | 1 | 100000 | 100000 |
| Computers | 3 | 60000 | 180000 |
| Transport | 3 | 20000 | 60000 |

# Maintenance and Support Cost [OpEx]

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Category** | **Details** | **Qty** | **Cost per qty per annum** | **Cost per item** |
| People | System and DB Manager  Customer Support | 2 | 3,000,000 | 6,000,000 |
| License | Operating System  Database  Middleware  IDE | 3 | 20000 | 60000 |
| Infrastructures | Server, Storage and  Networking | 2 | 60000 | 120000 |

# Project Team Formation

# Identification Team members

|  |  |  |
| --- | --- | --- |
| **Name** | **Role** | **Responsibilities** |
| Yaswanth sai | Key Business User (Product Owner) | Provide clear business and user requirements |
| Bhanu prakash | Project Manager | Manage the project |
| Dasu sarat chandra sridhar | Business Analyst | Discuss and Document Requirements |
| Dr.Ramraj | Technical Lead | Design the end-to-end architecture |
| Bhanu prakash | UX Designer | Design the user experience |
| Yaswanth sai | Frontend Developer | Develop user interface |
| Dasu sarat chandra sridhar | Backend Developer | Design, Develop and Unit Test Services/API/DB |
| Dr.Ramraj | Tester | Define Test Cases and Perform Testing |

5.2 Responsibility Assignment Matrix

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **RACI Matrix** | **Team Members** | | | |
| **Activity** | **Name (BA)** | **Name (Developer)** | **Name (Project Manager)** | **Key Business User** |
| User Requirement Documentation | A | C/I | I | R |
|  | Dasu sarat chandra sridhar | Yaswanth sai | Bhanu prakash | Restaurants that purchase  the software |

|  |  |
| --- | --- |
| A | Accountable |
| R | Responsible |
| C | Consult |
| I | Inform |

**Result:**

Thus, the Project Plan was documented successfully

**EXP 5:**

**Aim**

To Prepare the Work, Breakdown Structure based on timelines, Risk Identification and Plan

**Table of Contents**

[**1. Executive Summary** **4**](#_Toc15521)

[**2. WBS With Project Schedule** **4**](#_Toc15522)

[**3. Risk Identification** **7**](#_Toc15523)

[**3.1. List (Describe) Register** **7**](#_Toc15524)

[**3.2. Managing Risk** **9**](#_Toc15525)

**Requirements:**

# Executive Summary

Documentations, increments and user feed backs are the major project milestones. The project is estimated to be completed by 5-6 months. SWOT analysis was performed to identify potential risks. Risks were thoroughly analysed and prioritized. Appropriate management strategies were formulated.

# WBS With Project Schedule

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Module  (#) | Activity  (#) | Sub-Task (#) | Assignee(s) | Planned  Start Date | Planned  End  Date | Actual  Start Date | Actual  End  Date | Status |
| Login credentials | UI/UX  design  Database design Testing | Integrating with Database | Yaswanth Sai  Bhanu Prakash  Sarat  Dasu | 30/01  /21 | 14/02  /21 | 30/01  /21 | 14/02  /21 | Closed |
| Authentication | Database design Testing | Integrating  with database | Bhanu Prakash  Yaswanth  Sai | 14/02  /21 | 21/02  /21 | 14/02  /21 | 21/02  /21 | Closed |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| User | UI/UX  design  Database design Testing | Integrating with database |  | 30/01  /2021 | 14/02  /2021 | 30/01  /2021 | 14/02  /2021 | Closed |
| Chat | UI/UX  design  Connect ion to other users  Testing |  | Yaswanth Sai  Bhanu Prakash  Sarat  Dasu | 22/02  /21 | 22/03  /21 | 22/02  /21 | 22/03  /21 | Ongoing |
| Payment | UI/UX  design  Database design Testing | Establishing safe gateway for  payment  Connection to database | Yaswanth Sai  Bhanu Prakash  Sarat  Dasu | 23/03/2  021 | 24/04/2  021 | 23/03/2  021 | 24/04/2  021 | Pending |

**Activity**

**/**

**Time in**

**weeks**

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

Databas

e

Design

UI/UX

Design

Integrati

ng UI

with

Databse

Softwar

e testing

Software

deployment

# Risk Identification

STRENGTHS:

* User-friendly UI.
* Customised software.
* Organizes all information in a single platform.
* Reduces paper which saves the tress.

WEAKNESS:

* Frequent changes in UI.
* Bugs can be found rarely.
* Should have proper net connection.
* Costs a lot.

OPPURTUNITIES

* This software has good market because many restaurants are using.
* We will deliver more user-friendly software.
* Would gain interest since restaurants get a one-stop solution.

THREATS

* Security breaches.
* Miscommunication between stakeholders or implementing changes frequently can add to the cost and cause delays.
* Breakage of tablets due to mishandling by customers.
* Once the customer places the order they have the choice to cancel the order within 20 seconds or else they cannot.

## List (Describe) Register

|  |  |  |
| --- | --- | --- |
| **Risk ID (#)** | **Risk Description** | **Impact Description** |
| R01 | Server Crash-down | Medium |
| R02 | Restaurant backing down to give personal information | High |
| R03 | Developers Personal issue | Low |
| R04 | Design not liked by the consumer | Medium |
| R05 | Competition Making a same app | High |

## Managing Risk

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Risk ID (#)** | **Status [Open / Closed]** | **Risk Appetite [ Accept/ Mitigate/ Transfer/Avoid]** | **Action** | **Action Owner** | **Target Date** | **Remarks** |
| R01 | Open | Accept | Security Breaches | Yaswanth Sai | within 2-3 days | - |
| RO2 | Open | Mitigate | Application  loading | - | - | Having decent internet connect-  ion. |
| R03 | Open | Accept | Insecure data storage | K. Bhanu Prakash |  | - |
| R04 | Open | Mitigate | End user  engagement | Sarat Dasu | about a week | - |

**Result:**

Thus, the WBS and Risk Plan was documented successfully.

**EXP 6:**

**Aim**

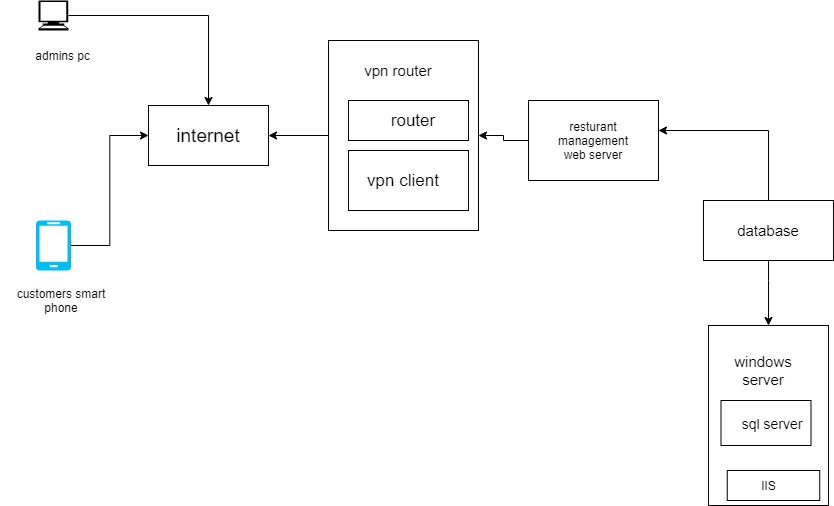
To prepare architecture and design of the system

**Software Used:**

**Star UML**

**Architecture Diagram with description:**

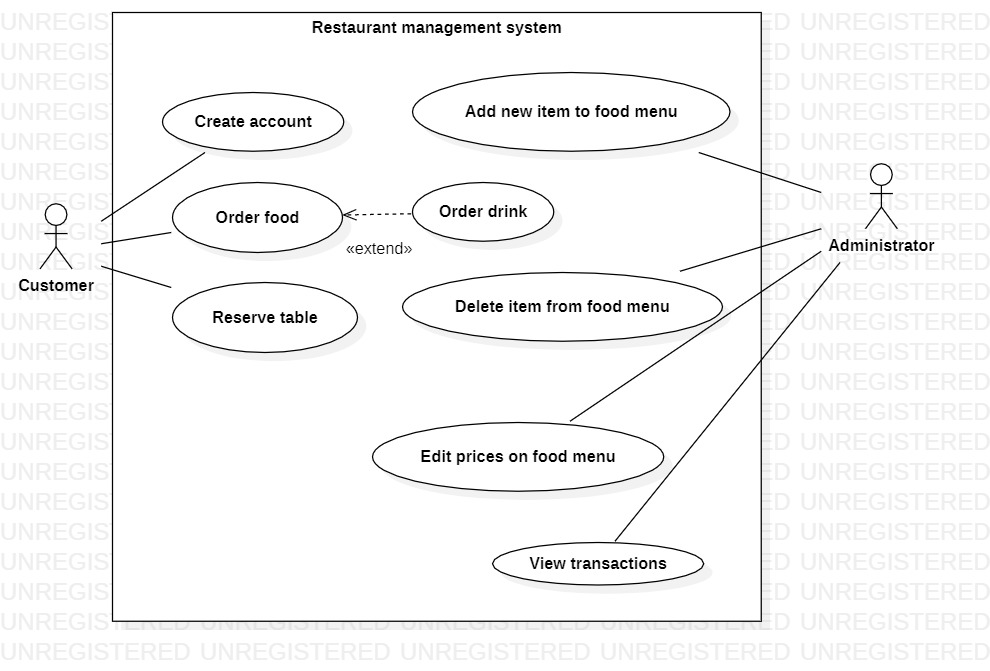
An architectural diagram is a diagram of a system that is used to abstract the overall outline of the software system and the relationships, constraints, and boundaries between components. It is an important tool as it provides an overall view of the physical deployment of the software system and its evolution roadmap

