

A PROJECT REPORT
ON
Online Food Ordering System

A Project Report Submitted in the Partial Fulfilment of requirements to
Jawaharlal Nehru Technological University, Kakinada

For the Award of the Degree of
BACHELOR OF TECHNOLOGY

In
CSE - DATA SCIENCE

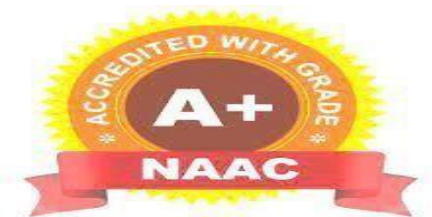
Submitted by

Sk.Shareefa	(20KE1A4448)
Ch.Rukmini	(20KE1A4406)
J.Pravalika	(20KE1A4417)
P.Asifa Kousar	(20KE1A4440)
R.Poojitha	(20KE1A4446)

under the guidance of

Dr.G.Rama Swamy M.Tech.Ph.D

Professor



DEPARTMENT OF CSE- DATA SCIENCE

MALINENI LAKSHMAIAH WOMEN'S ENGINEERING COLLEGE

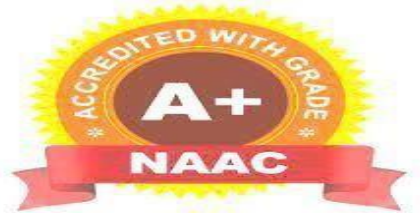
(Accredited by NBA(CSE&ECE), NAAC A+, Approved by AICTE, NEW DELHI, Affiliated to JNTU

Kakinada) Pulladigunta(V), Vatticherukuru(Md), Guntur(Dt),AP,-522017. 2020 – 2024

MALINENI LAKSHMAIAH WOMEN'S ENGINEERING COLLEGE

Pulladigunta(V), Vatticherukuru(Md), Guntur(Dt),AP,-522017.

DEPARTMENT OF CSE- DATA SCIENCE



CERTIFICATE

This is to certify that the project entitled “**Online Food Ordering System**” is the bonafide work carried out by

Sk.Shareefa	(20KE1A4448)
Ch.Rukmini	(20KE1A4406)
J.Pravalika	(20KE1A4417)
P.Asifa Kousar	(20KE1A4440)
R.Poojitha	(20KE1A4446)

B.Tech, CSE - DS, Affiliated to JNTU Kakinada in partial fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY** with the specialization in **CSE - DATA SCIENCE** during the Academic year 2020-2024.

Signature of the Guide
Dr.G.Rama swamy,Ph.D

Head of the Department
Prof. N. HariKrishna

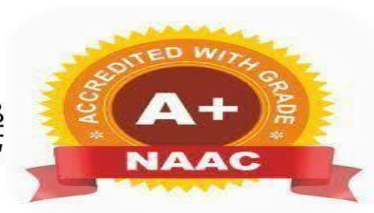
Viva – voice Held on _____

External Examiner

MALINENI LAKSHMAIAH WOMEN'S ENGINEERING COLLEGE

Pulladigunta(V), Vatticherukuru(Md), Guntur(Dt),AP,-522017.

DEPARTMENT OF CSE- DATA SCIENCE



DECLARATION

We hereby declare that the project work entitled **“Online Food Ordering System”** is entirely our original work carried out under the guidance of **Dr.G.Rama swamy**, Professor in the Department of Computer Science and Engineering, Malineni Lakshmaiah Women's Engineering College, Pulladigunta ,Guntur ,JNTU Kakinada, A.P ,India, for the award of the **DEGREE OF BACHELOR OF TECHNOLOGY** with the specialization in **CSE-DATA SCIENCE**. The results carried out in this project report have not been submitted in a part or full for the award of any degree or diploma of this or any other university or institute.

Submitted By

Sk.Shareefa	(20KE1A4448) Signature_____
Ch.Rukmini	(20KE1A4406) Signature_____
J.Pravalika	(20KE1A4417) Signature_____
P.Asifa Kousar	(20KE1A4440)Signature_____
R.Poojitha	(20KE5A4446) Signature_____

ACKNOWLEDGEMENT

All endeavors over a long period can be successful only with the advice and support of many well-wishers. I take this opportunity to express my gratitude and appreciation to all of them.

We wish to express deep sense of gratitude to my beloved and respected guide **Dr.G.Rama swamy,Ph.D**, Professor in **the department of CSE**, Malineni Lakshmaiah Women's Engineering College, Guntur, for his valuable guidance, suggestions and constant encouragement and keen interest enriched throughout the course of project work.

We extend sincere thanks to the **HOD, Professor. N.Hari Krishna** for his kind co-operation in completing and making this project a success.

We extend sincere thanks to the **Principal, Prof. Dr.J.APPARAO** for his kind co-operation in completing and making this project a success.

We would like to thank the **Management** for their kind co-operation and for providing infrastructure facilities.

We extend thanks to all the **Teaching staff** of the Department of CSE-DS for their support and encouragement during the course of my project work. I also thank the **non-Teaching staff** of CSE Data Science department for being helpful in many ways in successful completion of my work.

Finally, we thank all those who helped me directly or indirectly in successful completion of this project work.

Sk.Shareefa (20KE1A4448)

Ch.Rukhmini (20KE1A4406)

J.Pravalika (20KE1A4417)

P.Asifa Kousar (20KE1A4440)

R.Poojitha (20KE1A4446)

ABSTRACT

online food ordering system designed to provide users with a convenient and efficient platform for ordering food from their favorite restaurants. This project employs a full-stack development approach, utilizing HTML, CSS, and React or Angular for the frontend, MongoDB for efficient database management, and Express, Node.js, and Python for a robust and dynamic backend. It offers features such as menu exploration, order customization, secure payments, and real-time order tracking, ensuring a seamless and enjoyable experience for users. The project's purpose is to develop software that will cut down on the time spent manually managing Item Category, Food, Customer, and Delivery Address. It saves the Delivery Address, Order, and Shopping Cart information.

INDEX

S.NO	CONTENTS	PAGE NO
1.	INTRODUCTION	1
2.	LITERATURE SURVEY	2
3.	FEASIBILITY STUDY	3
3.1	Market Analysis	
3.2	Technical Feasibility	
3.3	Financial Feasibility	
4	SYSTEM ANALYSIS	4-6
4.1.	Existing System	
4.1.1	Disadvantages of Existing System	
4.2	Proposed System	
4.2.1	Advantages of Proposed System	
4.3	Requirements Analysis	
4.3.1	Identify Stakeholders	
4.3.2	Gather Requirement	
4.3.3	Software Requirement	
4.3.4	Hardware Requirement	
4.4	Input and output Design	
4.4.1	Input Design	
	Search Functionality	
	Menu Exploration	
	Item Selection	
	Add to Cart	
4.4.2	Output Design	
4.5	System Requirement	
4.5.1	Hardware Requirements	
4.5.2	Software Requirements	
5.	SYSTEM DESIGN	7-12
5.1	Complete Visualization Of Online Food System	
5.2	System Design Model	
5.3	Admin Workflow Process	
5.4	Customer Workflow Process	
5.5	Use Case Diagram	
6.	SYSTEM ENVIRONMENT	13-21
6.1	Python	
6.2	Flak	
7.	IMPLEMENTATION	22
8.	SOURCE CODE	23-52
9.	TEST CASES	53
10.	SCREEN SHOTS	54-61
11.	CONCLUSION	62
12.	REFERENCES	63

LIST OF FIGURES

S.NO	FIGURE NAME	PAGE NO
1.	Online Food Ordering System	1
2.	Financial Feasibility	3
3.	Complete Visualization	7-8
4.	System Design Model	9
5.	Admin Workflow Process	10
6.	Customer Workflow Process	11
7.	Use Case Diagram	12
8.	Flask	22

1. INTRODUCTION

An online food ordering system is a digital platform that allows customers to browse menus, place orders, and make payments for food delivery or pickup from various restaurants. These systems typically offer features such as menu customization, order tracking, secure payment processing, and user reviews. They provide convenience for customers by eliminating the need for phone calls and physical menus, while also streamlining operations for restaurants by managing orders efficiently. Overall, online food ordering systems enhance the dining experience by providing a convenient and efficient way to order food from the comfort of one's home or office.

- 1. Wide Variety:** With online food ordering, you have access to a wide range of cuisines. Whether you're craving pizza, sushi, or Indian food, you can find it all in one place.
- 2. Convenience:** Ordering food online is incredibly convenient. You can place an order from anywhere, anytime, using your smartphone or computer. No need to wait in long lines or deal with busy phone lines.
- 3. Time-Saving:** With just a few clicks, customers can place orders quickly, saving time and effort compared to traditional methods of ordering food.
- 4. Data Insights:** These systems generate valuable data and insights on customer preferences, ordering patterns, and popular dishes, which restaurants can use to tailor their menus and marketing strategies.
- 5. User-Friendly Interface:** Online food ordering systems boast interfaces designed for seamless navigation, ensuring a hassle-free experience for customers as they peruse menus and finalize their orders.



Figure1: Online Food Ordering System

FURTHER ENHANCEMENT: In the future we would like to increase the accuracy and efficiency of our project by visualizing remaining errors and using further optimization techniques.

2.LITERATURE SURVEY

A literature survey on online food ordering systems would typically involve reviewing existing research, academic papers, industry reports, and relevant publications related to various aspects of online food ordering platforms.

Here's a structured approach you could follow:

1. **Convenience and User Experience:** Many studies have highlighted the convenience and positive user experience of online food ordering systems. Users appreciate the ease of browsing menus, placing orders, and tracking deliveries from the comfort of their own homes.
2. **Customer Satisfaction:** Research has shown that online food ordering systems contribute to higher customer satisfaction levels. The ability to customize orders, view ratings and reviews, and have clear communication with the restaurant enhances the overall dining experience.
3. **Challenges and Opportunities:** Literature also addresses the challenges and opportunities associated with online food ordering systems. These include issues like food quality control, delivery logistics, data security, and the potential for personalized recommendations and targeted marketing.
4. **Mobile Apps and Accessibility:** Many studies have focused on the rise of mobile apps for online food ordering. They highlight the importance of user-friendly interfaces, seamless navigation, and accessibility features to accommodate a wide range of users.
5. **Social Media Integration:** Researchers have explored the integration of social media platforms with online food ordering systems. They discuss how features like sharing reviews, photos, and recommendations on social media can enhance the visibility and reputation of restaurants.
6. **Sustainability and Green Practices:** A few studies have also touched upon the environmental impact of online food ordering systems. They discuss the potential for reducing food waste, optimizing delivery routes, and promoting sustainable packaging practices.
7. **Payment Systems and Security:** Review research on payment gateways, transaction processing, and security measures in online food ordering platforms.

3.FEASIBILITY STUDY

A feasibility study for an online food ordering system involves assessing the viability, potential risks, and benefits of implementing such a system.

Three key considerations involved in the feasibility analysis are:

1. **Market Analysis:**
2. **Technical Feasibility:**
3. **Financial Feasibility:**

1. Market Analysis:

- ◆ Identify the target market for the online food ordering system (e.g., local area, specific demographics).
- ◆ Analyze the size, growth trends, and competitive landscape of the online food delivery market.
- ◆ Determine the demand for online food ordering services and potential gaps or opportunities in the market.

Technical Feasibility:

- ◆ Evaluate the technical requirements for developing and maintaining the online food ordering platform (e.g., software development, hosting infrastructure).
- ◆ Assess the availability of suitable technology solutions and expertise to build and support the system.
- ◆ Consider compatibility with existing systems and integration with third-party services and systems they (e.g., payment gateways, delivery logistics).

3. Financial Feasibility:

- ◆ Estimate the initial investment required to develop the online food ordering platform and launch the service.
- ◆ Forecast the revenue potential based on market demand, pricing strategy, and projected sales volume.
- ◆ Conduct a cost-benefit analysis to determine the profitability and return on investment (ROI) of the project.



Figure 3: Financial Feasibility

4. SYSTEM ANALYSIS

4.1 Existing System:

Traditional food ordering methods often involve phone calls, physical menus, manual payment processes. The existing system review emphasizes the need for modernized and digital solution that simplifies the food ordering process, enhances user experience, and facilitates efficient transactions.

4.1.1 Dis Advantages of Existing System:

1. Past apps doesn't specify history.
2. Doesn't Specific amount calculation.
3. No Budget planning.

4.2 Proposed System:

◆ It proposes an innovative solution that leverages full- stack technologies to streamline the online food ordering experience. The frontend, developed with HTML, CSS, and React or Angular, offers an intuitive and visually appealing interface. The backend, built on Express, Node.js and Python, integrates features such as menu display, order processing, payment handling, and real time updates for and enhanced user experience.

4.2.1 Advantages of Proposed System:

1. To overcome all those disadvantages we want include these things:
2. Clear detailing history.
3. Maintaining budget per month.
4. Amount calculation as per our food orders.
5. Challenges for delivery people.

4.3 Requirement Analysis:

Requirement analysis is a crucial step in the development of any software system, including an online food ordering system. This process involves gathering, documenting, and analyzing the needs and expectations of stakeholders to define the features and functionality of the system.

Here's a breakdown of the requirement analysis process for an online food ordering system:

4.3.1 Identify Stakeholders

4.3.2 Gather Requirement

4.3.3 Software Requirement

4.3.4 Hardware Requirement

Identify Stakeholders: Identify all the stakeholders involved in the system, including customers, restaurant owners, delivery personnel, and administrators.

Gather Requirement: Conduct interviews, surveys, and workshop with stakeholders to understand their needs and expectations. Gather both functional and non-functional requirements. Functional requirements define what the system should do, while non-functional requirements define qualities such

as performance, security, and usability.

Software Requirement:

Python

Operating Systems supported:

Windows 10 64bit OS

Hardware Requirement Tools:

Processor: Intel i9

RAM: 32 GB

Space on Hard Disk: minimum 1 TB

4.4 Input And Output Design:

4.4.1 Input Design: When designing the input for an online food ordering system, it's important to make it user-friendly and intuitive. Here are some key considerations for input design.

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

Here are some key considerations for input design:

Search Functionality: Provide a search bar where users can easily find specific and special dishes.

Menu Navigation: Organize the menu in a clear and logical manner, with categories subcategories if necessary. Use visual cues like icons or images to enhance the user experience.

Item Selection: Allow users to easily select items from the menu, including options for quantity, size and any customizations.

Add to Cart: Provide a prominent "Add to Cart" button for each selected item, allowing users to the easily add multiple items before proceeding to checkout.

4.4.2 Output Design:

After the user selects their desired items and proceeds to the checkout, the system would generate an order confirmation page. This page would display a summary of the items ordered, including the quantity, name, and price of each item. It may also include any customization options selected by the user.

Below the order summary, there would be a section for the user enter their delivery address. Next, the system would ask for the user's contact information, such as their name, phone number, and email and address. This information is necessary for the restaurant to contact the user if needed, and for the delivery person to locate the customer.

4.5 System Requirements:

4.5.1 Hardware Requirements:

High-Performance Server for Hosting.

System : Intel Core i5.

Hard Disk : 1TB.

