**Boston International College**

**(Affiliated to Pokhara University)**

**Bharatpur-10, Chitwan**

**[Subject Code: PRJ439]**

**A MAJOR PROJECT ON**

“**CANTEEN MANAGEMENT-BOSTON BITES**”



**Submitted By:**

**Elis Kandel**

**Sagar Kandel**

**Pawan Poudel Sharma**

**Saroj Chettri**

**Submitted To:**

**Department of Computer Information System**

**Boston International College**

**Bharatpur-10, Chitwan**

**JULY, 2024**

## DECLARATION

We hereby declare that the project work report entitled “Canteen Management- Boston Bites”. Submitted for the BCSIT is our original work and the Project Work Report has not formed the basis for the award of any degree, diploma, or other similar titles.

Signature:

Name of the students:

Date:

## CERTIFICATE

This is to certify that the Project Work titled “Canteen Management” submitted by Elis Kandel, Sagar Kandel, Pawan Poudel Sharma, Saroj Chettri for the partial fulfillment of the requirements of BCSIT embodies the bonafide work done by them under my supervision.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Signature of the Supervisor

Name

Date

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Signature of the Internal Evaluator

Name

Date

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Signature of the External Evaluator

Name

Date

# ACKNOWLEDGEMENT

This project is prepared in the partial fulfillment of the requirement for the degree of Bachelor in Computer Science of Information and Technology (BCSIT). The satisfaction and success of completion of this task would be incomplete without heartfelt thanks to people whose constant guidance, support and encouragement made this work successful. On doing this undergraduate project we have been fortunate to have help, support and encouragement from many people we would like to acknowledge them for their cooperation.

Our first thanks go to Pokhara University for designing such a worthy syllabus and making us do this project. Then we would like to thank Boston International College for giving us such a great opportunity and providing all support and guidance which helped to complete the project on time. Our very sincere and heartfelt thanks go to Mr. Nabin Kumar Shrestha who is the Program Coordinator as well as our project supervisor who constantly guided us through the project time period. Without his guidance, our project would have been impossible.

Finally, we would like to express our sincere gratitude to our friends, families and faculty members for their support, cooperation and encouragement without whom the completion of this project would be impossible. This project has been a wonderful experience where we have learnt and experienced many beneficial things.

# ABSTRACT

**Canteen Management** is a holistic solution that provides campus food service with an opportunity to optimize and improve the work of all related processes. This system allows the integration of inventory management, orders, payments and reporting into one platform to ensure more convenient access of canteen staff and campus students to the services. Moreover, the use of cutting-edge technologies such as mobile applications and electronic payments systems help to eliminate existing barriers between consumers and the service and introduction of payment systems that are accountable, transparent and data-driven.

**Key Words:** Django, ReactJS.

Table of Content

[DECLARATION I](#_Toc172704442)

[CERTIFICATE II](#_Toc172704443)

[ACKNOWLEDGEMENT III](#_Toc172704444)

[ABSTRACT IV](#_Toc172704445)

[LIST OF ABBREVIATIONS V](#_Toc172704446)

[CHAPTER 1: INTRODUCTION 1](#_Toc172704447)

[1.1 Background 1](#_Toc172704448)

[1.2 Problem Statement 2](#_Toc172704449)

[1.3 Objectives 3](#_Toc172704450)

[CHAPTER 2: LITERATURE REVIEW 4](#_Toc172704451)

[CHAPTER 3: METHODOLOGY 5](#_Toc172704452)

[Requirements Analysis: 5](#_Toc172704453)

[System Design: 5](#_Toc172704454)

[Backend Development with Django: 5](#_Toc172704455)

[Frontend Development with React: 5](#_Toc172704456)

[Integration and Testing: 6](#_Toc172704457)

[Deployment and Maintenance: 6](#_Toc172704458)

[CHAPTER 4: EPILOGUE 7](#_Toc172704459)

[4.1 Project Result 7](#_Toc172704460)

[4.1.1 Output and Codes 7](#_Toc172704461)