****

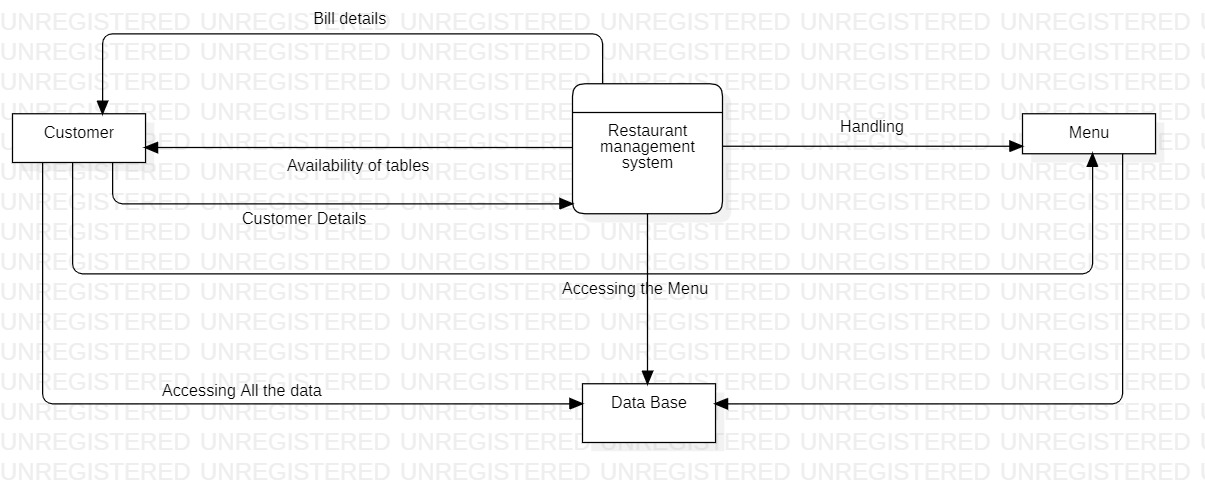
**Use Case Diagram with Description:** This Use Case Diagram is a graphic depiction of the interactions among the elements of

Restaurant Management system. It represents the methodology used in system analysis to identify, clarify, and organize system requirements of application.

The main actors of Restaurant Management system in this Use Case Diagram are Customer and Administrator as below mentioned.

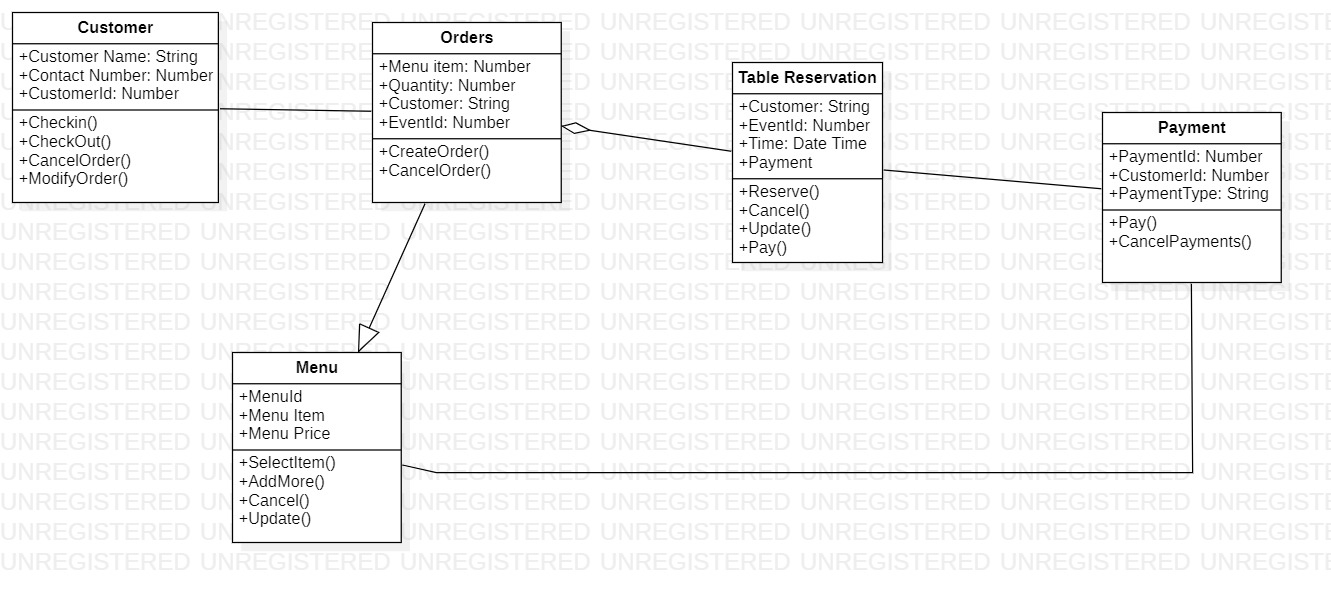


**DFD Diagram (process) With Description: A data-flow diagram is a way of representing a flow of data through a process or a system. The DFD also provides information about the outputs and inputs of each entity and the process itself.**

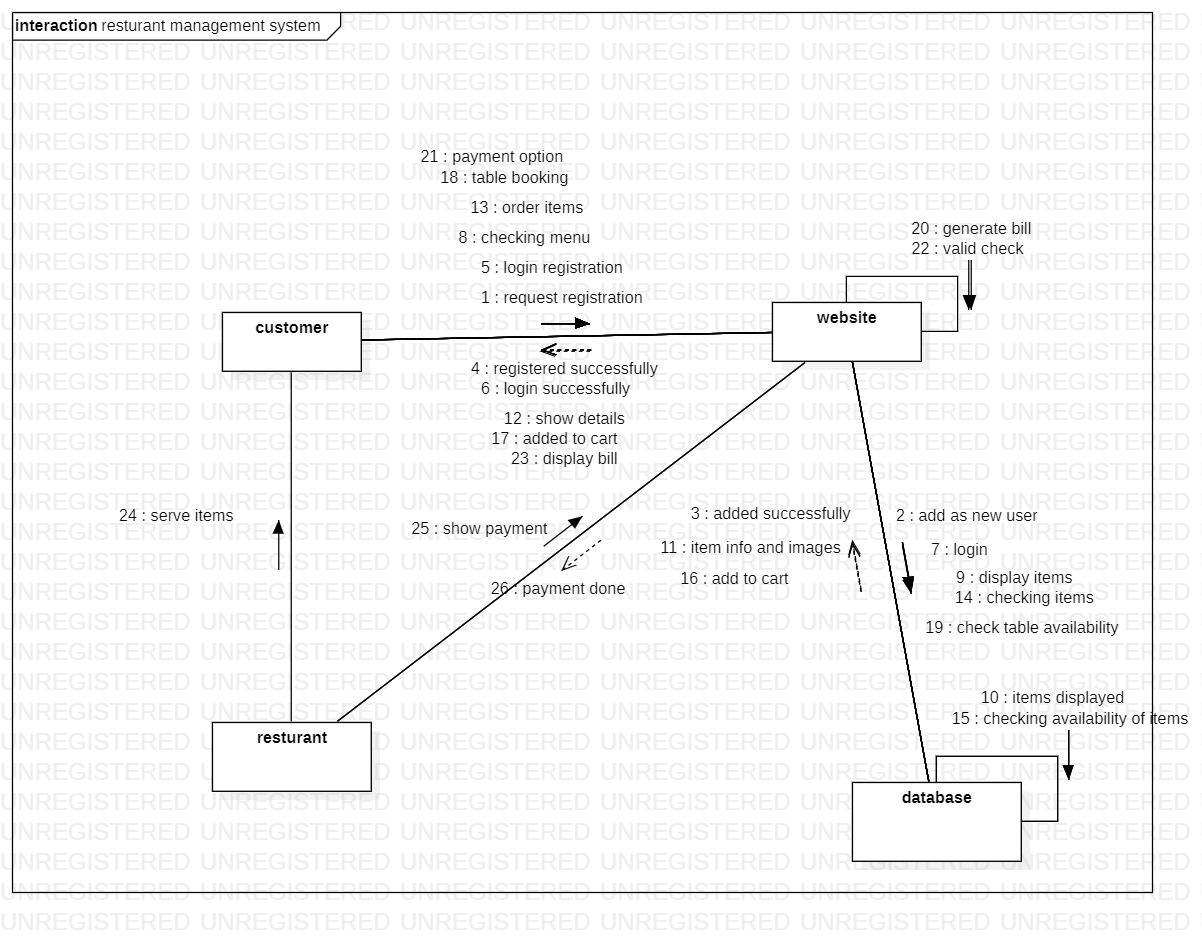
****

**Class Diagram (Applied for OOPS based Project):** A class diagram in the unified modelling 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.

Class diagram describes the attributes and operations of a class and also the constraints imposed on the system



**Collaboration Diagram (Applied for OOPS based Project):** The collaboration diagram is used to show the relationship between the objects in a system. Both the sequence and the collaboration diagrams represent the same information but differently. Instead of showing the flow of messages, it depicts the architecture of the object residing in the system as it is based on object-oriented programming. It is used to portray the object's architecture in the system.



Result:

Thus, architecture and design of the system was documented successfully.