ROM : Min .500GB SSD

Ram : 16GB+

Internet adapter : 1 Gbps

4.5.2 Software Requirements:

Frontend Framework : HTML & CSS & React

Backend Framework : Express.js, Node.js, python

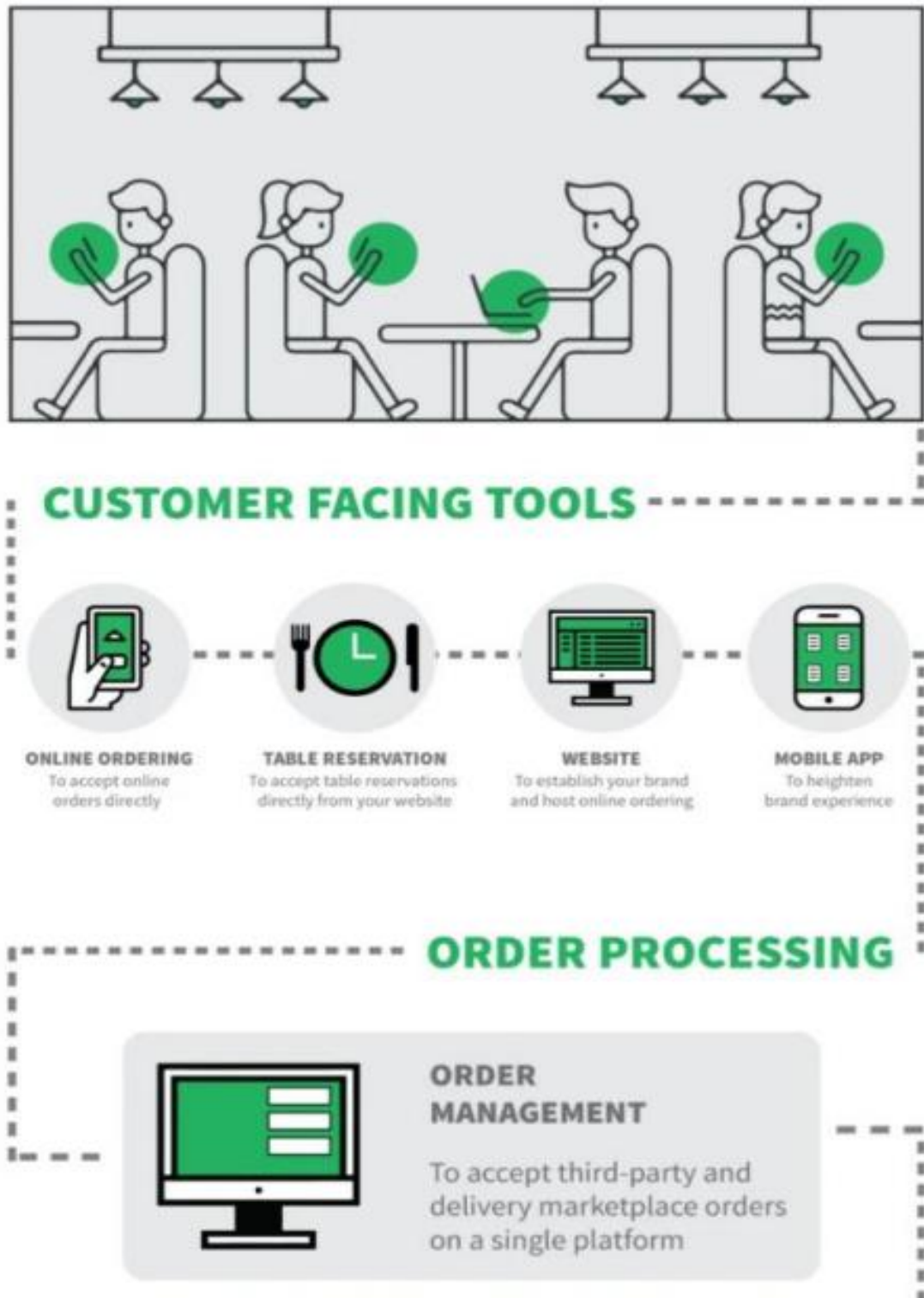
Mongo DB : Database Management

Operating System : Windows 10

Tool : Visual Studio Code

5.SYSTEM DESIGN

5.1 Complete Visualization of Online Food Ordering System:



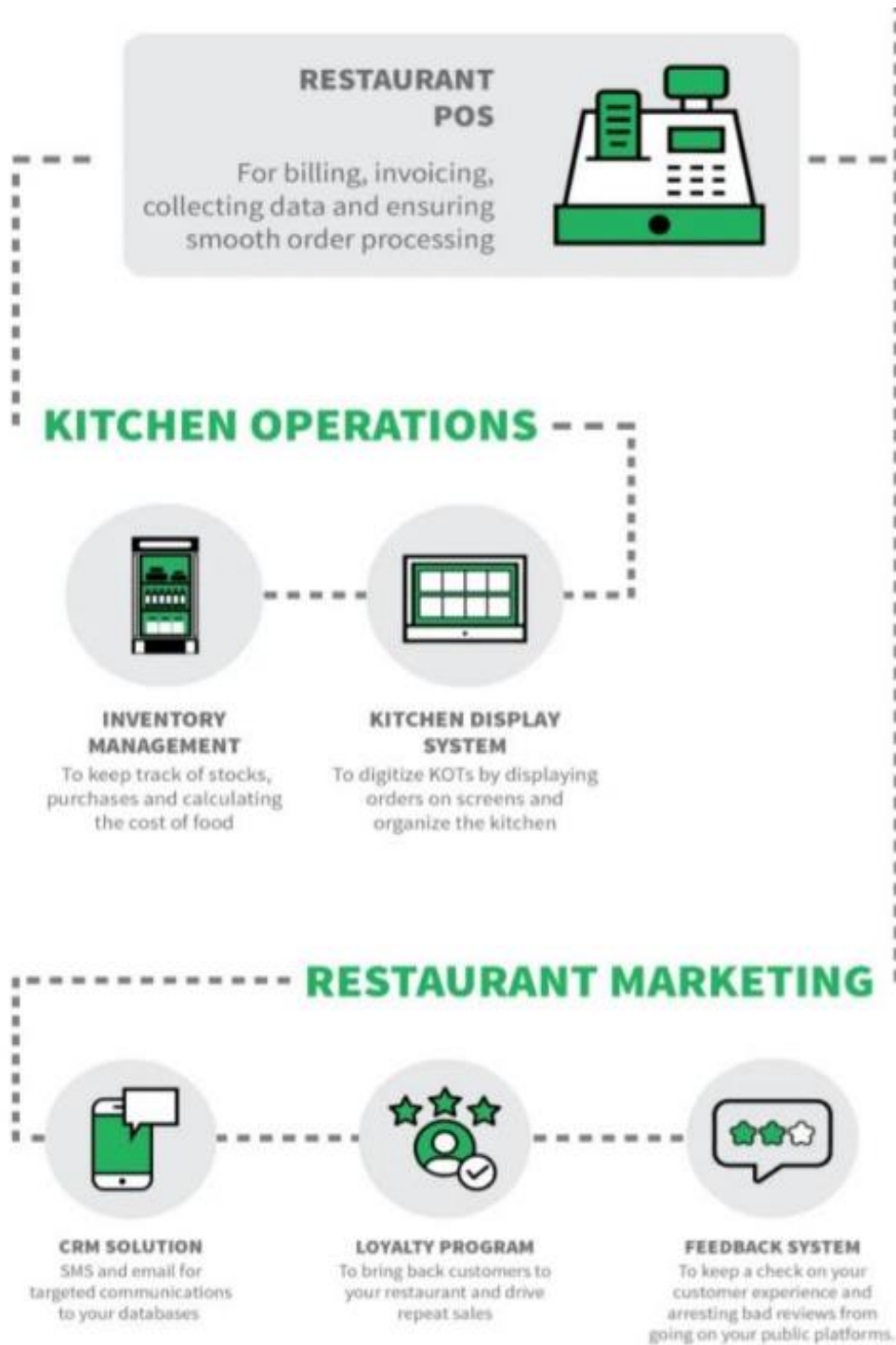


Figure 5.1: Complete Visualization

5.2 System Design Model:



Figure 5.2: System Design Model

5.3 Admin workflow Process: User goes to home page of the domain. If he/she has an account then he/she can login in restaurant management system otherwise he/she need an account after successful registration, they can login in home page.

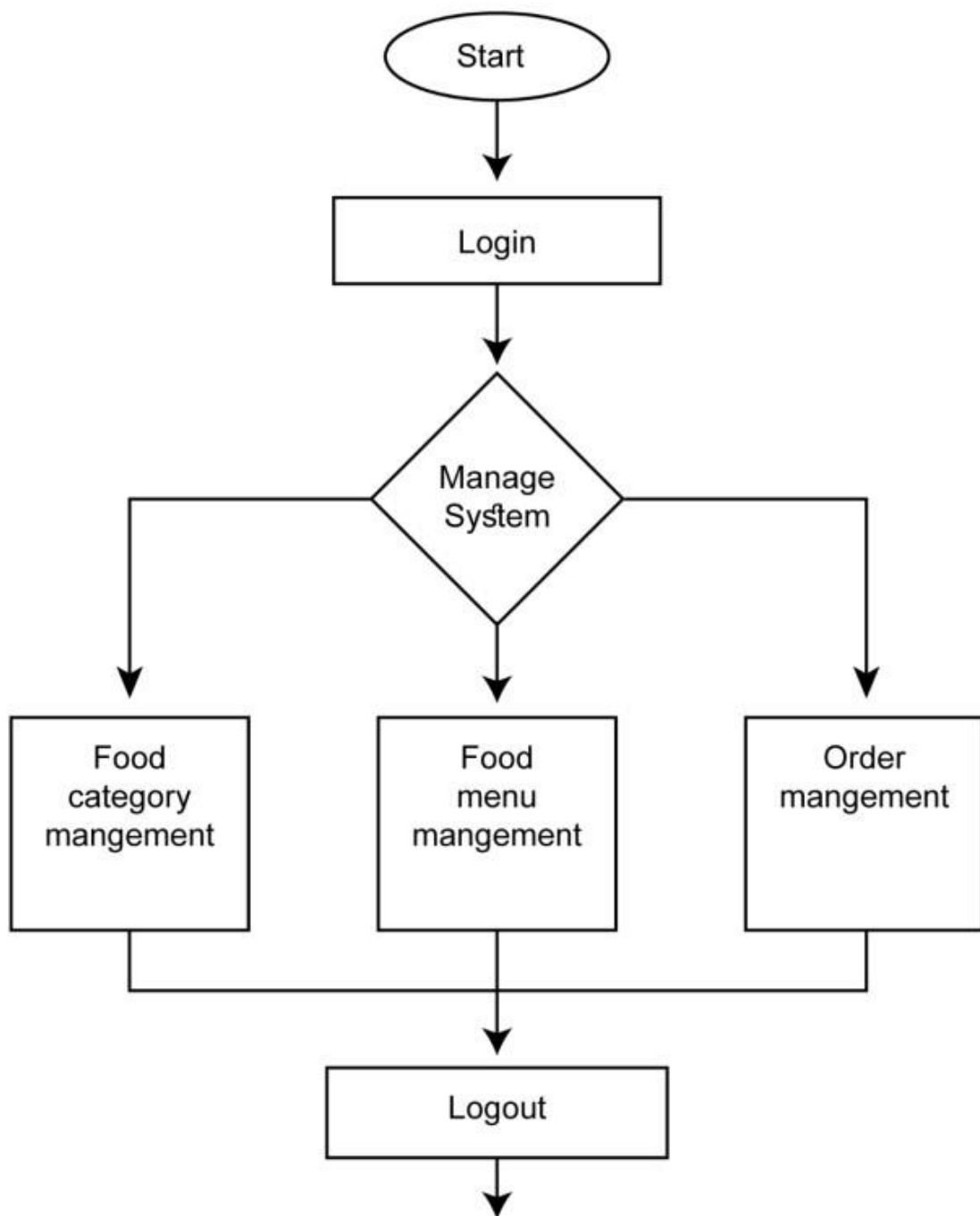


Figure 5.3: User Workflow Process

5.4 Customer Workflow Process: Initially to visit the food categories or food menu, users don't need to login/register an account. After checking out the categories and menu items,

if the user finds his/her desired menu and if they want to order that particular item they can go to order page. During placing any order the customer needs to provide his/her required information mentioned the order section.

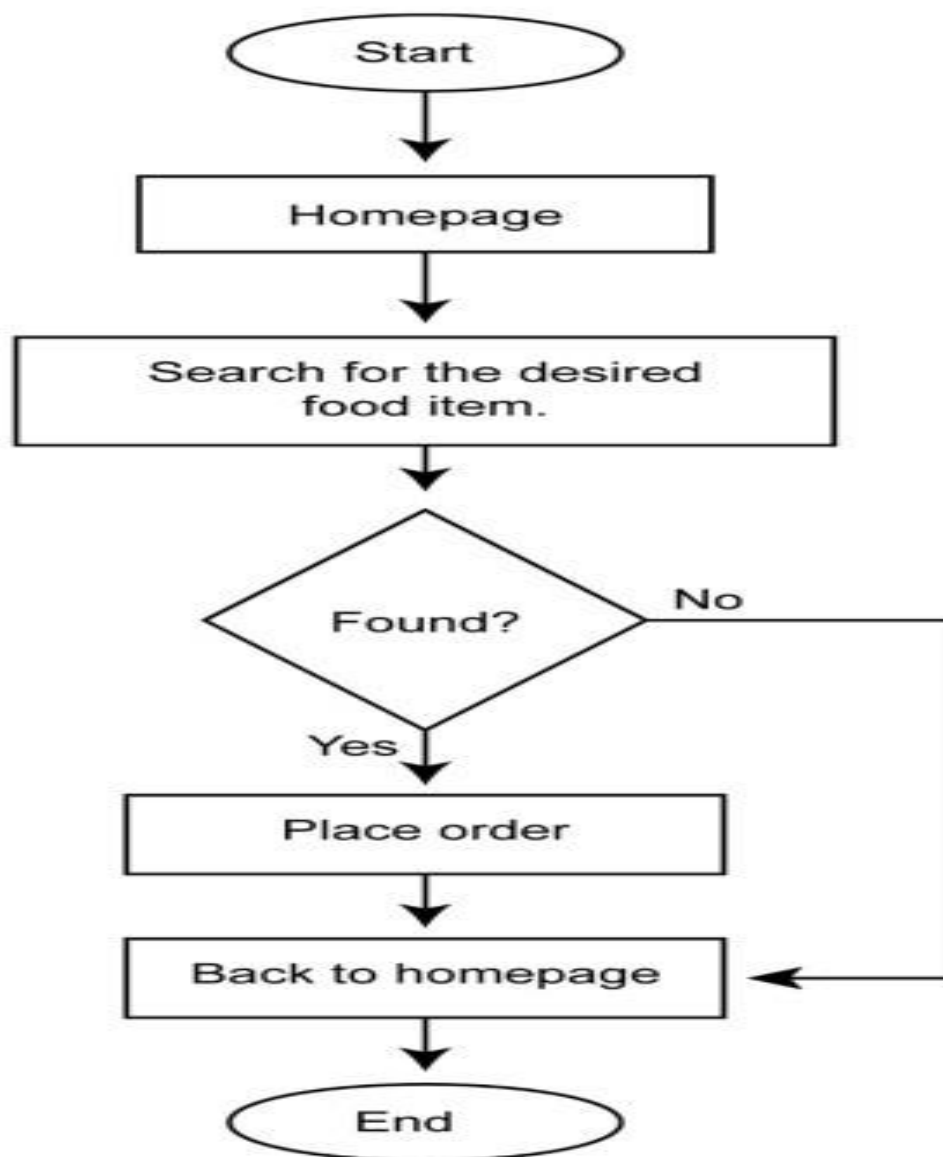


Figure 5.4: Customer Workflow Process

5.5 Use Case Diagram: A use case diagram for an online food ordering system outlines the different interactions between actors (users) and the system itself. Here's a basic example of a use case diagram for an online food ordering system:

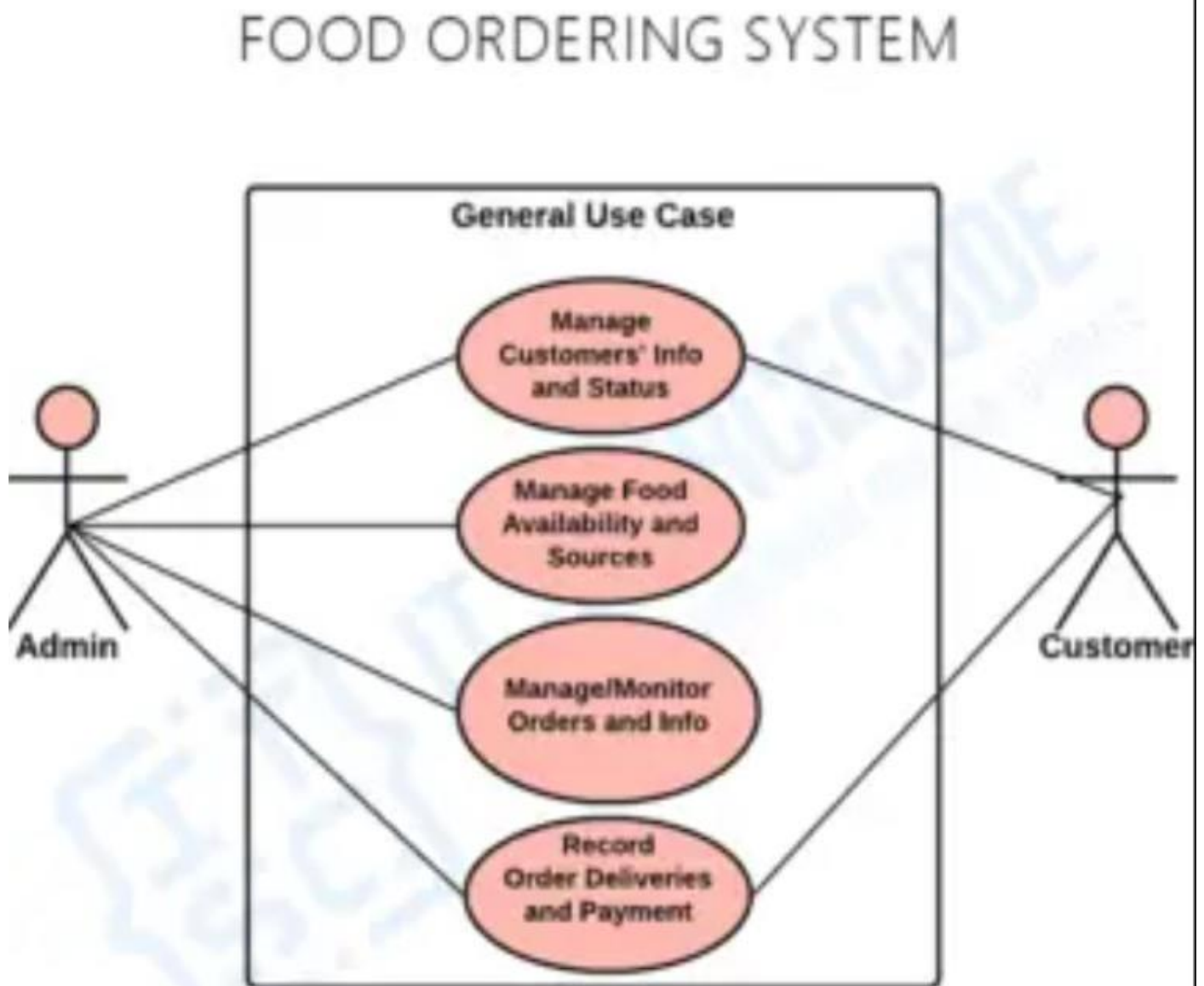


Figure 5.5: Use Case Diagram

6.SOFTWARE ENVIRNOMENT

6.1 PYTHON

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. An interpreted language, Python has a design philosophy emphasizes code readability (notably using whitespace indentation to delimit code blocks rather than curly brackets or keywords), and a syntax that allows programmers to express concepts in fewer lines of code than might be used in languages such as C++or Java. It provides constructs that enable clear programming on both small and large scales. C, Python, the reference implementation of Python, is open source software and has a community-based development model, nearly all of its variant implementations. It supports multiple programming paradigms, including object oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