Table Of Figures

[Figure 4. 1 Login Page 10](#_Toc172702321)

[Figure 4. 2 Home Page 14](#_Toc172702322)

[Figure 4. 3 Home Page-2 15](#_Toc172702323)

[Figure 4. 4 Order Page 20](#_Toc172702324)

[Figure 4. 5 Feedback Page 21](#_Toc172702325)

# LIST OF ABBREVIATIONS

API Application Programming Interface

App Application

BCIS Bachelor of Computer Information System

DOM Document Object Model

ERD Entity Relationship Diagram

GUI Graphical User Interface

GPU Graphical Processing Unit

IOS iPhone Operating System

MCQ Multiple Choice Question

NoSQL non-Structured Query Language

SDK Software Development Kit

UI User Interface

# CHAPTER 1: INTRODUCTION

## 1.1 Background

Institutions such as schools, universities, and corporate offices often have canteens or food service facilities to cater to the nutritional needs of their students, staff, or employees. These canteens play a crucial role in providing balanced meals that support the health and well-being of individuals within the institution. However, managing these canteens efficiently poses several challenges that require comprehensive solutions.

Inventory management is also a significant concern. Efficiently tracking and managing the stock of ingredients and supplies is vital to prevent food wastage and ensure that fresh ingredients are always available. This involves forecasting demand accurately, managing suppliers, and implementing effective storage solutions. A robust inventory management system can help in reducing costs, minimizing waste, and ensuring a consistent supply of high-quality food.

Time management is another critical factor. Efficiently managing the time taken for food preparation, serving, and cleanup is essential to ensure that customers receive their meals promptly, especially during peak hours. Streamlining kitchen operations, employing skilled staff, and utilizing technology for order management can significantly improve service speed and efficiency.

In conclusion, the need for effective canteen management is crucial not only for ensuring the well-being of individuals but also for promoting a healthy and productive environment within the institution. Implementing a comprehensive canteen management system that addresses menu planning, inventory management, and time management can lead to more efficient operations, better food quality, and higher levels of customer satisfaction.

## 1.2 Problem Statement

Despite the importance of canteen management, many institutions struggle with issues such as food wastage, long waiting times, lack of variety in menu options, and dissatisfaction among customers. These challenges can lead to financial losses and a negative impact on the overall reputation of the institution. Food wastage not only results in unnecessary costs but also raises environmental concerns, reflecting poorly on the institution's commitment to sustainability. Long waiting times can cause frustration and reduce productivity, as students and employees spend valuable time queuing instead of focusing on their academic or professional responsibilities. A lack of variety in menu options can lead to dissatisfaction and decreased patronage, especially among those with specific dietary needs and preferences.

Furthermore, dissatisfaction among customers can spread through word-of-mouth and social media, tarnishing the institution's image and potentially affecting student enrollment or employee retention. These issues underscore the need for a systematic approach to canteen management that incorporates innovative solutions and technologies to optimize operations. Addressing these challenges requires a multi-faceted strategy that includes efficient inventory management to reduce waste, streamlined food preparation processes to minimize waiting times, and diverse menu planning to cater to a wide range of tastes and dietary requirements.

Therefore, there is a pressing need to address these issues through the implementation of effective canteen management strategies. This involves leveraging technology, such as digital ordering systems and inventory management software, as well as adopting best practices in kitchen management and customer service. By tackling these problems, institutions can enhance the dining experience, reduce operational costs, and improve their reputation, ultimately contributing to a healthier and more productive environment.

## 1.3 Objectives

The main objectives of our project are given below:

* To create a website that streamlines the canteen management process and enhances user experience.
* To reduce the maximum time wastage of customers, students, and employees.
* To develop strategies for improving menu planning and inventory management.
* To assess the impact of the proposed canteen management framework on customer satisfaction and institutional outcomes.

# CHAPTER 2: LITERATURE REVIEW

Canteen management systems are essential for efficiently handling food services in various settings. Existing systems like NutriKids, EZCanteen, MealSuite, and Caterease offer a range of features that cater to different aspects of food service management, such as menu planning, online payments, and inventory management.