**EXP 7:**

**Aim**

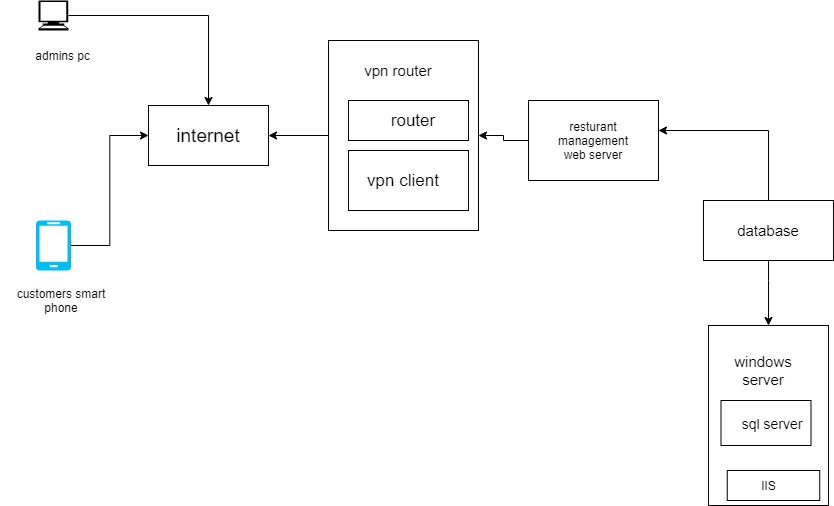
To Design State, Collaboration, Deployment Diagram, Sample Frontend Design (UI/UX) for the project.

**Software Used**

**Star UML,** Rational Rose, Etc…

**Architecture Diagram with description**

An architectural diagram is a diagram of a system that is used to abstract the overall outline of the software system and the relationships, constraints, and boundaries between components. It is an important tool as it provides an overall view of the physical deployment of the software system and its evolution roadmap

****

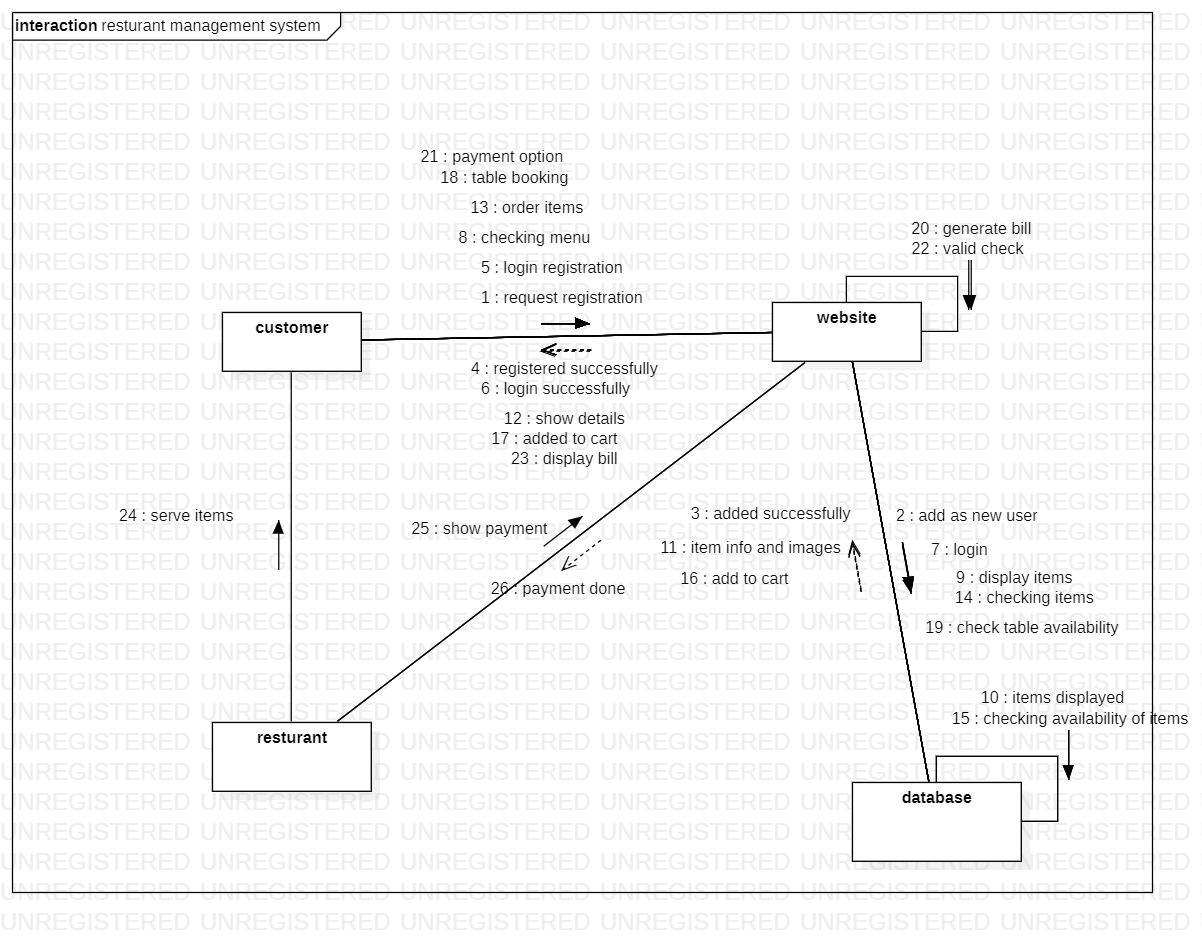
**State Diagram with Description**

A state diagram is a type of diagram used in computer science and related fields to describe the behavior of systems. State diagrams require that the system described is composed of a finite number of states; sometimes, this is indeed the case, while at other times this is a reasonable abstraction.

****

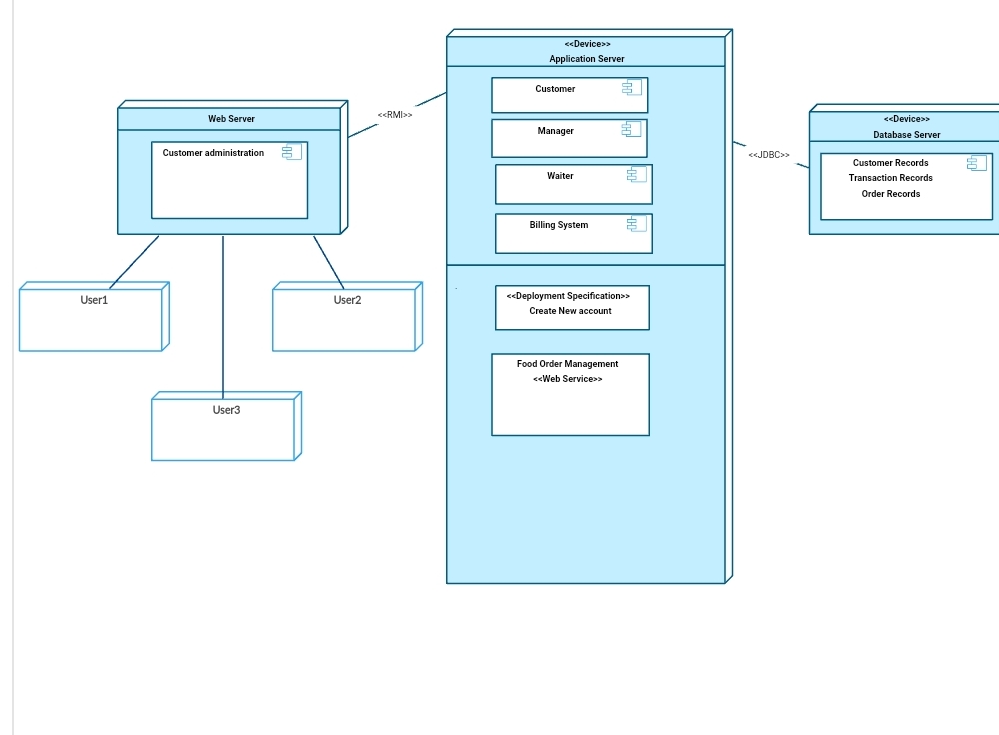
**Collaboration Diagram with Description**

The collaboration diagram is used to show the relationship between the objects in a system. Both the sequence and the collaboration diagrams represent the same information but differently. Instead of showing the flow of messages, it depicts the architecture of the object residing in the system as it is based on object-oriented programming. It is used to portray the object's architecture in the system.



**Deployment Diagram with Description**

A UML deployment diagram is a diagram that shows the configuration of run time processing nodes and the components that live on them. Deployment diagrams is a kind of structure diagram used in modeling the physical aspects of an object-oriented system.



**Result:**

Thus, above mentioned designs of the system were documented successfully.