Interactive Mode Programming

Invoking the interpreter without passing a script file as a parameter brings up the following prompt – \$
python

```
Python 2.8.2 (#1, Nov 25 2020, 13:34:43)
[GCC 4.1.2 20080704 (Red Hat 4.1.2-48)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Type the following text at the Python prompt and press the Enter –

```
>>> print "Hello, Python!"
```

If you are running new version of Python, then you would need to use print statement with parenthesis as in print ("Hello, Python!");. However, in Python version 2.8.2, this produces the following result – Hello, Python!

Script Mode Programming

Invoking the interpreter with a script parameter begins execution of the script and continues until the script is finished. When the script is finished, the interpreter is no longer active.

Let us write a simple Python program in a script. Python files have extension .py. Type the following source code in a test.py file –

```
Live    Demo    print
```

```
"Hello, Python!"
```

We assume that you have Python interpreter set in PATH variable. Now, try to run this program as follows –

```
$ python test.py
```

This produces the following result –

```
Hello, Python!
```

Let us try another way to execute a Python script. Here is the modified test.py file – Live

```
Demo
```

```
#!/user/bin/python print
```

```
"Hello, Python!"
```

We assume that you have Python interpreter available in /usr/bin directory. Now, try to run this program as follows –

```
$ chmod +x test.py    # This is to make file executable
```

```
$. /test.py
```

This produces the following result – Hello,

Python!

Python Identifiers

A Python identifier is a name used to identify a variable, function, class, module or other object. An identifier starts with a letter A to Z or a to z or an underscore (_) followed by zero or more letters, underscores and digits (0 to 9).

Python does not allow punctuation characters such as @, \$, and % within identifiers. Python is a case sensitive programming language. Thus, Manpower and manpower are two different identifiers in Python.

Here are naming conventions for Python identifiers –

Class names start with an uppercase letter. All other identifiers start with a lowercase letter.

Starting an identifier with a single leading underscore indicates that the identifier is private.

Starting an identifier with two leading underscores indicates a strongly private identifier.

If the identifier also ends with two trailing underscores, the identifier is a language-defined special name.

Reserved Words

These are reserved words and you cannot use them as constant or variable or any other identifier names.

All the Python keywords contain lowercase letters only.

Lines and Indentation

Python provides no braces to indicate blocks of code for class and function definitions or flow control.

Blocks of code are denoted by line indentation, which is rigidly enforced.

The number of spaces in the indentation is variable, but all statements within the block must be indented the same amount. For example – if True: print "True"

```
else:
```

```
    print "False"
```

However, the following block generates an error – if

```
True:
```

```
print "Answer"
```

```
print    "True"
```

```
else:
```

```
print "Answer" print
```

```
"False"
```

Thus, in Python all the continuous lines indented with same number of spaces would form a block. The following example has various statement blocks –

Note – Do not try to understand the logic at this point of time. Just make sure you understood various blocks even if they are without braces.

```
#!/usr/bin/python
```

```
import sys try:
```

```

# Open file stream file =
open (file_name, "w")
except IOError:
    print "There was an error writing to", file_name
    sys.exit()
print "Enter '", file_finish, print "'
When finished" while file_text !=
file_finish: file_text =
raw_input("Enter text: ") if file_text
== file_finish:
    # close the file
    file.close break
    file.write(file_text)
    file.write("\n")
file.close() file_name = raw_input("Enter
filename: ") if len(file_name) == 0:
    print "Next time please enter something"
    sys.exit()
try:
    file = open(file_name, "r")
except IOError:
    print "There was an error reading file"
    sys.exit()
file_text = file.read()
file.close() print
file_text

```

Multi-Line Statements

Statements in Python typically end with a new line. Python does, however, allow the use of the line continuation character (\) to denote that the line should continue. For example – total = item_one + \ item_two + \ item_three

Statements contained within the [], {}, or () brackets do not need to use the line continuation character. For example – days = ['Monday',

```

'Tuesday', 'Wednesday', 'Thursday',
'Friday']

```

Quotation in Python

Python accepts single ('), double (") and triple (" or """) quotes to denote string literals, as long as the same type of quote starts and ends the string.

The triple quotes are used to span the string across multiple lines. For example, all the following are legal – word = 'word' sentence = "This is a sentence." paragraph = """This is a paragraph. It is made up of multiple lines and sentences."""

Comments in Python

A hash sign (#) that is not inside a string literal begins a comment. All characters after the # and up to the end of the physical line are part of the comment and the Python interpreter ignores them.

Live Demo

```
#!/usr/bin/python #
```

First comment

```
print "Hello, Python!" # second comment
```

This produces the following result – Hello,

Python!

You can type a comment on the same line after a statement or expression –

```
name = "Madisetti" # This is again comment
```

You can comment multiple lines as follows –

```
# This is a comment.
```

```
# This is a comment, too.
```

```
# This is a comment, too.
```

```
# I said that already.
```

Following triple-quoted string is also ignored by Python interpreter and can be used as a multiline-comments:

```
"""
```

```
This is a multiline comment.
```

```
"""
```

Using Blank Lines

A line containing only whitespace, possibly with a comment, is known as a blank line and Python totally ignores it.

In an interactive interpreter session, you must enter an empty physical line to terminate a multiline statement.

Waiting for the User

The following line of the program displays the prompt, the statement saying “Press the enter key to exit”, and waits for the user to take action –

```
#!/usr/bin/python
```

```
raw_input("\n\nPress the enter key to exit.")
```

Here, "\n\n" is used to create two new lines before displaying the actual line. Once the user presses the key, the program ends. This is a keep a console window open until the user is done with an application.

Multiple Statements on a Single Line

The semicolon (;) allows multiple statements on the single line given that neither statement starts a new code block. Here is a sample snip using the semicolon.

```
import sys; x = 'foo'; sys.stdout.write(x + '\n')
```

Multiple Statement Groups as Suites

A group of individual statements, which make a single code block are called suites in Python. Compound or complex statements, such as if, while, def, and class require a header line and a suite.

Header lines begin the statement (with the keyword) and terminate with a colon (:) and are followed by one or more lines which make up the suite. For example – if expression: suite elif expression:

```
    suite
else:
    suite
```

Command Line Arguments

Many programs can be run to provide you with some basic information about how they should be run. Python enables you to do this with -h –

```
$ python -h usage: python [option] ... [-c cmd | -m mod | file | -] [arg]
```

... Options and arguments (and corresponding environment variables):

-c cmd: program passed in as string (terminates option list)

-d : debug output from parser (also PYTHONDEBUG=x)

-E: ignore environment variables (such as PYTHONPATH)

-h: print this help message and exit

You can also program your script in such a way that it should accept various options. Command Line Arguments is an advanced topic and should be studied a bit later once you have gone through rest of the Python concepts.

Python Lists

The list is a most versatile datatype available in Python which can be written as a list of comma-separated values (items) between square brackets. Important thing about a list is that items in a list need not be of the same type.

Creating a list is as simple as putting different comma-separated values between square brackets. For example – list1 = ['physics', 'chemistry', 1997, 2000]; list2 = [1, 2, 3, 4, 5]; list3 = ["a", "b", "c", "d"]

Similar to string indices, list indices start at 0, and lists can be sliced, concatenated and so on.

A tuple is a sequence of immutable Python objects. Tuples are sequences, just like lists. The differences between tuples and lists are, the tuples cannot be changed unlike lists and tuples use parentheses, whereas lists use square brackets.

Creating a tuple is as simple as putting different comma-separated values. Optionally you can put these comma-separated values between parentheses also. For example – tup1 = ('physics', 'chemistry', 1997, 2000); tup2 = (1, 2, 3, 4, 5); tup3 = "a", "b", "c", "d";

The empty tuple is written as two parentheses containing nothing – tup1

= (0);

To write a tuple containing a single value you have to include a comma, even though there is only one value – `tup1 = (50,);`

Like string indices, tuple indices start at 0, and they can be sliced, concatenated, and so on.

Accessing Values in Tuples

To access values in tuple, use the square brackets for slicing along with the index or indices to obtain value available at that index. For example –

Live Demo

```
#!/usr/bin/python tup1 = ('physics',  
'chemistry', 1997, 2000); tup2 = (1, 2, 3,  
4, 5, 6, 7); print "tup1[0]: ", tup1[0]; print  
"tup2[1:5]: ", tup2[1:5];
```

When the above code is executed, it produces the following result

– `tup1[0]: physics tup2[1:5]: [2, 3, 4, 5]`

Updating Tuples

Accessing Values in Dictionary

To access dictionary elements, you can use the familiar square brackets along with the key to obtain its value. Following is a simple example –

Live Demo

```
#!/usr/bin/python  
dict = {'Name': 'Zara', 'Age': 7, 'Class':  
'First'} print "dict['Name']: ", dict['Name']  
print "dict['Age']: ", dict['Age']
```

When the above code is executed, it produces the following result

– `dict['Name']: Zara dict['Age']: 7`

If we attempt to access a data item with a key, which is not part of the dictionary, we get an error as follows – Live Demo

```
#!/usr/bin/python dict = {'Name': 'Zara',  
'Age': 7, 'Class': 'First'} print "dict['Alice']: ",  
dict['Alice']
```

When the above code is executed, it produces the following result –

`dict['Alice']:`

Traceback (most recent call last):

File "test.py", line 4, in <module> print

```
"dict['Alice']: ", dict['Alice'];
```

KeyError: 'Alice'

Updating Dictionary

You can update a dictionary by adding a new entry or a key-value pair, modifying an existing entry, or deleting an existing entry as shown below in the simple example –

Live Demo

```
#!/usr/bin/python dict = {'Name': 'Zara', 'Age':  
7, 'Class': 'First'} dict['Age'] = 8; # update  
existing entry dict['School'] = "DPS School"; #  
Add new entry print "dict['Age']: ", dict['Age']  
print "dict['School']: ", dict['School']
```

When the above code is executed, it produces the following result –

```
dict['Age']: 8
```

```
dict['School']: DPS School
```

Delete Dictionary Elements

You can either remove individual dictionary elements or clear the entire contents of a dictionary. You can also delete entire dictionary in a single operation.

To explicitly remove an entire dictionary, just use the del statement. Following is a simple example –

Live Demo

```
#!/usr/bin/python dict = {'Name': 'Zara', 'Age': 7,  
'Class': 'First'} del dict['Name']; # remove entry  
with key 'Name' dict.clear(); # remove all entries  
in dict del dict ; # delete entire dictionary print  
"dict['Age']: ", dict['Age'] print "dict['School']: ",  
dict['School']
```

This produces the following result. Note that an exception is raised because after del dict dictionary does not exist anymore –

```
dict['Age']:
```

Traceback (most recent call last):

File "test.py", line 8, in <module> print

```
"dict['Age']: ", dict['Age'];
```

TypeError: 'type' object is unsubscriptable

Note – del () method is discussed in subsequent section.

Properties of Dictionary Keys

Dictionary values have no restrictions. They can be any arbitrary Python object, either standard objects or user-defined objects. However, same is not true for the keys.

There are two important points to remember about dictionary keys –

(a) More than one entry per key not allowed. Which means no duplicate key is allowed. When duplicate keys encountered during assignment, the last assignment wins. For example –

Live Demo

```
#!/usr/bin/python dict = {'Name': 'Zara', 'Age':  
7, 'Name': 'Manni'} print "dict['Name']: ",  
dict['Name']
```

When the above code is executed, it produces the following result –

dict['Name']: Manni

(b) Keys must be immutable. Which means you can use strings, numbers or tuples as dictionary keys but something like ['key'] is not allowed. Following is a simple example –

Live Demo

```
dict = {'Name': 'Zara', 'Age': 7}  
print "dict['Name']: ", dict['Name']
```

When the above code is executed, it produces the following result –

Traceback (most recent call last):

```
File "test.py", line 3, in <module> dict  
= {'Name': 'Zara', 'Age': 7};
```

TypeError: unhashable type: 'list'

Tuples are immutable which means you cannot update or change the values of tuple elements. You are able to take portions of existing tuples to create new tuples as the following example demonstrates

–Live

Demo

```
#!/usr/bin/python tup1  
= (12, 34.56); tup2 =  
( 'abc', 'xyz');  
# tup1[0] = 100;  
# So let's create a new tuple as  
follows tup3 = tup1 + tup2; print  
tup3;
```

When the above code is executed, it produces the following result –

(12, 34.56, 'abc', 'xyz')

Delete Tuple Elements

Removing individual tuple elements is not possible. There is, of course, nothing wrong with putting together another tuple with the undesired elements discarded.

To explicitly remove an entire tuple, just use the del statement. For example – Live

Demo

```
#!/usr/bin/python tup = ('physics',  
'chemistry', 1997, 2000); print tup;  
tup;  
print "After deleting tup: ";  
print tup;
```

This produces the following result. Note an exception raised, this is because after del tup tuple does not exist anymore –

```
('physics', 'chemistry', 1997, 2000)
```

After deleting tup:

Traceback (most recent call last):

File "test.py", line 9, in <module>

```
print tup;
```

NameError: name 'tup' is not defined

6.2 FLASK

Flask is a Python framework for building web apps. It's small, light and simple compared with the other widely used Python framework.

Setup for Flask

We will create a new folder with **a new virtual environment** for Flask work projects. Various new modules will be installed here.

Create a directory and change into it

The first step is to create a new folder (directory) for all your Flask projects. Mine is here: Documents/python/flask

Change into that directory. For me, the command would be: `cd Documents/python/flask`



7.IMPLEMENTAION

Module Description:

User Management: Allows users to create accounts, log in, and manage their profiles, including saved addresses and payment methods.

Restaurant Management: Enables restaurants to register, add menu items, set prices, and manage orders.

Menu Management: Allows restaurants to create and update their menus, including adding descriptions and images for each item.

Order Management: Handles the process of placing and managing orders, including order tracking and notifications.

Payment Gateway Integration: Integrates with a payment gateway to process payments securely.

Search and Filtering: Enables users to search for restaurants and menu items based on various criteria such as cuisine type, price range, and location.

Cart Management: Manages the items added to the cart before checkout, including quantity adjustments and item removal.

Checkout Process: Guides users through the checkout process, including selecting delivery or pickup options and entering payment information.

8. SOURCE CODE

Create a Project:

First open VScode, and we have to create a folder with any name you like. In VScode first we have to complete the frontend part and backend part like we have to create login page, registration page, home page,like that.

First our project starts with get started page like.

GET STARTED PAGE:

```
<html>
  <head>

    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Navigation Box</title>
    <style>
      body {
        background-image:
url('https://static.wixstatic.com/media/04d06d_1786b3bc2168490f991fb23aeaf8e72d~mv2.jpg/v1/fill/w_864,h_486,al_c,q_85,usm_0.66_1.00_0.01,enc_auto/04d06d_1786b3bc2168490f991fb23aeaf8e72d~mv2.jpg'); /*
Provide the URL for your background image */
        background-size: cover;
        font-family: Arial, sans-serif;
        margin: 0;
        padding: 0;
        color: white;
      }</style>
    </head>
    <center><body>
      <a href="/h">