Key trends in canteen management systems include sustainability initiatives like reducing food waste and promoting eco-friendly practices. Mobile apps and digital payments are increasingly enhancing user experience by providing convenient and quick transaction options. These features cater to the growing demand for seamless and contactless interactions in canteens. However, with the rise of digital solutions, data security and compliance remain critical considerations. Ensuring that these systems comply with data protection regulations and safeguard user information is paramount to maintaining trust and reliability.

In conclusion, the evolution of canteen management systems reflects a growing emphasis on efficiency, sustainability, and user experience. By incorporating advanced technologies and focusing on specialized needs, these systems are transforming how food services are managed across various institutions.

# CHAPTER 3: METHODOLOGY

## Requirements Analysis:

* Identify key functionalities such as menu management, order processing, payment integration, user authentication, and reporting.
* Gather requirements from stakeholders including canteen managers, users (students/employees/patients), and administrators.

## System Design:

* Design database models for menus, orders, users, payments, and reporting using Django ORM.
* Create wireframes and UI mockups for the React frontend based on user interface design principles.

## Backend Development with Django:

* Set up Django project and apps for canteen management functionalities.
* Implement Django REST Framework for API development.
* Create models, serializers, views, and URL patterns for CRUD operations on menus, orders, users, and payments.
* Implement authentication and authorization using Django's built-in authentication system or third-party packages like Django REST Framework JWT.

## Frontend Development with React:

* Initialize React app and set up necessary dependencies (e.g., fetch for API calls, React Router for navigation).
* Develop components for menu display, order creation, user authentication, payment processing, and reporting.
* Implement forms with validation for user input (e.g., order quantities, payment details).
* Context API for state management, especially for user authentication functionality.

## Integration and Testing:

* Connect the React frontend to the Django backend APIs using Fetch API.
* Conduct integration testing to ensure data flow between frontend and backend components.
* Test user authentication, order creation, payment processing, and reporting functionalities.
* Implement unit tests and end-to-end tests using tools like Django's TestCase and React Testing Library/Jest.

## Deployment and Maintenance:

Deploy the application using platforms like Heroku, AWS, or Docker containers.

Set up continuous integration/continuous deployment (CI/CD) pipelines for automated testing and deployment.

Monitor application performance, handle bug fixes, and implement feature updates based on user feedback and evolving requirements.Technologies and Tools:

* Django for backend development and API creation.
* Django REST Framework for building RESTful APIs.
* React for frontend development and user interface.
* Redux or Context API for state management in React.
* Fetch API for API communication.
* PostgreSQL or SQLite for database management.

## 

# CHAPTER 4: EPILOGUE

## 4.1 Project Result

### 4.1.1 Output and Codes

import React, { useState } from "react";

import './login.css';

const Login = () => {

const [username, setUsername] = useState('');

const [password, setPassword] = useState('');

const [error, setError] = useState('');

const handleUsernameChange = (e) => {

setUsername(e.target.value);

};

const handlePasswordChange = (e) => {

setPassword(e.target.value);

};

const handleRedirect = () => {

window.location.href = '/home'; // Redirect to "/home" after successful login

};

const handleSubmit = async (e) => {

e.preventDefault();

if (!username || !password) {

setError('Please enter username and password');

return;

}

try {

const response = await fetch('http://127.0.0.1:8000/user/login/', {

method: 'POST',

headers: {

'Content-Type': 'application/json',

},

body: JSON.stringify({ username, password }),

});

const responseData = await response.json();

if ('token' in responseData) {

const token = responseData.token;

localStorage.setItem('token', token);

setError('');

console.log('Login successful');

handleRedirect(); // Redirect user after successful login

} else {

const errorData = responseData;

setError(errorData.message || 'Login failed');

}

} catch (error) {

console.error('Login error:', error);

setError('An error occurred during login');