**EXP 8:**

**Aim**

To describe modules and implement Module1

**Software Used**

**C, C++, Python, HTML, Mysql**, Etc…

**Code of Module 1:**

import sqlite3

import datetime

class Booking:

def getInfo(self,idNumber):

self.id=idNumber

conn=sqlite3.connect("database.db")

c = conn.cursor()

c.execute("select \* from dates WHERE id=?", idNumber)

info=c.fetchone()

print("\n\tYOUR INFO\nOwner: %s\nPeople: %s\nDate: %s\nTime: %s" % (str(info[1]),str(info[2]),str(info[3]),str(info[4])))

conn.close()

def getInfoBrief(self,idNumber):

self.id=int(idNumber)

conn=sqlite3.connect("database.db")

c = conn.cursor()

c.execute("select \* from dates WHERE id=?", str(idNumber))

info=c.fetchone()

print("People: %s\tDate: %s\tTime: %s" % (str(info[2]),str(info[3]),str(info[4])))

conn.close()

def cancel(self):

conn=sqlite3.connect("database.db")

c = conn.cursor()

sure=str(input("Are you sure to cancel[Y/N]: "))

if sure=="Y":

c.execute("delete from dates WHERE id=?", self.id)

conn.commit()

print("The booking was deleted.")

else:

print("The cancellation was stopped.")

conn.close()

def setDate(self,dateOfParameter):

self.dateOf=dateOfParameter

def setTime(self,timeOfParameter):

self.timeOf=timeOfParameter

def setPeople(self,numberOfPeopleParameter):

self.numberOfPeople=numberOfPeopleParameter

def setOwner(self,ownerParameter):

self.owner=ownerParameter

def saveToDB(self):

conn=sqlite3.connect("database.db")

c = conn.cursor()

c.execute('insert into dates(owner,howmanypeople,dateof,timeof) values(?,?,?,?)', [self.owner,self.numberOfPeople,self.dateOf,self.timeOf])

conn.commit()

return c.lastrowid

conn.close()

def newBooking():

aBooking=Booking()

dateRaw=input("Select a date(dd-mm-yyyy): ")

dateProcessed=datetime.datetime.strptime(dateRaw, '%d-%m-%Y')

aBooking.setDate(dateProcessed)

ownerRaw=str(input("Owner name: "))

aBooking.setOwner(ownerRaw)

numberOfPeople=input("Number of people(3,4,5 etc): ")

aBooking.setPeople(numberOfPeople)

timeRaw=input("Select a time: ")

aBooking.setTime(timeRaw)

bookingID=aBooking.saveToDB()

print("\n\tYour booking ID is: %s, please note for other proceess like cancelation.\t" % str(bookingID))

def deleteBooking():

idRaw=input("Please type your booking ID number: ")

aBooking=Booking()

aBooking.getInfo(idRaw)

aBooking.cancel()

def bookingStatus():

dateRaw=input("Select a date(dd-mm-yyyy): ")

dateProcessed=datetime.datetime.strptime(dateRaw, '%d-%m-%Y')

conn=sqlite3.connect("database.db")

c = conn.cursor()

c.execute("select \* from dates WHERE dateOf=(?)", (str(dateProcessed),))

info=c.fetchall()

print("STATUS OF the DAY:\n")

for booking in info:

aBooking=Booking()

aBooking.getInfoBrief(booking[0])

input("Enter a key to continue...")

conn.close()

import sys

def prepareDB():

try:

conn=sqlite3.connect("database.db")

c = conn.cursor()

print("Database connection is successfull.")

c.execute('''CREATE TABLE IF NOT EXISTS dates

(id INTEGER PRIMARY KEY AUTOINCREMENT,

owner TEXT NOT NULL,

howmanypeople INT NOT NULL,

dateof DATE NOT NULL,

timeof INT NOT NULL);''')

print("Table checking is ok.")

print("\n---\n")

conn.commit()

conn.close()

except:

print("Hata")

raise

def whishChoosing():

print("\nPLEASE MAKE A CHOICE:")

print("[1] New booking\n[2] Delete booking\n[3] Check the status for a day\n[4] Exit")

try:

choice = int(input("YOUR CHOICE: "))

return choice

except:

print("Please, just type a number from 1 to 3.")

whishChoosing()

def processChoosing(choosingParameter):

if choosingParameter==1:

newBooking()

elif choosingParameter==2:

deleteBooking()

elif choosingParameter==3:

bookingStatus()

elif choosingParameter==4:

sys.exit()

prepareDB()

currentDT=datetime.datetime.now().strftime("%d-%m-%Y")

print ("Today: %s\nYou should select a date in the next 10 days." % str(currentDT))

while 1:

choosing=whishChoosing()

processChoosing(choosing)

class Booking:

def getInfo(self,idNumber):

self.id=idNumber

conn=sqlite3.connect("database.db")

c = conn.cursor()

c.execute("select \* from dates WHERE id=?", idNumber)

info=c.fetchone()

print("\n\tYOUR INFO\nOwner: %s\nPeople: %s\nDate: %s\nTime: %s" % (str(info[1]),str(info[2]),str(info[3]),str(info[4])))

conn.close()

def getInfoBrief(self,idNumber):

self.id=int(idNumber)

conn=sqlite3.connect("database.db")

c = conn.cursor()

c.execute("select \* from dates WHERE id=?", str(idNumber))

info=c.fetchone()

print("People: %s\tDate: %s\tTime: %s" % (str(info[2]),str(info[3]),str(info[4])))

conn.close()

def cancel(self):

conn=sqlite3.connect("database.db")

c = conn.cursor()

sure=str(input("Are you sure to cancel[Y/N]: "))

if sure=="Y":

c.execute("delete from dates WHERE id=?", self.id)

conn.commit()

print("The booking was deleted.")

else:

print("The cancellation was stopped.")

conn.close()

def setDate(self,dateOfParameter):

self.dateOf=dateOfParameter

def setTime(self,timeOfParameter):

self.timeOf=timeOfParameter

def setPeople(self,numberOfPeopleParameter):

self.numberOfPeople=numberOfPeopleParameter

def setOwner(self,ownerParameter):

self.owner=ownerParameter

def saveToDB(self):

conn=sqlite3.connect("database.db")

c = conn.cursor()

c.execute('insert into dates(owner,howmanypeople,dateof,timeof) values(?,?,?,?)', [self.owner,self.numberOfPeople,self.dateOf,self.timeOf])

conn.commit()

return c.lastrowid

conn.close()

def newBooking():

aBooking=Booking()

dateRaw=input("Select a date(dd-mm-yyyy): ")

dateProcessed=datetime.datetime.strptime(dateRaw, '%d-%m-%Y')

aBooking.setDate(dateProcessed)

ownerRaw=str(input("Owner name: "))

aBooking.setOwner(ownerRaw)

numberOfPeople=input("Number of people(3,4,5 etc): ")

aBooking.setPeople(numberOfPeople)

timeRaw=input("Select a time: ")

aBooking.setTime(timeRaw)

bookingID=aBooking.saveToDB()

print("\n\tYour booking ID is: %s, please note for other proceess like cancellation.\t" % str(bookingID))

def deleteBooking():

idRaw=input("Please type your booking ID number: ")

aBooking=Booking()

aBooking.getInfo(idRaw)

aBooking.cancel()

def bookingStatus():

dateRaw=input("Select a date(dd-mm-yyyy): ")

dateProcessed=datetime.datetime.strptime(dateRaw, '%d-%m-%Y')

conn=sqlite3.connect("database.db")

c = conn.cursor()

c.execute("select \* from dates WHERE dateOf=(?)", (str(dateProcessed),))

info=c.fetchall()

print("STATUS OF the DAY:\n")

for booking in info:

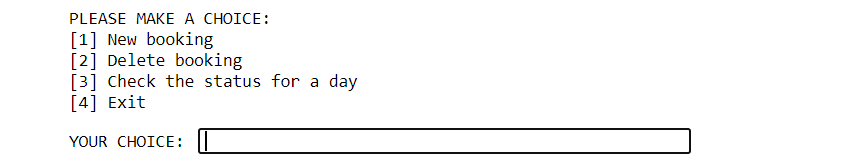
aBooking=Booking()

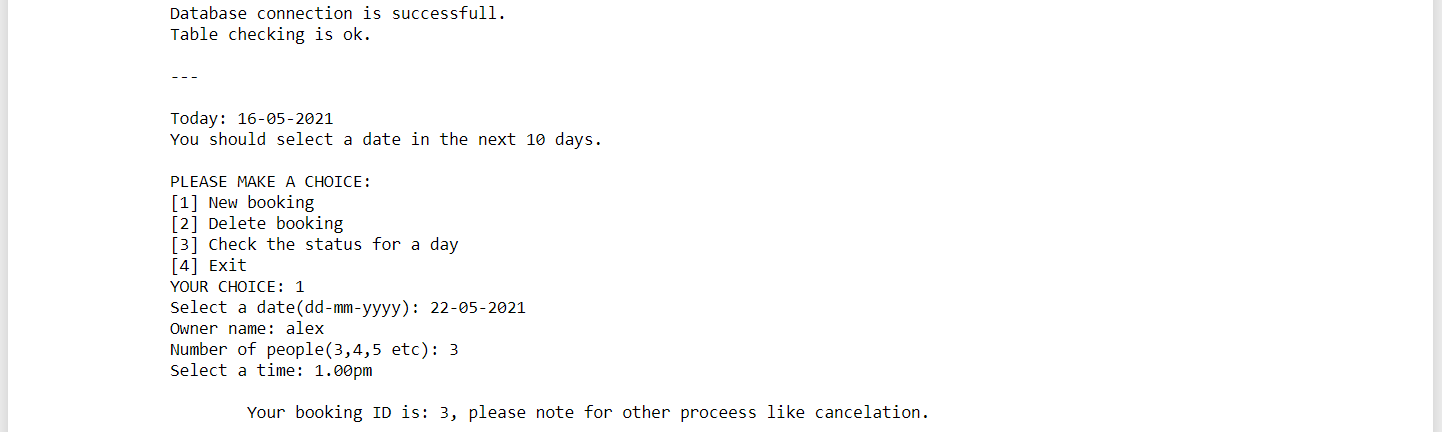
aBooking.getInfoBrief(booking[0])

input("Enter a key to continue...")

conn.close()

**Result of Module 1:**

****

****

Result:

Thus, modules are described, Module 1 was implemented and documented successfully.