        <button type="button" style="width: 120px;height:40px;margin-top:300px;background-color:rgb(64, 201, 85)">Get started!</button>
      </a>
    </body></center>
  </html>
```

REGISTRATION PAGE:

```
<!DOCTYPE html>
<html>
  <head lang="en">

    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Navigation Box</title>
    <style>
      body {
        background-image: url('https://cdn.openart.ai/uploads/image_dDGRyRXa_1695812593344_raw.jpg'); /*
Provide the URL for your background image */
        background-size: cover;
        font-family: Arial, sans-serif;
```

```

        margin: 0;
        padding: 0;
    }</style>
</head>
<head>
    <title>Registration Page</title>
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
</head>
<body style="background-color: pink">
    <h3><div class="container mt-5">
        <div class="card">
            <div class="card-body">
                <h2 class="card-title">Registration Form</h2>
                <center><p>{{ status }}</p></center>
                <form action="/register" method="post">
                    <div class="form-group">
                        <label for="name">User Name:</label>
                        <input type="text" class="form-control" id="name" name="name" placeholder="Username">
                    </div>
                    <div class="form-group">
                        <label for="pass">Password:</label>
                        <input type="password" class="form-control" id="pass" name="pass" placeholder="Enter your
password">
                    </div>
                    <div class="form-group">
                        <label for="repass">Re-type password:</label>
                        <input type="password" class="form-control" id="repass" name="repass" placeholder="Retype
your password">
                    </div>
                    <div class="form-group">
                        <label for="phone">Phone:</label>
                        <input type="text" class="form-control" id="phone" name="phone" placeholder="Enter your
number">
                    </div>
                    <button type="submit" class="btn btn-primary">Submit</button>
                    <button type="reset" class="btn btn-secondary">Reset</button>
                </form>
            </div>
        </div>
    </div></h3>
</body>
</html>

```

LOGIN PAGE:

```

<html lang="en">
<head>

    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Navigation Box</title>
    <style>
        body {
            background-image: url('https://img.freepik.com/premium-photo/plate-food-with-chicken-rice-biryani-

```

```

with-fire-smoke-background_885092-227.jpg'); /* Provide the URL for your background image */
    background-size: cover;
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0
    color: white;
} </style>
</head>
<body bgcolor="pink">
    <center> <h1> LOGIN </h1> </center>
    <center><p>{{ status }} </p></center>
    <form action="/login" method="post">
        <center><h1><div class="container">
            <label>Username : </label>
            <input type="text" placeholder="Enter Username" name="username" required> <br>
            <label>Password : </label>
            <input type="password" placeholder="Enter Password" name="password" required><br>
            <input type="submit">
        </div> </h1> </center>
    </form>
    <center>
        
    </center>
</body>
</html>

```

HOME PAGE:

```

<html lang="en">
<head>

    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Navigation Box</title>
    <style>
        body {
            background-image: url('https://img.freepik.com/premium-photo/plate-food-with-chicken-rice-biryani-
with-fire-smoke-background_885092-227.jpg'); /* Provide the URL for your background image */
            background-size: cover;
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
            color: white;
        } </style>
    </head>
    <body bgcolor="pink">
        <center> <h1> LOGIN </h1> </center>
        <form action="/login" method="post">
            <center><h1><div class="container">
                <label>Username : </label>
                <input type="text" placeholder="Enter Username" name="username" required> <br>
                <label>Password : </label>

```



```

        <input type="password" placeholder="Enter Password" name="password" required><br>
        <input type="submit">
    </div> </h1> </center>
</form>
<center>
    
    </center>
</body>
</html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta
        name="viewport"
        content="width=device-width, initial-scale=1.0,viewport-fit=cover"
    />
    <title>Food Website</title>
    <link rel="stylesheet" href="{ {url_for('static',filename='index.css')}}" />
    <link
        rel="stylesheet"
        href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css"
    />
    <script>
        var data = { {dishes | tojson} }
        var sea;
        function s(){
            var inp=document.getElementById('search')
            var p=document.getElementById('sea')
            var res= document.getElementById('results')
            var val = inp.value
            if (val==="){
                p.innerText=""
                res.innerHTML="<p></p>"
                return
            }
            else{
                var str=""
                for (let i in data){
                    p.innerText="Search Results"

                    if(data[i]['name'].toLowerCase().includes(val.toLowerCase())){
                        console.log(data[i])
                        str+=` <a href=/addToCart?id=${data[i]['id']}><div id='item-card'><div id='card-top'><i class='fa fa-
star' id='rating'> ${data[i]['rating']}</i><i class='fa fa-cart-plus add-to-cart' id='1'></i></div><img
src=${data[i]['img']} /><p id='item-name'>${data[i]['name']}</p><p id='item-price'>Price : Rs.
${data[i]['price']}</p></div></a>`
                    }
                }
                res.innerHTML=str
            }
        }
    </script>

```

```

    }
  }
</script>
</head>
<body data-new-gr-c-s-check-loaded="14.1060.0" data-gr-ext-installed="">
  <!-- desktop view -->
  <div class="container" id="container">
    <div id="menu">
      <div class="title">
        
      </div>
      <div class="menu-item">
        <a href="/about">About</a>
        <a href="/services">Services</a>
        <a href="/account">Your Orders</a>
        <a href="/cart">Cart</a>
        <a href="/contact">Contact</a>
      </div>
    </div>

    <div id="food-container">
      <div id="header">

        <div class="util">
          <i class="fa fa-search" onclick="s()"><input type="text" id="search" style="margin: 3px;border:1px
solid red;border-radius:5px"> Search</i>
          <i class="fa fa-tags"> <a href="/offers" style="text-decoration: none;">Offers</a></i>
          <a href="/cart"><i class="fa fa-cart-plus" id="cart-plus"></i></a>
        </div>
      </div>
      <div id="food-items" class="d-food-items">

        <div id="search">
          <p id="sea"></p>
          <div id="results">

            </div>
          </div>

        <div id="biryani" class="d-biryani">
          <p id="category-name">Biryani</p>
          { % for i in dishes % }
          { % if i["category"] == "biryani" % }
          <a href="/addToCart?id={{ i['id'] }}">
            <div id="item-card">
              <div id="card-top">
                <i class="fa fa-star" id="rating"> {{ i['rating'] }} </i>
                <i class="fa fa-cart-plus add-to-cart" id="1"></i>
              </div>
              <img src={{ '/image?file=' + i['img'] }} />
              <p id="item-name">{{ i['name'] }}</p>
              <p id="item-price">Price : Rs. {{ i['price'] }}</p>
            </div>
          </a>
        </div>
      </div>
    </div>
  </body>
</html>

```

```

</a>
{% endif %}
{% endfor %}
</div>

<div id="chicken" class="d-chicken">
  <p id="category-name">Chicken Delicious</p>
  {% for i in dishes %}
  {% if i["category"] == "chicken" %}
  <a href="/addToCart?id={{ i['id'] }}">
    <div id="item-card">
      <div id="card-top">
        <i class="fa fa-star" id="rating"> {{ i['rating'] }}</i>
        ><i class="fa fa-cart-plus add-to-cart" id="1"></i>
      </div>
      <img src={{ '/image?file=' + i['img'] }} />
      <p id="item-name">{{ i['name'] }}</p>
      <p id="item-price">Price : Rs. {{ i['price'] }}</p>
    </div>
  </a>
  {% endif %}
  {% endfor %}
</div>

<div id="paneer" class="d-paneer">
  <p id="category-name">Paneer Mania</p>
  {% for i in dishes %}
  {% if i["category"] == "paneer" %}
  <a href="/addToCart?id={{ i['id'] }}">
    <div id="item-card">
      <div id="card-top">
        <i class="fa fa-star" id="rating"> {{ i['rating'] }}</i>
        ><i class="fa fa-cart-plus add-to-cart" id="1"></i>
      </div>
      <img src={{ '/image?file=' + i['img'] }} />
      <p id="item-name">{{ i['name'] }}</p>
      <p id="item-price">Price : Rs. {{ i['price'] }}</p>
    </div>
  </a>
  {% endif %}
  {% endfor %}
</div>

<div id="vegetable" class="d-vegetable">
  <p id="category-name">Pure Veg Dishes</p>
  {% for i in dishes %}
  {% if i["category"] == "vegetable" %}
  <a href="/addToCart?id={{ i['id'] }}">
    <div id="item-card">
      <div id="card-top">
        <i class="fa fa-star" id="rating"> {{ i['rating'] }}</i>
        ><i class="fa fa-cart-plus add-to-cart" id="1"></i>
      </div>

```

```

    <img src={{ '/image?file=' +i['img'] }} />
    <p id="item-name">{{ i['name'] }}</p>
    <p id="item-price">Price : Rs. {{ i['price'] }}</p>
  </div>
</a>
{% endif %}
{% endfor %}
</div>

<div id="chinese" class="d-chinese">
  <p id="category-name">Chinese Corner</p>
  {% for i in dishes %}
  {% if i["category"] == "chinese" %}
  <a href="/addToCart?id={{ i['id'] }}">
    <div id="item-card">
      <div id="card-top">
        <i class="fa fa-star" id="rating"> {{ i['rating'] }}</i>
        <i class="fa fa-cart-plus add-to-cart" id="1"></i>
      </div>
      <img src={{ '/image?file=' +i['img'] }} />
      <p id="item-name">{{ i['name'] }}</p>
      <p id="item-price">Price : Rs. {{ i['price'] }}</p>
    </div>
  </a>
  {% endif %}
  {% endfor %}
</div>

<div id="south-indian" class="d-south-indian">
  <p id="category-name">South Indian</p>
  {% for i in dishes %}
  {% if i["category"] == "south indian" %}
  <a href="/addToCart?id={{ i['id'] }}">
    <div id="item-card">
      <div id="card-top">
        <i class="fa fa-star" id="rating"> {{ i['rating'] }}</i>
        <i class="fa fa-cart-plus add-to-cart" id="1"></i>
      </div>
      <img src={{ '/image?file=' +i['img'] }} />
      <p id="item-name">{{ i['name'] }}</p>
      <p id="item-price">Price : Rs. {{ i['price'] }}</p>
    </div>
  </a>
  {% endif %}
  {% endfor %}
</div>

<div id="ice-cream" class="d-ice-cream">
  <p id="category-name">ice-cream</p>
  {% for i in dishes %}
  {% if i["category"] == "ice-cream" %}
  <a href="/addToCart?id={{ i['id'] }}">
    <div id="item-card">
      <div id="card-top">

```

```

        <i class="fa fa-star" id="rating"> {{i['rating']}}</i>
        ><i class="fa fa-cart-plus add-to-cart" id="1"></i>
    </div>
    <img src={{ '/image?file=' +i['img'] }} />
    <p id="item-name">{{i['name']}}</p>
    <p id="item-price">Price : Rs. {{i['price']}}</p>
</div>
</a>
{% endif %}
{% endfor %}
</div>
<div id="Dessert" class="d-Dessert">
    <p id="category-name">Dessert</p>
    {% for i in dishes %}
    {% if i["category"] == "Dessert" %}
    <a href="/addToCart?id={{i['id']}}">
        <div id="item-card">
            <div id="card-top">
                <i class="fa fa-star" id="rating"> {{i['rating']}}</i>
                ><i class="fa fa-cart-plus add-to-cart" id="1"></i>
            </div>
            <img src={{ '/image?file=' +i['img'] }} />
            <p id="item-name">{{i['name']}}</p>
            <p id="item-price">Price : Rs. {{i['price']}}</p>
        </div>
    </a>
    {% endif %}
    {% endfor %}
</div>
<div id="Fast food" class="d-Fast food">
    <p id="category-name">Fast food</p>
    {% for i in dishes %}
    {% if i["category"] == "Fast food" %}
    <a href="/addToCart?id={{i['id']}}">
        <div id="item-card">
            <div id="card-top">
                <i class="fa fa-star" id="rating"> {{i['rating']}}</i>
                ><i class="fa fa-cart-plus add-to-cart" id="1"></i>
            </div>
            <img src={{ '/image?file=' +i['img'] }} />
            <p id="item-name">{{i['name']}}</p>
            <p id="item-price">Price : Rs. {{i['price']}}</p>
        </div>
    </a>
    {% endif %}
    {% endfor %}
</div>
</div>
</div>
<div id="cart">

```

```

<div class="taste-header">
  <div class="user">
    <i class="fa fa-user-circle" id="circle"> <a href="/account" style="text-decoration:
none;">Account</a></i>
  </div>
</div>
<div id="category-list">
  <p class="item-menu">Go For Hunt</p>
  <div class="border"></div>
  <div class="list-card">
    <a
      class="list-name"
      href="#biryani"
    >biryani</a>
  >
</div>
<div class="list-card">
  <a
    class="list-name"
    href="#chicken"
  >chicken</a>
  >
</div>
<div class="list-card">
  <a
    class="list-name"
    href="#paneer"
  >paneer</a>
  >
</div>
<div class="list-card">
  <a
    class="list-name"
    href="#vegetable"
  >vegetable</a>
  >
</div>
<div class="list-card">
  <a
    class="list-name"
    href="#chinese"
  >chinese</a>
  >
</div>
<div class="list-card">
  <a
    class="list-name"
    href="#south-indian"
  >south indian</a>
  >
</div>
<div class="list-card">
  <a

```

```

        class="list-name"
        href="#ice-cream"
    >ice-cream</a>
    >
</div>
<div class="list-card">
    <a
        class="list-name"
        href="#Dessert"
    >Dessert</a>
    >
</div>
<div class="list-card">
    <a
        class="list-name"
        href="#Fast food"
    >Fast food</a>
    >
</div>

</div>

</div>
</div>

<!-- mobile view -->
<!-- <div id="mobile-view" class="mobile-view">
    <div class="mobile-top">

        <div class="top-menu">
            <i class="fa fa-search"></i>
            <i class="fa fa-tag"></i>
            <i class="fa fa-cart-plus"></i>
            <i class="fa fa-cart-plus" id="m-cart-plus"> 0</i>
        </div>
    </div>

    <div class="item-container">
        <div class="category-header" id="category-header">
            <div class="list-card">
                <a
                    class="list-name"
                    href="#biryani"
                >biryani</a>
                >
            </div>
            <div class="list-card">
                <a
                    class="list-name"
                    href="#chicken"
                >chicken</a>
                >
            </div>
        </div>
    </div>
</div>

```

```

</div>
<div class="list-card">
  <a
    class="list-name"
    href="#paneer"
    >paneer</a
  >
</div>
<div class="list-card">
  <a
    class="list-name"
    href="#vegetable"
    >vegetable</a
  >
</div>
<div class="list-card">
  <a
    class="list-name"
    href="#chinese"
    >chinese</a
  >
</div>
<div class="list-card">
  <a
    class="list-name"
    href="#south-indian"
    >south indian</a
  >
</div>
</div>

<div id="food-items" class="food-items">
  <div id="biryani" class="m-biryani">
    <p id="category-name">Biryani</p>
  </div>
  <div id="chicken" class="m-chicken">
    <p id="category-name">Chicken Delicious</p>
  </div>
  <div id="paneer" class="m-paneer">
    <p id="category-name">Paneer Mania</p>
  </div>
  <div id="vegetable" class="m-vegetable">
    <p id="category-name">Pure Veg Dishes</p>
  </div>
  <div id="chinese" class="m-chinese">
    <p id="category-name">Chinese Corner</p>
  </div>
  <div id="south-indian" class="m-south-indian">
    <p id="category-name">South Indian</p>
  </div>
</div>
</div>

```



```

<div class="mobile-footer">
  <p>Home</p>
  <p>Cart</p>
  <p>offers</p>
  <p>orders</p>
</div>
</div> -->

</body>
</html>

```

Adding the images to a folder which has saved in the project folder .after that using the images of that folder to create a menu of our restaurant.by this code menu will be explored with images like category wise for example : Biryani is a category and having some type of biriyani's like chicken biriyani, egg biriyani, fish biriyani, mutton biriyani.....with ratings and cart symbol.

MENU EXPLORATION PAGE:

```

[
  {
    "id": 1,
    "name": "Ambur Biryani",
    "category" : "biryani",
    "rating" : 4.0,
    "price": 275,
    "img": "images/biryani/Ambur-Chicken-Biryani.jpg",
    "quantity": 1
  },
  {
    "id": 2,
    "name": "Hyderabadi Biryani",
    "category" : "biryani",
    "rating" : 4.5,
    "price": 280,
    "img": "images/biryani/Chicken-Biryani-hyd.jpg",
    "quantity": 1
  },
  {
    "id": 3,
    "name": "Egg Biryani",
    "category" : "biryani",
    "rating" : 4.4,
    "price": 240,
    "img": "images/biryani/egg-biryani.jpeg",
    "quantity": 1
  },
  {
    "id": 4,
    "name": "Goan Fish Biryani",
    "category" : "biryani",
    "rating" : 4.0,
    "price": 300,

```