**EXP 9:**

**Aim**

To implement Module 2 of the project and display the output of the module with new requirements may assimilated

**Software Used**

**C, C++, Python, HTML, Mysql**, Etc…

**Code of Module 2:**

**from tkinter import\***

**import random**

**import time**

**root = Tk()**

**root.geometry("1600x700+0+0")**

**root.title("Restaurant Management System")**

**Tops = Frame(root,bg="white",width = 1600,height=50,relief=SUNKEN)**

**Tops.pack(side=TOP)**

**f1 = Frame(root,width = 900,height=700,relief=SUNKEN)**

**f1.pack(side=LEFT)**

**f2 = Frame(root ,width = 400,height=700,relief=SUNKEN)**

**f2.pack(side=RIGHT)**

**#------------------TIME--------------**

**localtime=time.asctime(time.localtime(time.time()))**

**#-----------------INFO TOP------------**

**lblinfo = Label(Tops, font=( 'aria' ,30, 'bold' ),text="Restaurant Management System",fg="steel blue",bd=10,anchor='w')**

**lblinfo.grid(row=0,column=0)**

**lblinfo = Label(Tops, font=( 'aria' ,20, ),text=localtime,fg="steel blue",anchor=W)**

**lblinfo.grid(row=1,column=0)**

**#---------------Calculator------------------**

**text\_Input=StringVar()**

**operator =""**

**txtdisplay = Entry(f2,font=('ariel' ,20,'bold'), textvariable=text\_Input , bd=5 ,insertwidth=7 ,bg="white",justify='right')**

**txtdisplay.grid(columnspan=4)**

**def btnclick(numbers):**

**global operator**

**operator=operator + str(numbers)**

**text\_Input.set(operator)**

**def clrdisplay():**

**global operator**

**operator=""**

**text\_Input.set("")**

**def eqals():**

**global operator**

**sumup=str(eval(operator))**

**text\_Input.set(sumup)**

**operator = ""**

**def Ref():**

**x=random.randint(12980, 50876)**

**randomRef = str(x)**

**rand.set(randomRef)**

**cof =float(Fries.get())**

**colfries= float(Largefries.get())**

**cob= float(Burger.get())**

**cofi= float(Filet.get())**

**cochee= float(Cheese\_burger.get())**

**codr= float(Drinks.get())**

**costoffries = cof\*25**

**costoflargefries = colfries\*40**

**costofburger = cob\*35**

**costoffilet = cofi\*50**

**costofcheeseburger = cochee\*50**

**costofdrinks = codr\*35**

**costofmeal = "Rs.",str('%.2f'% (costoffries + costoflargefries + costofburger + costoffilet + costofcheeseburger + costofdrinks))**

**PayTax=((costoffries + costoflargefries + costofburger + costoffilet + costofcheeseburger + costofdrinks)\*0.33)**

**Totalcost=(costoffries + costoflargefries + costofburger + costoffilet + costofcheeseburger + costofdrinks)**

**Ser\_Charge=((costoffries + costoflargefries + costofburger + costoffilet + costofcheeseburger + costofdrinks)/99)**

**Service="Rs.",str('%.2f'% Ser\_Charge)**

**OverAllCost="Rs.",str( PayTax + Totalcost + Ser\_Charge)**

**PaidTax="Rs.",str('%.2f'% PayTax)**

**Service\_Charge.set(Service)**

**cost.set(costofmeal)**

**Tax.set(PaidTax)**

**Subtotal.set(costofmeal)**

**Total.set(OverAllCost)**

**def qexit():**

**root.destroy()**

**def reset():**

**rand.set("")**

**Fries.set("")**

**Largefries.set("")**

**Burger.set("")**

**Filet.set("")**

**Subtotal.set("")**

**Total.set("")**

**Service\_Charge.set("")**

**Drinks.set("")**

**Tax.set("")**

**cost.set("")**

**Cheese\_burger.set("")**

**btn7=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="7",bg="powder blue", command=lambda: btnclick(7) )**

**btn7.grid(row=2,column=0)**

**btn8=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="8",bg="powder blue", command=lambda: btnclick(8) )**

**btn8.grid(row=2,column=1)**

**btn9=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="9",bg="powder blue", command=lambda: btnclick(9) )**

**btn9.grid(row=2,column=2)**

**Addition=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="+",bg="powder blue", command=lambda: btnclick("+") )**

**Addition.grid(row=2,column=3)**

**#---------------------------------------------------------------------------------------------**

**btn4=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="4",bg="powder blue", command=lambda: btnclick(4) )**

**btn4.grid(row=3,column=0)**

**btn5=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="5",bg="powder blue", command=lambda: btnclick(5) )**

**btn5.grid(row=3,column=1)**

**btn6=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="6",bg="powder blue", command=lambda: btnclick(6) )**

**btn6.grid(row=3,column=2)**

**Substraction=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="-",bg="powder blue", command=lambda: btnclick("-") )**

**Substraction.grid(row=3,column=3)**

**#-----------------------------------------------------------------------------------------------**

**btn1=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="1",bg="powder blue", command=lambda: btnclick(1) )**

**btn1.grid(row=4,column=0)**

**btn2=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="2",bg="powder blue", command=lambda: btnclick(2) )**

**btn2.grid(row=4,column=1)**

**btn3=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="3",bg="powder blue", command=lambda: btnclick(3) )**

**btn3.grid(row=4,column=2)**

**multiply=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="",bg="powder blue", command=lambda: btnclick("") )**

**multiply.grid(row=4,column=3)**

**#------------------------------------------------------------------------------------------------**

**btn0=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="0",bg="powder blue", command=lambda: btnclick(0) )**

**btn0.grid(row=5,column=0)**

**btnc=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="c",bg="powder blue", command=clrdisplay)**

**btnc.grid(row=5,column=1)**

**btnequal=Button(f2,padx=16,pady=16,bd=4,width = 16, fg="black", font=('ariel', 20 ,'bold'),text="=",bg="powder blue",command=eqals)**

**btnequal.grid(columnspan=4)**

**Decimal=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text=".",bg="powder blue", command=lambda: btnclick(".") )**

**Decimal.grid(row=5,column=2)**

**Division=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="/",bg="powder blue", command=lambda: btnclick("/") )**

**Division.grid(row=5,column=3)**

**#---------------------------------------------------------------------------------------**

**rand = StringVar()**

**Fries = StringVar()**

**Largefries = StringVar()**

**Burger = StringVar()**

**Filet = StringVar()**

**Subtotal = StringVar()**

**Total = StringVar()**

**Service\_Charge = StringVar()**

**Drinks = StringVar()**

**Tax = StringVar()**

**cost = StringVar()**

**Cheese\_burger = StringVar()**

**lblreference = Label(f1, font=( 'aria' ,16, 'bold' ),text="Order No.",fg="steel blue",bd=10,anchor='w')**

**lblreference.grid(row=0,column=0)**

**txtreference = Entry(f1,font=('ariel' ,16,'bold'), textvariable=rand , bd=6,insertwidth=4,bg="powder blue" ,justify='right')**

**txtreference.grid(row=0,column=1)**

**lblfries = Label(f1, font=( 'aria' ,16, 'bold' ),text="Fries Meal",fg="steel blue",bd=10,anchor='w')**

**lblfries.grid(row=1,column=0)**

**txtfries = Entry(f1,font=('ariel' ,16,'bold'), textvariable=Fries , bd=6,insertwidth=4,bg="powder blue" ,justify='right')**

**txtfries.grid(row=1,column=1)**

**lblLargefries = Label(f1, font=( 'aria' ,16, 'bold' ),text="Lunch Meal",fg="steel blue",bd=10,anchor='w')**

**lblLargefries.grid(row=2,column=0)**

**txtLargefries = Entry(f1,font=('ariel' ,16,'bold'), textvariable=Largefries , bd=6,insertwidth=4,bg="powder blue" ,justify='right')**

**txtLargefries.grid(row=2,column=1)**

**lblburger = Label(f1, font=( 'aria' ,16, 'bold' ),text="Burger Meal",fg="steel blue",bd=10,anchor='w')**

**lblburger.grid(row=3,column=0)**

**txtburger = Entry(f1,font=('ariel' ,16,'bold'), textvariable=Burger , bd=6,insertwidth=4,bg="powder blue" ,justify='right')**

**txtburger.grid(row=3,column=1)**

**lblFilet = Label(f1, font=( 'aria' ,16, 'bold' ),text="Pizza Meal",fg="steel blue",bd=10,anchor='w')**

**lblFilet.grid(row=4,column=0)**

**txtFilet = Entry(f1,font=('ariel' ,16,'bold'), textvariable=Filet , bd=6,insertwidth=4,bg="powder blue" ,justify='right')**

**txtFilet.grid(row=4,column=1)**

**lblCheese\_burger = Label(f1, font=( 'aria' ,16, 'bold' ),text="Cheese burger",fg="steel blue",bd=10,anchor='w')**

**lblCheese\_burger.grid(row=5,column=0)**

**txtCheese\_burger = Entry(f1,font=('ariel' ,16,'bold'), textvariable=Cheese\_burger , bd=6,insertwidth=4,bg="powder blue" ,justify='right')**

**txtCheese\_burger.grid(row=5,column=1)**

**#--------------------------------------------------------------------------------------**

**lblDrinks = Label(f1, font=( 'aria' ,16, 'bold' ),text="Drinks",fg="steel blue",bd=10,anchor='w')**

**lblDrinks.grid(row=0,column=2)**

**txtDrinks = Entry(f1,font=('ariel' ,16,'bold'), textvariable=Drinks , bd=6,insertwidth=4,bg="powder blue" ,justify='right')**

**txtDrinks.grid(row=0,column=3)**

**lblcost = Label(f1, font=( 'aria' ,16, 'bold' ),text="cost",fg="steel blue",bd=10,anchor='w')**

**lblcost.grid(row=1,column=2)**

**txtcost = Entry(f1,font=('ariel' ,16,'bold'), textvariable=cost , bd=6,insertwidth=4,bg="powder blue" ,justify='right')**