```

    "img": "images/biryani/goan-fish-biryani.jpg",
    "quantity": 1
  },
  {
    "id": 5,
    "name": "Mutton Biryani",
    "category": "biryani",
    "rating": 4.8,
    "price": 340,
    "img": "images/biryani/hyd-Mutton-Biryani.jpg",
    "quantity": 1
  },
  {
    "id": 6,
    "name": "Kamrupi Biryani",
    "category": "biryani",
    "rating": 4.5,
    "price": 290,
    "img": "images/biryani/kamrupi-biryani.jpg",
    "quantity": 1
  },
  {
    "id": 7,
    "name": "Kashmiri Biryani",
    "category": "biryani",
    "rating": 4.6,
    "price": 350,
    "img": "images/biryani/kashmiri.pulao.jpg",
    "quantity": 1
  },
  {
    "id": 8,
    "name": "Memoni Biryani",
    "category": "biryani",
    "rating": 4.3,
    "price": 270,
    "img": "images/biryani/memonibiryani.png",
    "quantity": 1
  },
  {
    "id": 9,
    "name": "Mughlai Biryani",
    "category": "biryani",
    "rating": 4.3,
    "price": 290,
    "img": "images/biryani/mughlai-biryani.jpg",
    "quantity": 1
  },
  {
    "id": 10,
    "name": "Chicken Roast",
    "category": "chicken",
    "rating": 4.4,

```

```

    "price": 300,
    "img": "images/chicken/Chicken_roast.jpg",
    "quantity": 1
  },
  {
    "id": 11,
    "name": "Chicken Curry",
    "category" : "chicken",
    "rating" : 4.2,
    "price": 200,
    "img": "images/chicken/Chicken-Curry.jpg",
    "quantity": 1
  },
  {
    "id": 12,
    "name": "Chicken Do Pyaza",
    "category" : "chicken",
    "rating" : 3.8,
    "price": 290,
    "img": "images/chicken/Chicken-do-Pyaza.jpg",
    "quantity": 1
  },
  {
    "id": 13,
    "name": "Chicken Masala",
    "category" : "chicken",
    "rating" : 4.5,
    "price": 250,
    "img": "images/chicken/Chicken-Masala.jpeg",
    "quantity": 1
  },
  {
    "id": 14,
    "name": "Handi Chicken",
    "category" : "chicken",
    "rating" : 4.7,
    "price": 250,
    "img": "images/chicken/Handi-chicken.jpg",
    "quantity": 1
  },
  {
    "id": 15,
    "name": "Murgh Musallam",
    "category" : "chicken",
    "rating" : 4.2,
    "price": 240,
    "img": "images/chicken/Murgh-Musallam.jpg",
    "quantity": 1
  },
  {
    "id": 16,
    "name": "Matar Paneer",
    "category" : "paneer",

```

```

    "rating" : 4.1,
    "price": 200,
    "img": "images/paneer/Matar-Paneer.jpg",
    "quantity": 1
  },
  {
    "id": 17,
    "name": "Palak Paneer",
    "category" : "paneer",
    "rating" : 4.0,
    "price": 250,
    "img": "images/paneer/palak-paneer.jpg",
    "quantity": 1
  },
  {
    "id": 18,
    "name": "Paneer Butter Masala",
    "category" : "paneer",
    "rating" : 4.5,
    "price": 260,
    "img": "images/paneer/paneer-butter-masala.jpg",
    "quantity": 1
  },
  {
    "id": 19,
    "name": "Paneer Do Pyaza",
    "category" : "paneer",
    "rating" : 4.4,
    "price": 270,
    "img": "images/paneer/Paneer-Do-Pyaza.jpg",
    "quantity": 1
  },
  {
    "id": 20,
    "name": "Hyderabadi Paneer",
    "category" : "paneer",
    "rating" : 4.0,
    "price": 280,
    "img": "images/paneer/Paneer-Hyderabadi.jpg",
    "quantity": 1
  },
  {
    "id": 21,
    "name": "Paneer Lababdar",
    "category" : "paneer",
    "rating" : 4.1,
    "price": 290,
    "img": "images/paneer/paneer-lababdar.jpg",
    "quantity": 1
  },
},

```

```

{
  "id": 22,
  "name": "Shahi Paneer",
  "category" : "paneer",
  "rating" : 4.5,
  "price": 300,
  "img": "images/paneer/shahi-paneer.jpg",
  "quantity": 1
},
{
  "id": 23,
  "name": "Navratan Korma",
  "category" : "vegetable",
  "rating" : 4.4,
  "price": 200,
  "img": "images/vegetable/navratan-korma_-vegetable.png",
  "quantity": 1
},
{
  "id": 24,
  "name": "Veg Jalfrezi",
  "category" : "vegetable",
  "rating" : 4.3,
  "price": 210,
  "img": "images/vegetable/VEG-JALFREZI.jpg",
  "quantity": 1
},
{
  "id": 25,
  "name": "Veg Biryani",
  "category" : "vegetable",
  "rating" : 4.6,
  "price": 230,
  "img": "images/vegetable/vegetable-biryani.jpg",
  "quantity": 1
},
{
  "id": 26,
  "name": "Veg Curry",
  "category" : "vegetable",
  "rating" : 4.3,
  "price": 260,
  "img": "images/vegetable/vegetable-curry.jpeg",
  "quantity": 1
},
{
  "id": 27,
  "name": "Veg Kolhapur",
  "category" : "vegetable",
  "rating" : 4.3,
  "price": 250,
  "img": "images/vegetable/vegetable-kolhapuri.jpg",
  "quantity": 1
}

```

```

},
{
  "id": 28,
  "name": "Veg Masala",
  "category" : "vegetable",
  "rating" : 4.2,
  "price": 230,
  "img": "images/vegetable/vegetable-masala.jpg",
  "quantity": 1
},
{
  "id": 29,
  "name": "Veg Pakora",
  "category" : "vegetable",
  "rating" : 3.5,
  "price": 240,
  "img": "images/vegetable/vegetable-pakora.jpg",
  "quantity": 1
},
{
  "id": 30,
  "name": "Momos",
  "category" : "chinese",
  "rating" : 4.3,
  "price": 150,
  "img": "images/chinese/cabbage-momos-.jpg",
  "quantity": 1
},
{
  "id": 31,
  "name": "Chicken Manchurian",
  "category" : "chinese",
  "rating" : 4.5,
  "price": 100,
  "img": "images/chinese/ChickenManchurian.jpg",
  "quantity": 1
},
{
  "id": 32,
  "name": "Chili Chicken",
  "category" : "chinese",
  "rating" : 180,
  "price": 180,
  "img": "images/chinese/Chili-Chicken.jpg",
  "quantity": 1
},
{
  "id": 33,
  "name": "Chowmein",
  "category" : "chinese",
  "rating" : 4.3,
  "price": 100,
  "img": "images/chinese/chowmin.jpg",

```

```

    "quantity": 1
  },
  {
    "id": 34,
    "name": "Spring Roll",
    "category" : "chinese",
    "rating" : 4.3,
    "price": 120,
    "img": "images/chinese/spring-rolls.jpg",
    "quantity": 1
  },
  {
    "id": 35,
    "name": "Szechuan Chicken",
    "category" : "chinese",
    "rating" : 4.3,
    "price": 190,
    "img": "images/chinese/szechuan-chicken.jpg",
    "quantity": 1
  },
  {
    "id": 36,
    "name": "Fried Rice",
    "category" : "chinese",
    "rating" : 4.3,
    "price": 90,
    "img": "images/chinese/veg-fried-rice.jpg",
    "quantity": 1
  },
  {
    "id": 37,
    "name": "Butter Masala Dosa",
    "category" : "south indian",
    "rating" : 4.3,
    "price": 50,
    "img": "images/south-indian/Butter-Masala-Dosa.png",
    "quantity": 1
  },
  {
    "id": 38,
    "name": "idli",
    "category" : "south indian",
    "rating" : 4.3,
    "price": 30,
    "img": "images/south-indian/idli-with-rice-flour.jpg",
    "quantity": 1
  },
  {
    "id": 39,
    "name": "Masala Dosa",
    "category" : "south indian",
    "rating" : 4.3,
    "price": 60,

```

```

    "img": "images/south-indian/masala-dosa.jpg",
    "quantity": 1
  },
  {
    "id": 40,
    "name": "Mysore Bonda",
    "category" : "south indian",
    "rating" : 4.3,
    "price": 30,
    "img": "images/south-indian/mysore-bonda.jpg",
    "quantity": 1
  },
  {
    "id": 41,
    "name": "Onion Uttapam",
    "category" : "south indian",
    "rating" : 4.3,
    "price": 40,
    "img": "images/south-indian/onion-uttapam.jpg",
    "quantity": 1
  },
  {
    "id": 42,
    "name": "Plain Dosa",
    "category" : "south indian",
    "rating" : 4.3,
    "price": 20,
    "img": "images/south-indian/plain-dosa.jpeg",
    "quantity": 1
  },
  {
    "id": 43,
    "name": "Rava Uttapam",
    "category" : "south indian",
    "rating" : 4.3,
    "price": 40,
    "img": "images/south-indian/Rava-Uttapam.jpg",
    "quantity": 1
  },
  {
    "id": 44,
    "name": "Sambhar Vada",
    "category" : "south indian",
    "rating" : 4.3,
    "price": 40,
    "img": "images/south-indian/sambhar-vada.jpg",
    "quantity": 1
  },
  {
    "id": 45,
    "name": "chocolate icecream",
    "category" : "ice-cream",
    "rating" : 4.4,

```



```

    "price": 60,
    "img": "images/icecream/chocolate1.jpg",
    "quantity": 1
  },
  {
    "id": 46,
    "name": "Butter scotch",
    "category" : "ice-cream",
    "rating" : 4.5,
    "price": 50,
    "img": "images/icecream/butterscotch.jpg",
    "quantity": 1
  },
  {
    "id": 47,
    "name": "vanila",
    "category" : "ice-cream",
    "rating" : 4.6,
    "price": 40,
    "img": "images/icecream/vanila.jpg",
    "quantity": 1
  },
  {
    "id": 47,
    "name": "custard apple",
    "category" : "ice-cream",
    "rating" : 4.8,
    "price": 80,
    "img": "images/icecream/custard.jpg",
    "quantity": 1
  },
  {
    "id": 48,
    "name": "mango",
    "category" : "ice-cream",
    "rating" : 4.9,
    "price": 70,
    "img": "images/icecream/mango.webp",
    "quantity": 1
  },
  {
    "id": 49,
    "name": "strawberry",
    "category" : "ice-cream",
    "rating" : 4.6,
    "price": 50,
    "img": "images/icecream/strawberry.jpg",
    "quantity": 1
  },
  {
    "id": 50,
    "name": "cake",

```

```

"category" : "Dessert",
"rating" : 4.4,
"price": 50,
"img": "images/Dessert/cake.avif",
"quantity": 1
},
{
  "id": 51,
  "name": "kunafa",
  "category" : "Dessert",
  "rating" : 4.7,
  "price": 100,
  "img": "images/Dessert/kunafa.jpg",
  "quantity": 1
},
{
  "id": 52,
  "name": "gulab jamun",
  "category" : "Dessert",
  "rating" : 4.5,
  "price": 100,
  "img": "images/Dessert/gulabjamun.jpg",
  "quantity": 1
},
{
  "id": 53,
  "name": "Halwa",
  "category" : "Dessert",
  "rating" : 4.4,
  "price": 60,
  "img": "images/Dessert/halwa.jpg",
  "quantity": 1
},
{
  "id": 54,
  "name": "soan papdi",
  "category" : "Dessert",
  "rating" : 4.6,
  "price": 80,
  "img": "images/Dessert/soanpapdi.jpg",
  "quantity": 1
},
{
  "id": 55,
  "name": "Burger",
  "category" : "Fast food",
  "rating" : 4.6,
  "price": 99,
  "img": "images/Fast food/burger.jpg",
  "quantity": 1
},
{
  "id": 56,

```

```

    "name": "pizza",
    "category" : "Fast food",
    "rating" : 4.6,
    "price": 120,
    "img": "images/Fast food/pizza.jpg",
    "quantity": 1
  },
  {
    "id": 57,
    "name": "french fries",
    "category" : "Fast food",
    "rating" : 4.6,
    "price": 99,
    "img": "images/Fast food/fries.jpg",
    "quantity": 1
  },
  {
    "id": 58,
    "name": "shawarma",
    "category" : "Fast food",
    "rating" : 4.6,
    "price": 99,
    "img": "images/Fast food/shawarma.jpeg",
    "quantity": 1
  }
]

```

The page will be having menu of all items which have mentioned in the code and the page is having about , services , cart , your order , contact , account ,search, and then offers .

1.ABOUT PAGE:

```

<!DOCTYPE html>
<html lang="en">
  <head>

    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Navigation Box</title>
    <style>
      body {
        background-image: url('https://wallpaperaccess.com/full/2975527.jpg'); /* Provide the URL for your
background image */
        background-size: cover;
        font-family: Arial, sans-serif;
        margin: 0;
        padding: 0;
        color: white;
      }</style>
    </head>
  </head>

```

```

<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>About Us</title>
</head>
<center><body bgcolor="pink">
  <header>
    <h1>About Us</h1>
  </header>
  <section>

```

```

    <h2><p>Yumjoy, is a comprehensive online food ordering system designed to give users with a convenient
    and efficient platform for ordering food from their favorite restaurants . Online food ordering is the process of
    ordering food from a website. <br>The effort to create an online food ordering system aims to replace the manual
    method of taking orders with a digital one. The ability to rapidly and correctly create order summary reports
    whenever necessary is a key factor in the development of this project</p></h2><br><br>

```

```

</section>

```

```

<section>

```

```

  <h1>Our Features</h1>

```

```

  <ul>

```

```

    <h2><li>History</li>

```

```

    <li>Menu Exploration</li>

```

```

    <li>Offers</li></h2>

```

```

  </ul><br><br><br>

```

```

<center> <h2><p>"From kitchen to doorstep in no time"<br>

```

```

  "Beyond the boundaries of taste"

```

```

</p></h2></center>

```

```

</section>

```

```

</body></center>

```

```

</html>

```

2.SERVICES PAGE:

```

<!DOCTYPE html>

```

```

<html lang="en">

```

```

  <head>

```

```

    <meta charset="UTF-8">

```

```

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

```

```

  <title>Navigation Box</title>

```

```

  <style>

```

```

    body {

```

```

      background-image:

```

```

url('https://static.vecteezy.com/system/resources/previews/002/001/840/large_2x/food-delivery-service-design-
vector.jpg'); /* Provide the URL for your background image */

```

```

      background-size: cover;

```

```

      font-family: Arial, sans-serif;

```

```

      margin: 0;

```

```

      padding: 0;

```

```

        color:green;
    }</style>
</head>
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Food App Services</title>
    <link rel="stylesheet" href="styles.css"> <!-- You can link an external stylesheet for styling -->
</head>
<body bgcolor="pink">

    <main>
        <section id="services">
            <h1>Our Services</h1>
            <div class="service">

                <h1><p>Hey foodies!!<br></p></h1>
                <h3><p>Fulfill your belly tummies with your favourite food from our restaurant.<br>you will feel the
same experience as like a home food.<br><center>checkout the restaurant guys!.</center> </p></h3>
            </div>
            <div class="service">
                <h1>Catering</h1>
                <h3><p>Let us cater your events with delicious food options tailored to your needs.we deliver your
food on time without any delay.And our food items are neatly packed without any leakage.we provide traditional
items for traditional events.</p></h3>
            </div>
            <!-- Add more service sections as needed -->
        </section>
    </main>

    <footer>

```

```

        <h1>Terms of services</h1>
        <h3>Acceptance of terms</h3>
        <p>Thank you for using yumjoy. These Terms of Service (the "Terms") are intended to make you
aware of your legal rights and responsibilities with respect to your access to and use of the yumjoy website at
www.hungryji.com (the "Site") and any related mobile or software applications ("yumjoy Platform") including
but not limited to delivery of information via the website whether existing now or in the future that link to the
Terms (collectively, the "Services").</p>
        <p>Please read these Terms carefully. By accessing or using the yumjoy Platform, you are agreeing to
these Terms and concluding a legally binding contract with yumjoy Limited (formerly known as yumjoy Private
Limited and yumjoy Media Private Limited) and/or its affiliates (excluding yumjoy Foods Private Limited)
(hereinafter collectively referred to as "yumjoy"). You may not use the Services if you do not accept the Terms
or are unable to be bound by the Terms. Your use of the yumjoy Platform is at your own risk, including the risk
that you might be exposed to content that is objectionable, or otherwise inappropriate.

```

In order to use the Services, you must first agree to the Terms. You can accept the Terms by:

Clicking to accept or agree to the Terms, where it is made available to you by yumjoy in the user interface for any particular Service; or

Actually using the Services. In this case, you understand and agree that yumjoy will treat your use of the Services as acceptance of the Terms from that point onwards.</p>