**txtcost.grid(row=1,column=3)**

**lblService\_Charge = Label(f1, font=( 'aria' ,16, 'bold' ),text="Service Charge",fg="steel blue",bd=10,anchor='w')**

**lblService\_Charge.grid(row=2,column=2)**

**txtService\_Charge = Entry(f1,font=('ariel' ,16,'bold'), textvariable=Service\_Charge , bd=6,insertwidth=4,bg="powder blue" ,justify='right')**

**txtService\_Charge.grid(row=2,column=3)**

**lblTax = Label(f1, font=( 'aria' ,16, 'bold' ),text="Tax",fg="steel blue",bd=10,anchor='w')**

**lblTax.grid(row=3,column=2)**

**txtTax = Entry(f1,font=('ariel' ,16,'bold'), textvariable=Tax , bd=6,insertwidth=4,bg="powder blue" ,justify='right')**

**txtTax.grid(row=3,column=3)**

**lblSubtotal = Label(f1, font=( 'aria' ,16, 'bold' ),text="Subtotal",fg="steel blue",bd=10,anchor='w')**

**lblSubtotal.grid(row=4,column=2)**

**txtSubtotal = Entry(f1,font=('ariel' ,16,'bold'), textvariable=Subtotal , bd=6,insertwidth=4,bg="powder blue" ,justify='right')**

**txtSubtotal.grid(row=4,column=3)**

**lblTotal = Label(f1, font=( 'aria' ,16, 'bold' ),text="Total",fg="steel blue",bd=10,anchor='w')**

**lblTotal.grid(row=5,column=2)**

**txtTotal = Entry(f1,font=('ariel' ,16,'bold'), textvariable=Total , bd=6,insertwidth=4,bg="powder blue" ,justify='right')**

**txtTotal.grid(row=5,column=3)**

**#-----------------------------------------buttons------------------------------------------**

**lblTotal = Label(f1,text="---------------------",fg="white")**

**lblTotal.grid(row=6,columnspan=3)**

**btnTotal=Button(f1,padx=16,pady=8, bd=10 ,fg="black",font=('ariel' ,16,'bold'),width=10, text="TOTAL", bg="powder blue",command=Ref)**

**btnTotal.grid(row=7, column=1)**

**btnreset=Button(f1,padx=16,pady=8, bd=10 ,fg="black",font=('ariel' ,16,'bold'),width=10, text="RESET", bg="powder blue",command=reset)**

**btnreset.grid(row=7, column=2)**

**btnexit=Button(f1,padx=16,pady=8, bd=10 ,fg="black",font=('ariel' ,16,'bold'),width=10, text="EXIT", bg="powder blue",command=qexit)**

**btnexit.grid(row=7, column=3)**

**def price():**

**roo = Tk()**

**roo.geometry("600x220+0+0")**

**roo.title("Price List")**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="ITEM", fg="black", bd=5)**

**lblinfo.grid(row=0, column=0)**

**lblinfo = Label(roo, font=('aria', 15,'bold'), text="\_\_\_\_\_", fg="white", anchor=W)**

**lblinfo.grid(row=0, column=2)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="PRICE", fg="black", anchor=W)**

**lblinfo.grid(row=0, column=3)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="Fries Meal", fg="steel blue", anchor=W)**

**lblinfo.grid(row=1, column=0)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="25", fg="steel blue", anchor=W)**

**lblinfo.grid(row=1, column=3)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="Lunch Meal", fg="steel blue", anchor=W)**

**lblinfo.grid(row=2, column=0)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="40", fg="steel blue", anchor=W)**

**lblinfo.grid(row=2, column=3)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="Burger Meal", fg="steel blue", anchor=W)**

**lblinfo.grid(row=3, column=0)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="35", fg="steel blue", anchor=W)**

**lblinfo.grid(row=3, column=3)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="Pizza Meal", fg="steel blue", anchor=W)**

**lblinfo.grid(row=4, column=0)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="50", fg="steel blue", anchor=W)**

**lblinfo.grid(row=4, column=3)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="Cheese Burger", fg="steel blue", anchor=W)**

**lblinfo.grid(row=5, column=0)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="30", fg="steel blue", anchor=W)**

**lblinfo.grid(row=5, column=3)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="Drinks", fg="steel blue", anchor=W)**

**lblinfo.grid(row=6, column=0)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="35", fg="steel blue", anchor=W)**

**lblinfo.grid(row=6, column=3)**

**roo.mainloop()**

**btnprice=Button(f1,padx=16,pady=8, bd=10 ,fg="black",font=('ariel' ,16,'bold'),width=10, text="PRICE", bg="powder blue",command=price)**

**btnprice.grid(row=7, column=0)**

**root.mainloop()**

**Result of Module 2:**

****

Result:

Thus, the module2 was implemented and documented successfully.

**EXP 10:**

**Aim**

To implement Module 3 of the project and display the output of the module with solving New Issues.

**Software Used**

**C, C++, Python, HTML, Mysql**, Etc…

**Code of Module 3:**

**from tkinter import\***

**import random**

**import time**

**root = Tk()**

**root.geometry("1600x700+0+0")**

**root.title("Restaurant Management System")**

**Tops = Frame(root,bg="white",width = 1600,height=50,relief=SUNKEN)**

**Tops.pack(side=TOP)**

**f1 = Frame(root,width = 900,height=700,relief=SUNKEN)**

**f1.pack(side=LEFT)**

**f2 = Frame(root ,width = 400,height=700,relief=SUNKEN)**

**f2.pack(side=RIGHT)**

**#------------------TIME--------------**

**localtime=time.asctime(time.localtime(time.time()))**

**#-----------------INFO TOP------------**

**lblinfo = Label(Tops, font=( 'aria' ,30, 'bold' ),text="Restaurant Management System",fg="steel blue",bd=10,anchor='w')**

**lblinfo.grid(row=0,column=0)**

**lblinfo = Label(Tops, font=( 'aria' ,20, ),text=localtime,fg="steel blue",anchor=W)**

**lblinfo.grid(row=1,column=0)**

**#---------------Calculator------------------**

**text\_Input=StringVar()**

**operator =""**

**txtdisplay = Entry(f2,font=('ariel' ,20,'bold'), textvariable=text\_Input , bd=5 ,insertwidth=7 ,bg="white",justify='right')**

**txtdisplay.grid(columnspan=4)**

**def btnclick(numbers):**

**global operator**

**operator=operator + str(numbers)**

**text\_Input.set(operator)**

**def clrdisplay():**

**global operator**

**operator=""**

**text\_Input.set("")**

**def eqals():**

**global operator**

**sumup=str(eval(operator))**

**text\_Input.set(sumup)**

**operator = ""**

**def Ref():**

**x=random.randint(12980, 50876)**

**randomRef = str(x)**

**rand.set(randomRef)**

**cof =float(Fries.get())**

**colfries= float(Largefries.get())**

**cob= float(Burger.get())**

**cofi= float(Filet.get())**

**cochee= float(Cheese\_burger.get())**

**codr= float(Drinks.get())**

**costoffries = cof\*25**

**costoflargefries = colfries\*40**

**costofburger = cob\*35**

**costoffilet = cofi\*50**

**costofcheeseburger = cochee\*50**

**costofdrinks = codr\*35**

**costofmeal = "Rs.",str('%.2f'% (costoffries + costoflargefries + costofburger + costoffilet + costofcheeseburger + costofdrinks))**

**PayTax=((costoffries + costoflargefries + costofburger + costoffilet + costofcheeseburger + costofdrinks)\*0.33)**

**Totalcost=(costoffries + costoflargefries + costofburger + costoffilet + costofcheeseburger + costofdrinks)**

**Ser\_Charge=((costoffries + costoflargefries + costofburger + costoffilet + costofcheeseburger + costofdrinks)/99)**

**Service="Rs.",str('%.2f'% Ser\_Charge)**

**OverAllCost="Rs.",str( PayTax + Totalcost + Ser\_Charge)**

**PaidTax="Rs.",str('%.2f'% PayTax)**

**Service\_Charge.set(Service)**

**cost.set(costofmeal)**

**Tax.set(PaidTax)**

**Subtotal.set(costofmeal)**

**Total.set(OverAllCost)**

**def qexit():**

**root.destroy()**

**def reset():**

**rand.set("")**

**Fries.set("")**

**Largefries.set("")**

**Burger.set("")**

**Filet.set("")**

**Subtotal.set("")**

**Total.set("")**

**Service\_Charge.set("")**

**Drinks.set("")**

**Tax.set("")**

**cost.set("")**

**Cheese\_burger.set("")**

**btn7=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="7",bg="powder blue", command=lambda: btnclick(7) )**

**btn7.grid(row=2,column=0)**

**btn8=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="8",bg="powder blue", command=lambda: btnclick(8) )**

**btn8.grid(row=2,column=1)**

**btn9=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="9",bg="powder blue", command=lambda: btnclick(9) )**

**btn9.grid(row=2,column=2)**

**Addition=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="+",bg="powder blue", command=lambda: btnclick("+") )**

**Addition.grid(row=2,column=3)**

**#---------------------------------------------------------------------------------------------**

**btn4=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="4",bg="powder blue", command=lambda: btnclick(4) )**

**btn4.grid(row=3,column=0)**

**btn5=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="5",bg="powder blue", command=lambda: btnclick(5) )**

**btn5.grid(row=3,column=1)**

**btn6=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="6",bg="powder blue", command=lambda: btnclick(6) )**

**btn6.grid(row=3,column=2)**

**Substraction=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="-",bg="powder blue", command=lambda: btnclick("-") )**