```
<h1>Privacy Policy</h1>
```

```
<p>You represent that you have read, understood and agreed to our Privacy Policy. Please note that we may disclose information about you to third parties or government authorities if we believe that such a disclosure is reasonably necessary to (i) take action regarding suspected illegal activities; (ii) enforce or apply our Terms and Privacy Policy; (iii) comply with legal process or other government inquiry, such as a search warrant, subpoena, statute, judicial proceeding, or other legal process/notice served on us; or (iv) protect our rights, reputation, and property, or that of our Customers, affiliates, or the general public</p>
```

```
<div class="social-icons">
```

```
<!-- Include social media icons here -->
```

```
</div>
```

```
</footer>
```

```
</body>
```

```
</html>
```

3.CART PAGE:

```
<html lang="en">
```

```
<head>
```

```
<meta charset="UTF-8">
```

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
<title>Document</title>
```

```
<link rel="stylesheet" href="{ {url_for('static',filename='index.css')}} " />
```

```
<link
```

```
rel="stylesheet"
```

```
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css"
```

```
/>
```

```
</head>
```

```
<body>
```

```
<div id="cart-page" class="cart-toggle">
```

```
<p id="cart-title">Cart Items</p>
```

```
<p id="m-total-amount">Total Amout : 100</p>
```

```
<table>
```

```
<thead>
```

```
<tr>
```

```
<td>Item</td>
```

```
<td>Name</td>
```

```
<td>Image</td>
```

```
<td>Price</td>
```

```
<td>Count</td>
```

```
<td>Del</td>
```

```
</tr>
```

```
</thead>
```

```
<tbody id="table-body">
```

```
{% for i in data %}
```

```
<tr>
```

```
<td>{{ loop.index }}</td>
```

```
<td>{{ i['name'] }}</td>
```

```
<td></td>
```

```
<td>{{ i['price'] }}</td>
```

```

<td>{ { i['count'] } }</td>
<td>
<button>
  <a href="/delItem?id={ { i['id'] } }">
    del
  </a>
</button>
</td>
</tr>
{ % endfor % }
</tbody>
</table>
<h3>{ { total } }</h3>
<div class="btn-box">
  <a href="/checkout?total={ { total } }">
    <button class="cart-btn">Checkout</button>
  </a>
</div>
</div>
</body>
</html>

```

4.PAYMENT PAGE:

```

<html lang="en">
<head>

  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Navigation Box</title>
  <style>
    body {
      background-image:url('https://e0.pxfuel.com/wallpapers/648/385/desktop-wallpaper-organic-
vegetables-for-cooking-spices-herbs-and-fresh-vegetables-for-cooking-on-dark-metal-background-with-sp-
vegetables-fresh-vegetables-organic-vegetables-organic-food-thumbnail.jpg'); /* Provide the URL for your
background image */
      background-size: cover;
      font-family: Arial, sans-serif;
      margin: 0;
      padding: 0;
      color: white;
    }</style>
</head>
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<center><body><br><br><br>
  <h1>Select payment mode</h1><br><br>
  <h1>
    <input type="checkbox" id="offline" name="fav_payment" value="cash on delivery">

```

```

<label for="cash on delivery">cash on delivery</label><br><br>
<input type="checkbox" id="online" name="fav_payment" value="phone pay">
<label for="phone pay" onclick="img()">phone pay</label><br><br>

</h1>
<a href="/placeOrder?total={{total}}"><button type="button" style="width:
120px;height:40px;background-color:rgb(201, 64, 91)">place order!</button></a>

```

```

<h2>Total Amount:{{total}}</h2>
</body></center>

```

```

<script>
function img(){
    i=document.getElementById('img');
    if(i.style.display==='block')
    {i.style.display='none'}
    else{i.style.display='block'}
}
</script>
</html>

```

5.CONTACT PAGE:

```

<html>
<head>

<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Navigation Box</title>
<style>
body {
    background-image:url('https://thumbs.dreamstime.com/b/different-fresh-foods-vegetables-herbs-spices-
black-background-various-foods-bowls-perfect-cover-photos-various-156085537.jpg'); /* Provide the URL for
your background image */
    background-size: cover;
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
    color: white;
}
</style>
</head>
<center><body bgcolor="pink">
<h1>Contact Us</h1><br>
<h1>Help</h1>
<h2><p>Do you have any queries please contact us with the details given below<br></p></h2>
<h2>Email id: shaikshareefa448@gmail.com</h2><h2><br>contact: 9652775249<br></h2></p>

</body></center>
</html>

```


6.OFFERS PAGE:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Navigation Box</title>
    <style>
      body {
        background-image: url('https://www.creativefabrica.com/wp-content/uploads/2020/06/18/Comic-book-
cartoon-cloud-for-text-Graphics-4398675-1.jpg'); /* Provide the URL for your background image */
        background-size: cover;
        font-family: Arial, sans-serif;
        margin: 0;
        padding: 0;
        color: black;
      }
    </style>
  </head>
  <body bgcolor="pink">
    <div style="background-color: pink; width: max-content">
      <h1>Hey Foodies!!</h1>
      <h1> Here are the offers to fill your tummies....</h1></div>
      <div class="container">
        <marquee behavior="scroll" direction="left" scrollamount="10">
          
          
          
          
          
        </marquee>
      </div>
    </body>
  </html>
```

7.ACCOUNT PAGE:

```
<!DOCTYPE html>
<html>
  <head>

    <meta charset="UTF-8">
```

```

<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Navigation Box</title>
<style>
  body {
    background-image: url('https://wallpaperaccess.com/full/462773.jpg'); /* Provide the URL for your
background image */
    background-size: cover;
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
    color:black;
  }</style>
</head>
<head>
  <title>User Account Photo</title>
</head>
<center><body bgcolor="pink">
  <div style="background-color:pink;width:max-content"><h1>User Account</h1>
  
  <h1>User name:{{user}}</h1>

<h2><table border="1",cellspacing="4" cellpadding="4">
  <caption><b>History</b></caption>
  <tr>
    <th> item Name</th>
    <th>Price</th>
    <th>Date</th>

  </tr>
  {% for i in data['orders'] %}
  {% for j in i['order'] %}
  <tr>
    <td>
      {{j['name']}}
    </td>
    <td>
      {{j['price']}}
    </td>
    <td>
      {{i['date']}}
    </td>
  </tr>
  {% endfor %}
  {% endfor %}
  <tr>
    <td>
      Total :
    </td>
    <td>{{total}}</td>
  </tr>

```

```

</table></h2>
</div>
</body></center>
</html>

```

8.ADMIN PAGE:

```

<html lang="en">
<head>

    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Navigation Box</title>
    <style>
        body {
            background-image: url('https://img.freepik.com/premium-photo/plate-food-with-chicken-rice-biryani-
with-fire-smoke-background_885092-227.jpg'); /* Provide the URL for your background image */
            background-size: cover;
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
            color: white;
        }</style>
</head>
<body bgcolor="pink">
    <center> <h1>ADMIN LOGIN </h1> </center>
    <center><p>{ { status } } </p></center>
    <form action="/adlog" method="post">
        <center><h1><div class="container">
            <label>Username : </label>
            <input type="text" placeholder="Enter Username" name="username" required> <br>
            <label>Password : </label>
            <input type="password" placeholder="Enter Password" name="password" required><br>
            <input type="submit">
        </div> </h1> </center>
    </form>
    <center>
        
        </center>
</body>
</html>

```

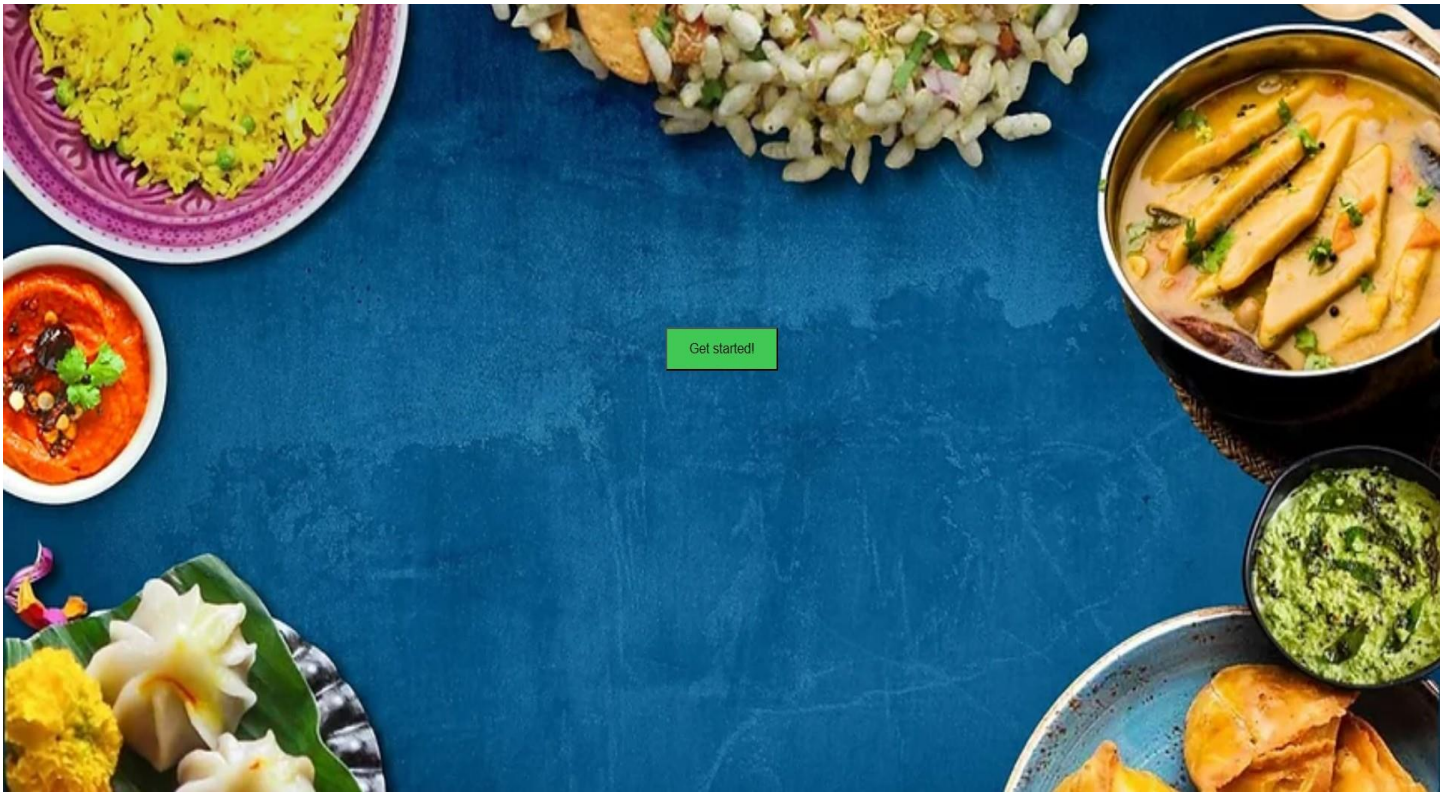
9.TEST CASES

Tab: 9.1 Test Cases

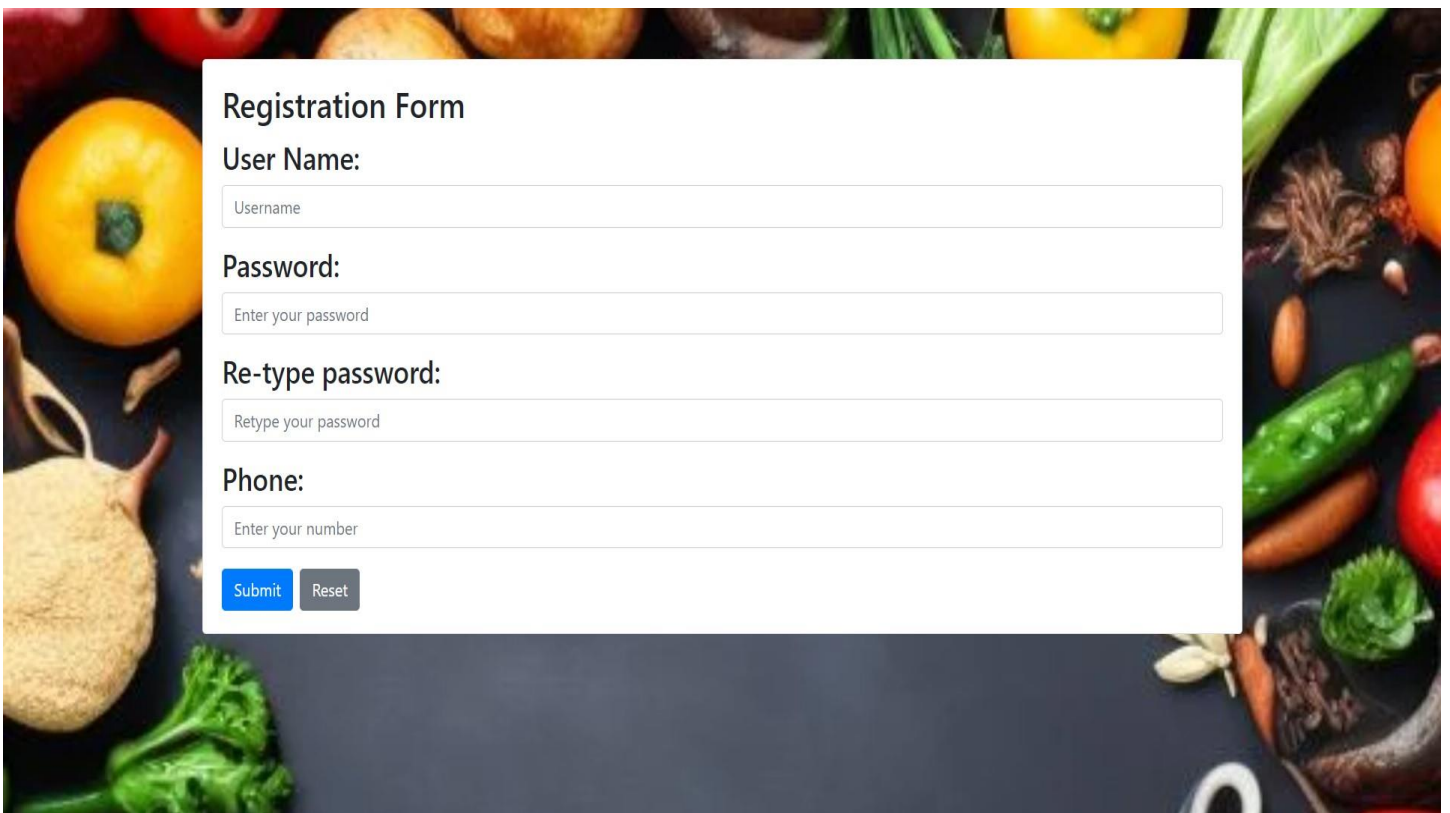
Sno	Test Case	Excepted Result	Result	Remarks (IF Fail)
1.	User Register	If User registration successfully	Pass	If already user name exists then it fails.
2.	User login	If user login successfully	pass	If password wrong then it fails
3.	Menu display	Menu displays correctly	pass	Menu displays incorrectly.
4.	Adding items to cart	User can place an order with items in their cart	pass	User cannot place an order without items in their cart
5.	Order placement	Order placed sucessfull	pass	Order doesnot placed
6.	User account management	Updating history in the account.	pass	If update was not done it exists fail.
7.	Total amount calculation	Amount will be calculated for every order with dates	Pass	If amount calculation was wrong then it will remains fail.
8.	Payment	Payment successful	Pass	Payment will be unsucessfull.
9.	Search functionality	Search functionality for finding specific items	Pass	Test search results with incomplete queries.

10. SCREEN SHOTS

Get Started Page:

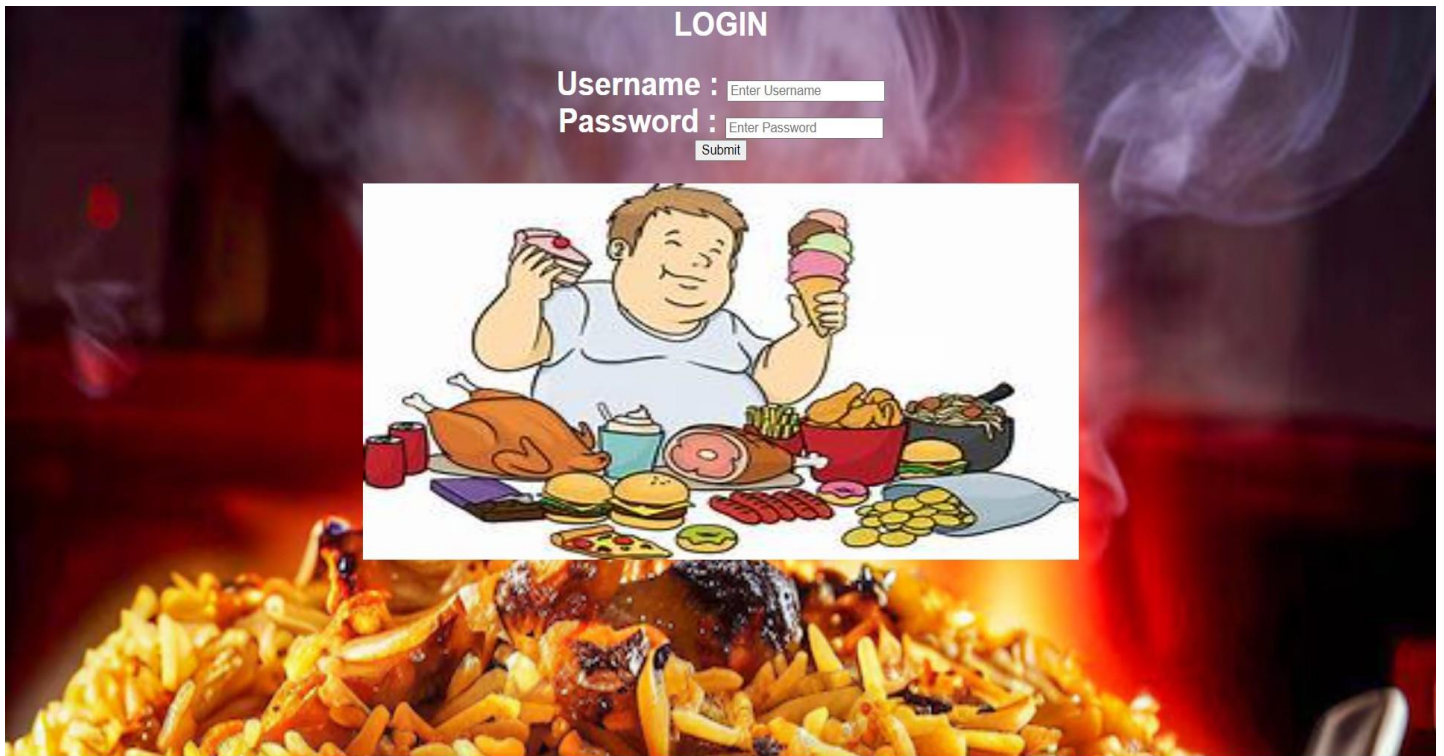


Registration Page:

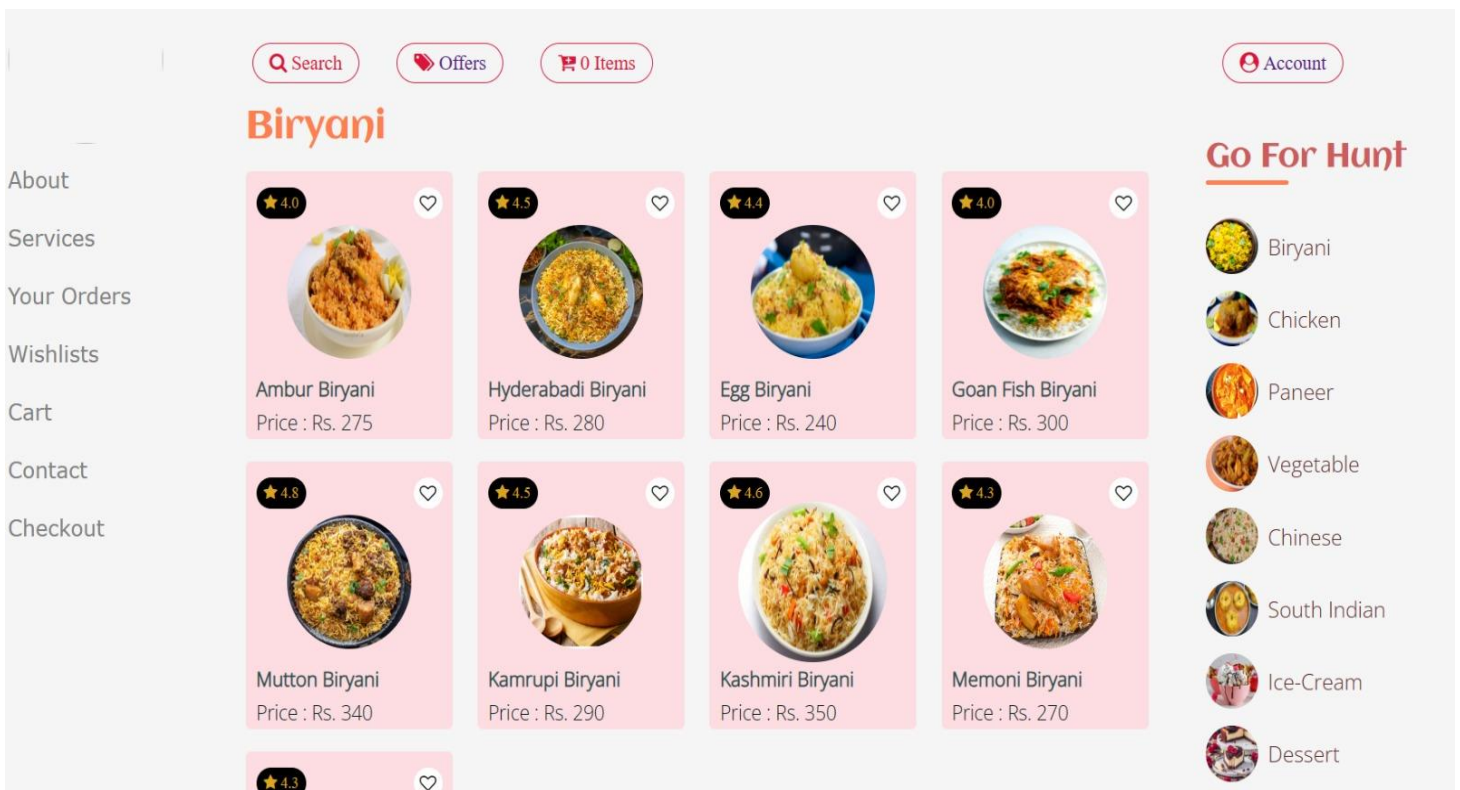
A screenshot of a registration form overlaid on a background of fresh vegetables. The form is titled 'Registration Form' and contains the following fields and buttons:

- User Name:** A text input field with the placeholder 'Username'.
- Password:** A text input field with the placeholder 'Enter your password'.
- Re-type password:** A text input field with the placeholder 'Retype your password'.
- Phone:** A text input field with the placeholder 'Enter your number'.
- Submit** and **Reset** buttons at the bottom left of the form.

Login Page:



Home Page:



Menu Exploration Page:

Go For Hunt



Biryani



Chicken



Paneer



Vegetable



Chinese



South Indian



Ice-Cream



Dessert



Fast Food

About Page:


About Us

Hungryji is a comprehensive online food ordering system designed to give users with a convenient and efficient platform for ordering food from their favorite restaurants . Online food ordering is the process of ordering food from a website. The effort to create an online food ordering system aims to replace the manual method of taking orders with a digital one. The ability to rapidly and correctly create order summary reports whenever necessary is a key factor in the development of this project

Our Features

- History
- Menu Exploration
- Offers

"From kitchen to doorstep in no time"
"Beyond the boundaries of taste"



Service Page

Our Services

Hey foodies!!

Fulfill your belly tumtummies with your favourite food from our restaurant. you will feel the same experience as like a home food.

Catering

Let us cater your events with delicious food options tailored to your needs. we deliver your food on time without any delay. And our food items are neatly packed without any leakage. we provide traditional items for traditional events.

Terms of services

Acceptance of terms

Thank you for using Hungryji. These Terms of Service (the "Terms") are intended to make you aware of your legal rights and responsibilities with respect to your access and use of the Hungryji website at www.hungryji.com (the "Site") and any related mobile or software applications ("Hungryji Platform") including but not limited to delivery of information via the website whether existing now or in the future that link to the Terms collectively, the "Services").

Please read these Terms carefully. By accessing or using the Hungryji Platform, you are entering into these Terms and concluding a legally binding contract with Hungryji Limited (formerly known as Hungryji Private Limited and Hungryji Media Private Limited) and/or its affiliates, including Hungryji Media Private Limited (hereinafter collectively referred to as "Hungryji"). You may not use the Services if you do not accept the Terms or are unable to be bound by the Terms. Your use of the Hungryji Platform is at your own risk, including the risk that you might be exposed to content that is objectionable, or otherwise inappropriate. In order to use the Services, you must first agree to the Terms. You can accept the Terms by: Clicking to accept or agree to the Terms, where it is made available to you by Hungryji in the user interface for any particular Service; or Actually using the Services. In this case, you understand and agree that Hungryji will treat your use of the Services as acceptance of the Terms from that point onwards.

Privacy Policy

You represent that you have read, understood and agreed to our Privacy Policy. Please note that we may disclose information about you to third parties or government authorities if we believe that such a disclosure is reasonably necessary to (i) take action regarding suspected illegal activities; (ii) enforce or apply our Terms and Privacy Policy; (iii) comply with legal process or other government inquiry, such as a search warrant, subpoena, statute, judicial proceeding, or other legal process/notice served on us; or (iv) protect our rights, reputation, and property, or that of our customers, affiliates, or the general public.

Cart Page:

Cart Items

Item

Name

Image


Price

Count

Del

1

Goan Fish Biryani




300

1

del

2

Veg Kolhapur




250

1

del

3

pizza




120

1

del

4

gulab jamun



100

1

del

770

CHECKOUT

Contact Page:

Contact Us

Help

Do you have any queries please contact us with the details given below

Email id: pasifa075@gmail.com

contact: 9640612567

Offers Page:

Hey Foodies!!

Here are the offers to fill your tummies....

**SPECIAL
OFFER**



Hey Foodies!!

Here are the offers to fill your tummies....

**COMBO
OFFER**

Burger+Fries+Ice Tea

Rajma Chawal

Dal Makhani+Lachcha Parantha

Pizza+Garlic Bread+Cold Drinks

COOL & DELICIOUS
ICE-CREAM
FOOD - THIS WEEKEND ONLY

**35%
DISCOUNT**

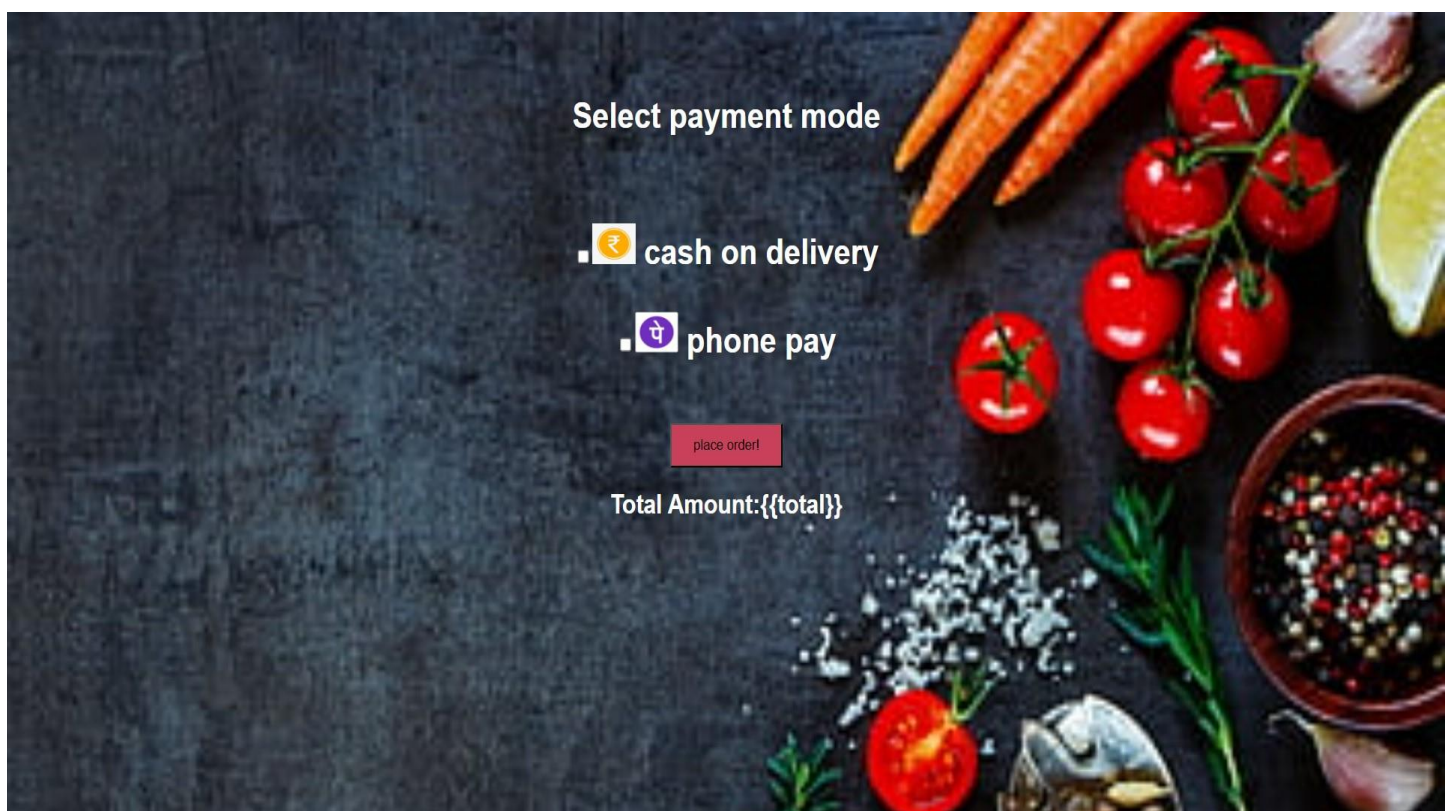
ORDER NOW

FREE DELIVERY
0269 8568 7909

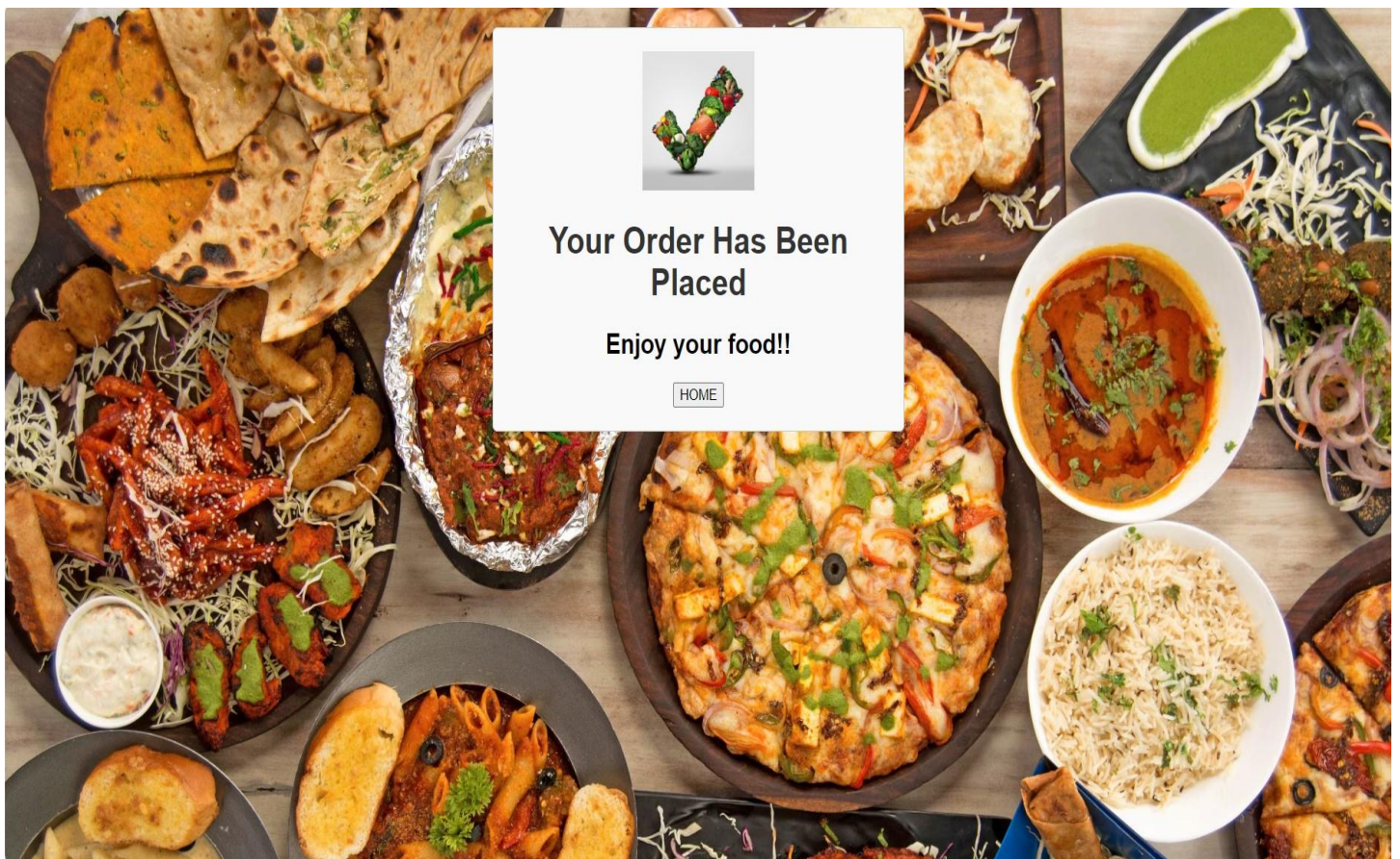
Account Page:



Payment Page:



Order Successful Page:



11. CONCLUSION

Restaurant Management System is a web-based technology that aids the restaurant industry in carrying out tasks effectively and efficiently. It aids in managing cash flow for managers. Managers can view analytics data to assess company growth. The manager can control orders and employee schedules by using this system.

The full complement is a restaurant management system. It provides access to the Online Order platform, third-party connectors software, and comprehensive CRM solution, which together cover a sizable portion of your restaurant's requirements. They are not the outdated hardware and software sets for restaurants that were previously offered. They are the hottest things around, smooth, manageable, inexpensive, and quick...

12. REFERENCES

- [1] C. Branigan (2001), Wireless, handheld solutions prevail at NECC 2001, retrieved January 10, 2007 from <http://www.eschoolnews.com/news/showStory.cfm?ArticleID=2865>.
- [2] Watkins Demien. Mobile web services technical roadmap. http://www.microsoft.com/serviceproviders/mobilewebservicess/mws_tech_roadmap.asp. 2003, 11.
- [3] Y. Xiang, W. Zhou and M. Chowdhury. Toward pervasive computing in restaurant, First International Conference on E-Business and Telecommunication Networks (ICETE 2004), Setubal, Portugal, August, 2004, pp312-317.
- [4] IEEE 802.11TM WIRELESS LOCAL AREA NETWORKS. IEEE Working Group (WG), <http://www.ieee802.org/11>.
- [5] Knikker, R., Guo, Y., Li, J.L., Albert Kwan, K.H., Yip, K.Y., Cheung, D.W., et al..A Web services choreography scenario for interoperating bioinformatics applications. BMC Bioinformatics 2004, 5:25-28.
- [6] FENSEL D, BUSSLER C. The Web service modeling framework WSMF. Electric Commerce Research and Application, 2002, Vol.1, No.2: pp113-137.
- [7] Meier J D, Mackman A, Dunner M, Vasireddy S. Web services security S. <http://msdn.microsoft.com/library/default.asp?url=/library/en us/dnnetsec/html/SecNetch10.asp>. 2002.
- [8] Powell Matt. Real SOAP security. <http://msdn.microsoft.com/library/default.asp?url=/library/en us/dnservice/html/service11212001.asp>. 2001.
- [9] Kirtland Mary. Secure web services using the SOAP toolkit. http://msdn.microsoft.com/archive/default.asp?url=/archive/en us/dnxml/html/websvcs_usingsoap.asp. 2001.
- [10] R. G. Duncan and M.M. Shabot Secure remote access to a clinical data repository using a wireless personal digital assistant (PDA) . Proceedings of the American Medical Informatics Association

Wireless Food Ordering System Based on Web Services

XU Hongzhen, TANG Bin, SONG Wenlin

Department of Computer Science and Technology
East China Institute of Technology
Fuzhou, Jiangxi Province, 344000, China
xhz_97@163.com

Abstract—Current wireless communications enable people to easily exchange information, while web services provide loosely-coupled and platform-independent ways of linking applications across the Internet or Intranet. This paper presents an integration of wireless communication technologies and web services technologies to realize a wireless food ordering system. In this system, it implements wired and wireless data access to the servers and food ordering functions through both desktop PCs and mobile devices such as PDAs over a wired/wireless integrated local area network. To sure the security of the system, the secure web service architecture and some security strategies to ensure mobile communication security are discussed. Web services-based wireless applications on mobile devices provide a means of convenience, improving efficiency and accuracy for restaurants by saving time, reducing human errors, etc.

Keywords *Web Services; Wireless; Food Ordering System; security*

I. INTRODUCTION

The rapid developments in information technology, particularly in wireless communication and web services technologies, are greatly changing the way people access and work with information. The convenience and powerful functionality offered by mobile devices such as PDAs, has encouraged many people to investigate the benefits of using them. Wireless and handheld devices abound as vendors pitch the common themes of one-to-one computing, instant communication and anytime, anywhere information access^[1]. While web services provide a technology for service-oriented computing. Web services allow programs written in different languages on different platforms to communicate with each other in a standard way^[2]. By integrating these technologies, consistent business models can be implemented on a broad array of devices: not just on mobile devices operating over mobile networks, but also on servers and PCs connected to the Internet.

The food ordering process in restaurants requires the coordination of simple tasks. Instruction flows mainly from customers to waiters then to kitchen and/or the bar staff, finally to the cashier^[3]. In a medium to large and busy restaurant this coordination is a challenge and requires an efficient ordering system. Errors in ordering processes lead to incorrect or out of sequence meal preparation or non-consumable and results in added cost to the business.

This paper presents an integration of wireless communication technologies and web services technologies to realize a wireless food ordering system. In this system, it implements wired and wireless data access to the servers and food ordering functions through both desktop PCs and mobile devices such as PDAs over a wired/wireless integrated local area network. The system is based on secure web service architecture and some security strategies to ensure mobile communication security are adopted. Web services-based wireless applications on mobile devices provide a means of convenience, improving efficiency and accuracy.

II. WIRELESS LAN AND WEB SERVICES

A. Wireless LAN

A wireless LAN (WLAN, Wireless Local Area Network) is a flexible data communication system implemented as an extension to or as an alternative for, a wired LAN within a building or campus^[4]. Using electromagnetic waves, WLANs transmit and receive data over the air, minimizing the need for wired connections. Thus, WLANs combine data connectivity with user mobility, and, through simplified configuration, enable movable LANs.