**Substraction.grid(row=3,column=3)**

**#-----------------------------------------------------------------------------------------------**

**btn1=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="1",bg="powder blue", command=lambda: btnclick(1) )**

**btn1.grid(row=4,column=0)**

**btn2=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="2",bg="powder blue", command=lambda: btnclick(2) )**

**btn2.grid(row=4,column=1)**

**btn3=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="3",bg="powder blue", command=lambda: btnclick(3) )**

**btn3.grid(row=4,column=2)**

**multiply=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="",bg="powder blue", command=lambda: btnclick("") )**

**multiply.grid(row=4,column=3)**

**#------------------------------------------------------------------------------------------------**

**btn0=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="0",bg="powder blue", command=lambda: btnclick(0) )**

**btn0.grid(row=5,column=0)**

**btnc=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="c",bg="powder blue", command=clrdisplay)**

**btnc.grid(row=5,column=1)**

**btnequal=Button(f2,padx=16,pady=16,bd=4,width = 16, fg="black", font=('ariel', 20 ,'bold'),text="=",bg="powder blue",command=eqals)**

**btnequal.grid(columnspan=4)**

**Decimal=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text=".",bg="powder blue", command=lambda: btnclick(".") )**

**Decimal.grid(row=5,column=2)**

**Division=Button(f2,padx=16,pady=16,bd=4, fg="black", font=('ariel', 20 ,'bold'),text="/",bg="powder blue", command=lambda: btnclick("/") )**

**Division.grid(row=5,column=3)**

**#---------------------------------------------------------------------------------------**

**rand = StringVar()**

**Fries = StringVar()**

**Largefries = StringVar()**

**Burger = StringVar()**

**Filet = StringVar()**

**Subtotal = StringVar()**

**Total = StringVar()**

**Service\_Charge = StringVar()**

**Drinks = StringVar()**

**Tax = StringVar()**

**cost = StringVar()**

**Cheese\_burger = StringVar()**

**lblreference = Label(f1, font=( 'aria' ,16, 'bold' ),text="Order No.",fg="steel blue",bd=10,anchor='w')**

**lblreference.grid(row=0,column=0)**

**txtreference = Entry(f1,font=('ariel' ,16,'bold'), textvariable=rand , bd=6,insertwidth=4,bg="powder blue" ,justify='right')**

**txtreference.grid(row=0,column=1)**

**lblfries = Label(f1, font=( 'aria' ,16, 'bold' ),text="Fries Meal",fg="steel blue",bd=10,anchor='w')**

**lblfries.grid(row=1,column=0)**

**txtfries = Entry(f1,font=('ariel' ,16,'bold'), textvariable=Fries , bd=6,insertwidth=4,bg="powder blue" ,justify='right')**

**txtfries.grid(row=1,column=1)**

**lblLargefries = Label(f1, font=( 'aria' ,16, 'bold' ),text="Lunch Meal",fg="steel blue",bd=10,anchor='w')**

**lblLargefries.grid(row=2,column=0)**

**txtLargefries = Entry(f1,font=('ariel' ,16,'bold'), textvariable=Largefries , bd=6,insertwidth=4,bg="powder blue" ,justify='right')**

**txtLargefries.grid(row=2,column=1)**

**lblburger = Label(f1, font=( 'aria' ,16, 'bold' ),text="Burger Meal",fg="steel blue",bd=10,anchor='w')**

**lblburger.grid(row=3,column=0)**

**txtburger = Entry(f1,font=('ariel' ,16,'bold'), textvariable=Burger , bd=6,insertwidth=4,bg="powder blue" ,justify='right')**

**txtburger.grid(row=3,column=1)**

**lblFilet = Label(f1, font=( 'aria' ,16, 'bold' ),text="Pizza Meal",fg="steel blue",bd=10,anchor='w')**

**lblFilet.grid(row=4,column=0)**

**txtFilet = Entry(f1,font=('ariel' ,16,'bold'), textvariable=Filet , bd=6,insertwidth=4,bg="powder blue" ,justify='right')**

**txtFilet.grid(row=4,column=1)**

**lblCheese\_burger = Label(f1, font=( 'aria' ,16, 'bold' ),text="Cheese burger",fg="steel blue",bd=10,anchor='w')**

**lblCheese\_burger.grid(row=5,column=0)**

**txtCheese\_burger = Entry(f1,font=('ariel' ,16,'bold'), textvariable=Cheese\_burger , bd=6,insertwidth=4,bg="powder blue" ,justify='right')**

**txtCheese\_burger.grid(row=5,column=1)**

**#--------------------------------------------------------------------------------------**

**lblDrinks = Label(f1, font=( 'aria' ,16, 'bold' ),text="Drinks",fg="steel blue",bd=10,anchor='w')**

**lblDrinks.grid(row=0,column=2)**

**txtDrinks = Entry(f1,font=('ariel' ,16,'bold'), textvariable=Drinks , bd=6,insertwidth=4,bg="powder blue" ,justify='right')**

**txtDrinks.grid(row=0,column=3)**

**lblcost = Label(f1, font=( 'aria' ,16, 'bold' ),text="cost",fg="steel blue",bd=10,anchor='w')**

**lblcost.grid(row=1,column=2)**

**txtcost = Entry(f1,font=('ariel' ,16,'bold'), textvariable=cost , bd=6,insertwidth=4,bg="powder blue" ,justify='right')**

**txtcost.grid(row=1,column=3)**

**lblService\_Charge = Label(f1, font=( 'aria' ,16, 'bold' ),text="Service Charge",fg="steel blue",bd=10,anchor='w')**

**lblService\_Charge.grid(row=2,column=2)**

**txtService\_Charge = Entry(f1,font=('ariel' ,16,'bold'), textvariable=Service\_Charge , bd=6,insertwidth=4,bg="powder blue" ,justify='right')**

**txtService\_Charge.grid(row=2,column=3)**

**lblTax = Label(f1, font=( 'aria' ,16, 'bold' ),text="Tax",fg="steel blue",bd=10,anchor='w')**

**lblTax.grid(row=3,column=2)**

**txtTax = Entry(f1,font=('ariel' ,16,'bold'), textvariable=Tax , bd=6,insertwidth=4,bg="powder blue" ,justify='right')**

**txtTax.grid(row=3,column=3)**

**lblSubtotal = Label(f1, font=( 'aria' ,16, 'bold' ),text="Subtotal",fg="steel blue",bd=10,anchor='w')**

**lblSubtotal.grid(row=4,column=2)**

**txtSubtotal = Entry(f1,font=('ariel' ,16,'bold'), textvariable=Subtotal , bd=6,insertwidth=4,bg="powder blue" ,justify='right')**

**txtSubtotal.grid(row=4,column=3)**

**lblTotal = Label(f1, font=( 'aria' ,16, 'bold' ),text="Total",fg="steel blue",bd=10,anchor='w')**

**lblTotal.grid(row=5,column=2)**

**txtTotal = Entry(f1,font=('ariel' ,16,'bold'), textvariable=Total , bd=6,insertwidth=4,bg="powder blue" ,justify='right')**

**txtTotal.grid(row=5,column=3)**

**#-----------------------------------------buttons------------------------------------------**

**lblTotal = Label(f1,text="---------------------",fg="white")**

**lblTotal.grid(row=6,columnspan=3)**

**btnTotal=Button(f1,padx=16,pady=8, bd=10 ,fg="black",font=('ariel' ,16,'bold'),width=10, text="TOTAL", bg="powder blue",command=Ref)**

**btnTotal.grid(row=7, column=1)**

**btnreset=Button(f1,padx=16,pady=8, bd=10 ,fg="black",font=('ariel' ,16,'bold'),width=10, text="RESET", bg="powder blue",command=reset)**

**btnreset.grid(row=7, column=2)**

**btnexit=Button(f1,padx=16,pady=8, bd=10 ,fg="black",font=('ariel' ,16,'bold'),width=10, text="EXIT", bg="powder blue",command=qexit)**

**btnexit.grid(row=7, column=3)**

**def price():**

**roo = Tk()**

**roo.geometry("600x220+0+0")**

**roo.title("Price List")**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="ITEM", fg="black", bd=5)**

**lblinfo.grid(row=0, column=0)**

**lblinfo = Label(roo, font=('aria', 15,'bold'), text="\_\_\_\_\_", fg="white", anchor=W)**

**lblinfo.grid(row=0, column=2)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="PRICE", fg="black", anchor=W)**

**lblinfo.grid(row=0, column=3)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="Fries Meal", fg="steel blue", anchor=W)**

**lblinfo.grid(row=1, column=0)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="25", fg="steel blue", anchor=W)**

**lblinfo.grid(row=1, column=3)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="Lunch Meal", fg="steel blue", anchor=W)**

**lblinfo.grid(row=2, column=0)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="40", fg="steel blue", anchor=W)**

**lblinfo.grid(row=2, column=3)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="Burger Meal", fg="steel blue", anchor=W)**

**lblinfo.grid(row=3, column=0)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="35", fg="steel blue", anchor=W)**

**lblinfo.grid(row=3, column=3)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="Pizza Meal", fg="steel blue", anchor=W)**

**lblinfo.grid(row=4, column=0)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="50", fg="steel blue", anchor=W)**

**lblinfo.grid(row=4, column=3)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="Cheese Burger", fg="steel blue", anchor=W)**

**lblinfo.grid(row=5, column=0)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="30", fg="steel blue", anchor=W)**

**lblinfo.grid(row=5, column=3)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="Drinks", fg="steel blue", anchor=W)**

**lblinfo.grid(row=6, column=0)**

**lblinfo = Label(roo, font=('aria', 15, 'bold'), text="35", fg="steel blue", anchor=W)**

**lblinfo.grid(row=6, column=3)**

**roo.mainloop()**

**btnprice=Button(f1,padx=16,pady=8, bd=10 ,fg="black",font=('ariel' ,16,'bold'),width=10, text="PRICE", bg="powder blue",command=price)**

**btnprice.grid(row=7, column=0)**

**root.mainloop()**

**Result of Module 3:**

****

Result:

Thus, the module3 was implemented and documented successfully.

**EXP 11:**

**Aim**

To Prepare master test plan and Test cases for testing the project

**Test plan template will be incorporated**

Test Plan, Test Case

Lab Session #11

Table of Contents

[1. Executive Summary 4](#_Toc61904347)

[2. Test Plan 4](#_Toc61904348)

[2.1. Scope of Testing 4](#_Toc61904349)

[2.2. Types of Testing , Methodology , Tools 5](#_Toc61904350)

[2.4. Test Deliverables 5](#_Toc61904352)

[3. Test Case 5](#_Toc61904353)

[3.1. Functional Test Cases 5](#_Toc61904354)

[3.1. Non-Functional Test Cases 6](#_Toc61904355)

[Reference 7](#_Toc61904358)

# Executive Summary

A Test Plan is a detailed document that describes the test strategy, objectives, schedule, estimation, deliverables, and resources required to perform testing for a software product.

Test Plan helps us determine the effort needed to validate the quality of the application under test.

The test plan serves as a blueprint to conduct software testing activities as a defined process, which is minutely monitored and controlled by the test manager.

# Test Plan

We have used White box Unit testing for the functional testing and Black Box testing for the non-functional testing

# Scope of Testing

The goal of utilizing numerous testing methodologies in the development process is to make sure that your software can successfully operate in multiple environments and across different platforms. These can typically be broken down between functional and non-functional testing.

**Functional:**

1.Functional testing involves testing the application against the business requirements.

2.It incorporates all test types designed to guarantee each part of a piece of software behaves as expected.

3.These testing methods are usually conducted in order and include:

·Unit testing

·Integration testing

·System testing

·Acceptance testing

**Non-Functional:**

Non-functional testing methods incorporate all test types    focused on the operational aspects of a piece of software. These include:

·Performance testing

·Security testing

·Usability testing

·Compatibility testing

# Types of Testing , Methodology , Tools

|  |  |  |
| --- | --- | --- |
| Category | Methodology | Tools Required |
| Non Functional Requirements | Manual | Testing |
| Functional Requirements | Manual -GUI testing, regression testing, frontend testing, backend testing | GUI testing, regression testing, front-end testing, password validation, back-end testing |

# Test Deliverables

·Test plan

·Test strategy

·Bug report

·Test execution report

# Test Case

# Functional Test Cases

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test ID (#)** | **Test Case** | **Test Scenario** | **Execution Steps** | **Expected Outcome** | **Actual Outcome** | **Status** | **Remarks** |
| 1 | Respective user to click on their respective job | Backend/Design | user clicks button | They have to get their respective booking page |  |  |  |
| 2 | To verify valid details | Backend | user clicks on login | User should book his slot |  |  |  |
| 3. | Booking page | Frontend/validation | User click enter | It should move to next step |  |  |  |
| 4. | User should get their respective  work | Python working properly- Design | user after booking | Display their work |  |  |  |
| 5. | To Check E-menu | Backend | To take the order details from the customer and generates bill automatically | python script executes correctly |  |  |  |
| 6. | Exit | Backend | click on the exit button to exit | We have to exit from application |  |  |  |

n

# Non-Functional Test Cases

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test ID (#)** | **Test Case** | **Test Scenario** | **Execution Steps** | **Expected Outcome** | **Actual Outcome** | **Status** | **Remarks** |
| 1 | Booking page | User friendly  design | NIL | Booking page should come | Booking page should come |  |  |
| 2. | Restaurant management system logo centred | User friendly  design | NIL | Restaurant management system logo expected to be centred | Restaurant management system  logo to be centred |  |  |
| 3. | Required buttons: price,reset, total, exit | User friendly  design | NIL | expected buttons are present on page | buttons are present on page |  |  |
| 4. | Button background color: light blue | User friendly  design | NIL | button background color expected to be beige | button background color to be beige |  |  |
| 5. | Button border color | User friendly  design | NIL | button border color expected | button border color |  |  |
| 6. | Button foreground color: black | User friendly  design | NIL | button foreground color expected to be black | button foreground color to be black |  |  |
| 7. | Buttons size,Height and width | User friendly  design | NIL | Buttons expected height,expected width | Buttons height , width |  |  |

# Test ReportReference

1. <https://www.pmi.org/>

Result:

Thus, the test plan and test cases are documented successfully

**EXP 12:**

**Aim**

To conduct manual test using Test cases and prepare test report for the project

***Manual Testing with report***

***Lab Session #12 ss***

**Table of Contents**

[1. Executive Summary 5](#_Toc70926979)

[2. Test Plan 6](#_Toc70926980)

[2.1. Scope of Testing 6](#_Toc70926981)

[·User friendly interface 6](#_Toc70926982)

[·Easy to operate 6](#_Toc70926983)

[·Ensures security of data 6](#_Toc70926984)

[·Reliable 6](#_Toc70926985)

[·Generates accurate reports 6](#_Toc70926986)

[·All the functional and non-functional requirements are fulfilled 6](#_Toc70926987)

[Functional: All modules have been covered and tested 6](#_Toc70926988)

[2.2. Types of Testing , Methodology , Tools 6](#_Toc70926989)

[2.3. Test Deliverables 6](#_Toc70926990)

[3. Test Case 7](#_Toc70926991)

[3.1. Functional Test Cases 7](#_Toc70926992)

[1.1. Non-Functional Test Cases 9](#_Toc70926993)

[2. Defect Log 13](#_Toc70926994)

[3. Test Report 13](#_Toc70926995)

[Reference 13](#_Toc70926996)

# Executive Summary

·Scope of testing

·Methodology

i)Functional

ii) Non-functional Requirements

·Tools Required

·Test Deliverables

·Test Case

·Test Report

# Test Plan

# Scope of Testing

# ·User friendly interface

# ·Easy to operate

# ·Ensures security of data

# ·All the functional and non-functional requirements are fulfilled

# 

# Functional: All modules have been covered and tested

**Non-Functional:** All Non-Functional requirements have been covered and tested

# Types of Testing , Methodology , Tools

|  |  |  |
| --- | --- | --- |
| Category | Methodology | Tools Required |
| Functional Requirements | Manual | Word Template |

# Test Deliverables

.Test plan

.Test strategy

.Bug report

.Test execution report

.Test summary

.Release note

# Test Case

# Functional Test Cases

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test ID (#)** | **Test Case** | **Test Scenario** | **Execution Steps** | **Expected Outcome** | **Actual Outcome** | **Status** | **Remarks** |
| 1 | Respective user to click on their respective job | Backend/Design | user clicks button | They have to get their respective booking page | Users have to get their respective booking page | Pass | success |
| 2 | To verify valid details | Backend | user clicks on login | User should book his slot | User should book his slot | Pass | success |
| 3. | Booking page | Frontend/validation | User click enter | It should move to next step | move to next step | Pass | success |
| 4. | User should get their respective  work | Python working properly- Design | user after booking | Display their work | Display their work | pass | success |
| 5. | To Check E-menu | Backend | To take the order details from the customer and generates bill automatically | python script executes correctly | python script executes correctly | pass | success |
| 6. | Exit | Backend | click on the exit button to exit | To exit from application | To exit from application | pass | success |

# Non-Functional Test Cases

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test ID (#)** | **Test Case** | **Test Scenario** | **Execution Steps** | **Expected Outcome** | **Actual Outcome** | **Status** | **Remarks** |
| 1 | Booking page | User friendly  design | NIL | Booking page should come | Booking page should come | Pass | success |
| 2. | Restaurant management system logo centred | User friendly  design | NIL | Restaurant management system logo expected to be centred | Restaurant management system  logo to be centred | Pass | success |
| 3. | Required buttons: price,reset, total, exit | User friendly  design | NIL | expected buttons are present on page | buttons are present on page | Pass | success |
| 4. | Button background color: light blue | User friendly  design | NIL | button background color expected to be beige | button background color to be beige | Pass | success |
| 5. | Button border color | User friendly  design | NIL | button border color expected | button border color | Pass | success |
| 6. | Button foreground color: black | User friendly  design | NIL | button foreground color expected to be black | button foreground color to be black | Pass | success |
| 7. | Buttons size,Height and width | User friendly  design | NIL | Buttons expected height,expected width | Buttons height , width | Pass | success |

# Defect Log

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Requirement #** | **Defect ID #** | **Defect Description** | **Assignee** | **Status** |
| Module 2 | 1 | Have to add more buttons | Dasu sarat | under scrutiny |
| Module 1 | 2 | Have to add more Category | S.yaswanth sai | under scrutiny |

# Test Report

Functional Testing is currently in Progress. Defects and Failures are currently being scrutinised by developers and testers.

Not Functional testing is complete, all test cases have been carefully looked into and successful.

Necessary updates will be pushed as soon as the defects are nullified and the necessary changes are made.

We currently have improvements to be considered on the Python Script present in Module 3 of the project which the testers are actively working on.

|  |  |  |
| --- | --- | --- |
| **Category** | **Progress Against Plan** | **Status** |
| Functional Testing | Amber | In-Progress |
| Non-Functional Testing | Amber | In-Progress |

|  |  |  |
| --- | --- | --- |
| **Functional** | **Test Case Coverage (%)** | **Status** |
| Module ID 1 | 80% | In-Progress |
| Module ID 2 | 60%-80% | In-Progress |

# 

# Reference

1. <https://www.pmi.org/>

**Result:**

Thus, the software test conducted and documented the report successfully