The IEEE 802.11 group of standards specifies the technologies for wireless LANs. 802.11 standards use the Ethernet protocol and CSMA/CA (carrier sense multiple access with collision avoidance) for path sharing and include an encryption method, the Wired Equivalent Privacy algorithm. The 802.11a, b, and g standards are the most common for home wireless access points and large business wireless systems.

A remote user can use WLAN to access the Internet through public access points ("hotspots") provided by service providers. When in the office, they may access WLAN through wireless access points. In enterprise environments, WLANs are usually complemented by security mechanisms, such as VPN (Virtual Private Network).

Over the last several years, WLANs have gained strong popularity in some markets, including the health-care, retail, manufacturing, and academic areas. These industries have profited from the productivity gains of using hand-held terminals and notebook computers to transmit real-time information to centralized hosts for processing. Today WLANs are becoming more widely recognized as a general-

purpose connectivity alternative for a broad range of business customers.

B. Web Services

Quickly becoming a significant technology in the evolution of the Internet is web services, a set of standards that can interconnect systems over a variety of networks. It is an open XML-based technology providing a generic data exchange format and has been rapidly adopted by many vendors. Web services can easily be built upon existing applications, no matter what the underlying technology is. Because they are expected to have a growing familiarity and acceptance among many users and offer great technological promises, Web services are an interesting subject for the investigation of their possible application in many systems [5].

Web services are a new generation of web application. It combines the advantages of the component-oriented methods and web techniques, and they can describe its own service. It can also publish, locate and transfer modularized application in web. The provided functions of web services may be simple, but it also contains extraordinary complicated business logic. Web services represent a kind of implementation of SOA (Service-Oriented Architecture), and they are the most popular one. In addition, the three operations of SOA can only process when the components of SOA interact. Therefore some standardized techniques are used in web services, including UDDI, WSDL, HTTP, SOAP, and XML and so on. Web services become the best choice for developing application of SOA [6].

III. DESIGN AND IMPLEMENTATION OF THE SYSTEM

A. System Architecture

In the system, we adopt four-tiered web-based client-server architecture. Figure 1 shows the overview of the system architecture. The system is conceptually composed of six main components: the web server, database server, cash register, mobile context server, mobile user and desktop user. The web server provided relevant information for mobile devices or desktop PCs on a wired/wireless integrated local area network using WSDL (Web Service Description Language) to describe functions and protocols. The web server then transmits to the mobile devices or desktop PCs. The user binds the web server and the WSDL. This enables the web service to be used by correspondence using SOAP (Simple Object Access Protocol). The database server saves all information of the system such as food information, ordering information, client information. The cash register is responsible for cost calculation of the consumer. The mobile context server applies context to the contents by using styles, an attribute override, and templates according to the resources of a given mobile device.

Desktop users can ask for services after checking the WSDL of the service from the web server. A desktop on a wired network can be used to browse full contents on one screen shot. When a user requests food information through a wired network, the web server serves the information by connecting to the database server. When a user requests food

information through a wireless network, the mobile context server divides the context pages according to the screen size of the mobile device. It also filters the pages according to mobile devices and then browses the adopted content from the context server to the mobile web browser. The mobile context server reconfigures contents offered by the web server.



Figure 1. The System Architecture

B. Web Service Security Model

Web service security can be applied at three levels [7]:

- Platform/transport-level (point-to-point) security;
- Application-level (customer) security;
- Message-level (end-to-end) security.

Each approach has different strengths and weaknesses. The choice of the approach is largely dependent on the characteristics of the architecture and platforms involved in the message exchange. This system focuses on platform- and application-level security, so the two security levels are described.

1. Platform/transport-level (point-to-point) security

The transport channel between two endpoints (web service client and web service) can be used to provide point-to-point security as illustrated in Figure 2.



Figure 2. Platform/transport-level security

In the platform-level model, the client sends an XML format request to the web server. The XML message is not

encrypted by the client. When the message is transported in the transport channel, the network encrypts the entire data stream to make sure that the transport is secure.

This system uses a tightly coupled Microsoft Windows operating system environment. The Internet Information Server (IIS) provides basic, integrated and certificate authentication. The ASP.NET web service inherits some of the ASP.NET authentication and authorization features. The Secure Sockets Layer (SSL) is used to provide message integrity and confidentiality.

2 Application-level security

With application-level security, the application controls security with custom security features (Figure 3).

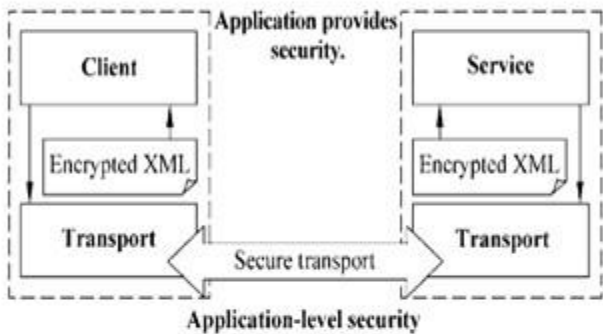


Figure 3. Application-level security

In the application-level security model, for example, an application can use a custom SOAP header to pass user credentials to authenticate the user with each web service request. A common approach is to pass a ticket (or user name or license) in the SOAP header. The application has the flexibility to generate its own principal object that contains roles. The application can optionally encrypt what it needs to, although this requires secure key storage and developers must have knowledge of the relevant cryptography APIs. An alternative technique uses SSL to provide confidentiality and integrity it with custom SOAP headers to perform authentication.

The system uses the SOAP Toolkit 2.0^[8, 9] offered by Microsoft, which provides support for internet security based on the IIS security infrastructure to implement the application-level security model.

C. The Implementation of the System

The whole system was built using the Microsoft .NET framework and .NET compact framework. Server application was implemented by Microsoft ASP.NET based on C#, the database was served by Microsoft SQL server 2000. The context server connected to the web server acted as IIS as the web server.

The function modules of this system mainly consist of 5 parts: system management, food management, client management, food ordering management and finance management, as shown in Figure 4.

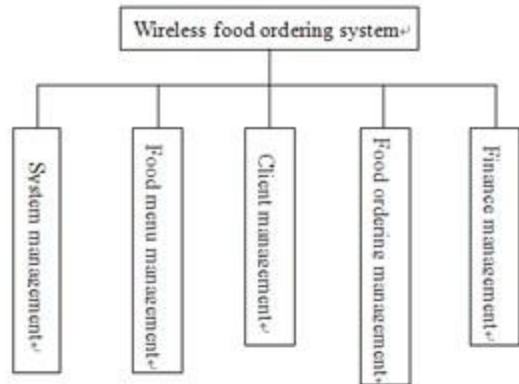


Figure 4. function modules of the system

The system management module is responsible for the initial setting of the system, administrator setting, wireless network setting, logs management etc. The food menu management includes setting name, prices, types, state of food, and so on. The client management supervises information of clients, which include the VIP information. The food ordering management is responsible for supervising the food ordering information from wire users and wireless users. The finance management administrates cash, bill, and financial audit of the restaurant. Some user interfaces of wire users and wireless users in this system are shown in figure 5 and figure 6.



Figure 5. A view of desktop PC



Figure 6. A view of PDA

D. Security strategies with mobile devices

Secure wireless connectivity between mobile terminal devices and the web server is an important aspect of this system. Some of the specific security threats associated with mobile devices include^[10]:

- Interception of data that passes over the wireless network
- Capture of data via wireless connections
- Mobile viruses

So the web services-based wireless food ordering system should be provided efficiently with a high level of security. In this system, we adopted the following strategies to ensure mobile security:

- The critical data are not stored permanently in the device. The application will delete data downloaded from the database before the mobile terminal is closed.
- Encryption. We use the WLAN encryption standard WEP (Wired Equivalence Privacy) to encrypt the data in transit.
- Individual authentication. The critical software on the system needs a username and password for use to implement individual authentication.
- MAC address filters. A MAC (Media Access Control) address is a unique identity burned into every network adapter during manufacture, with no way of changing it. Using this filter, the AP (Access Point) maintains a list of MAC addresses of mobile devices of the restaurant and only permits those devices on the list to connect to the server.

IV. CONCLUSIONS

The mobile devices have been widely used to provide easily access to the web content. We presented a wireless food ordering system based on web services over a wired/wireless integrated local area network, which implements wired and wireless data access to the servers and food ordering functions through both desktop PCs and mobile devices such as PDAs. The system is based on secure web service architecture and can increase efficiency for

restaurants by saving time, reducing human errors and by providing higher quality customer service.

ACKNOWLEDGMENT

The authors wish to thank the support by the Scientific Research Plan Projects of Jiangxi Education Department (No. GJJ09263) and the Jiangxi Province Natural Science Foundation (2007GZS0472)

REFERENCES

- [1] C. Branigan (2001), Wireless, handheld solutions prevail at NECC 2001, retrieved January 10, 2007 from <http://www.eschoolnews.com/news/showStory.cfm?ArticleID=2865>.
- [2] Watkins Demien. Mobile web services technical roadmap. http://www.microsoft.com/serviceproviders/mobilewebservices/mws_tech_roadmap.asp. 2003, 11.
- [3] Y. Xiang, W. Zhou and M. Chowdhury. Toward pervasive computing in restaurant, First International Conference on E-Business and Telecommunication Networks (ICETE 2004), Setubal, Portugal, August, 2004, pp312-317.
- [4] IEEE 802.11™ WIRELESS LOCAL AREA NETWORKS. IEEE Working Group (WG), <http://www.ieee802.org/11>.
- [5] Knikker, R., Guo, Y., Li, J.L., Albert Kwan, K.H., Yip, K.Y., Cheung, D.W., et al. A Web services choreography scenario for interoperating bioinformatics applications. BMC Bioinformatics 2004, 5:25-28.
- [6] FENSEL D, BUSSLER C. The Web service modeling framework WSMF. Electric Commerce Research and Application, 2002, Vol.1, No.2: pp113-137.
- [7] Meier J D, Mackman A, Dunner M, Vasireddy S. Web services security S. <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnnetsec/html/SecNetch10.asp>. 2002.
- [8] Powell Matt. Real SOAP security. <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnservice/html/service11212001.asp>. 2001.
- [9] Kirtland Mary. Secure web services using the SOAP toolkit. http://msdn.microsoft.com/archive/default.asp?url=/archive/en-us/dnaxml/html/websvcs_usingsoap.asp. 2001.
- [10] R. G. Duncan and M.M. Shabot "Secure remote access to a clinical data repository using a wireless personal digital assistant (PDA)". Proceedings of the American Medical Informatics Association Symposium, 2000, pp210-214.