}

};

return (

<div className="container">

<div>

<form onSubmit={handleSubmit} className="form">

<div className="form-group">

<label htmlFor="username">Username or Email:</label>

<input type="text" id="username" value={username} onChange={handleUsernameChange} />

</div>

<div className="form-group">

<label htmlFor="password">Password:</label>

<input type="password" id="password" value={password} onChange={handlePasswordChange} />

</div>

<button type="submit">Login</button>

{error && <div className="error-message">{error}</div>}

</form>

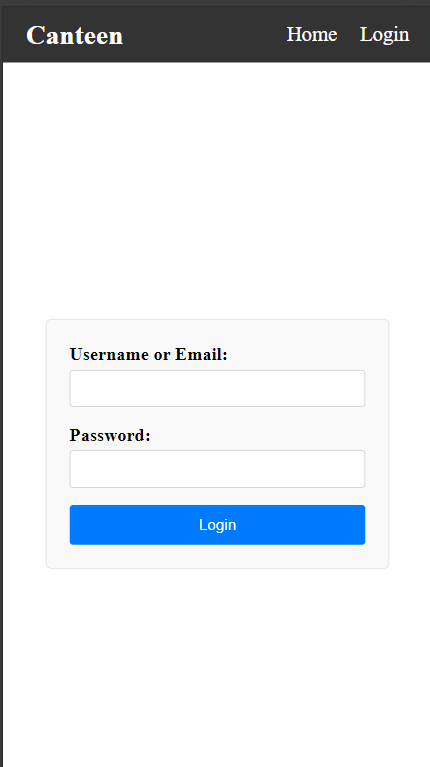
</div>

</div>

);

};

export default Login;



**Figure 4. 1 Login Page**

import React, { useState, useEffect } from 'react';

import './home.css';

import Samosa from './images/Samosa.webp';

const dailyProducts = {

Monday: [1, 2, 6],

Tuesday: [3, 4],

Wednesday:[],

Thursday:[4,5],

Friday:[1,2,3,4],

Sunday: [13, 14, 6,3, 4, 5],

};

const getCurrentDay = () => {

const daysOfWeek = [

"Sunday", "Monday", "Tuesday", "Wednesday",

"Thursday", "Friday", "Saturday"

];

const currentDayIndex = new Date().getDay();

return daysOfWeek[currentDayIndex];

};

const Home = () => {

const [items, setItems] = useState([]);

const token = localStorage.getItem('token');

useEffect(() => {

const fetchData = async () => {

try {

const response = await fetch('http://127.0.0.1:8000/product/');

if (!response.ok) {

throw new Error('Network response was not ok');

}

const data = await response.json();

const currentDay = getCurrentDay();

const productIdsForToday = dailyProducts[currentDay] || [];

const filteredProducts = data.filter(item => productIdsForToday.includes(item.id));

setItems(filteredProducts);

} catch (error) {

console.error('Error fetching data:', error);

}

};

fetchData();

}, []);

const addToOrder = async (itemId) => {

try {

if (!token) {

throw new Error('Authentication token not found');

}

const response = await fetch('http://127.0.0.1:8000/order/buy/', {

method: 'POST',

headers: {

'Authorization': `Token ${token}`,

'Content-Type': 'application/json',

},

body: JSON.stringify({ "product\_id": itemId, "quantity": 1 }),

});

if (!response.ok) {

throw new Error('Failed to add item to order');

}

// Handle responseData if needed

} catch (error) {

console.error('Error adding item to order:', error);

}

};

return (

<>

<h1>Items for {getCurrentDay()}</h1>

<div className="content-home">

{items.map(item => (

<div key={item.id} className='product-card' title={item.description}>

<img className="product-image" src={item.image} alt={item.name}></img>

<div className="product-details">

<h3>{item.name}</h3>

<p>Rs.{item.price}</p>

{token && ( // Conditional rendering for the "Buy" button

<p>

<button

style={{ cursor: 'pointer' }}

className='btn-buy'

onClick={() => addToOrder(item.id)}

>

Buy

</button>

</p>

)}

</div>

</div>

))}

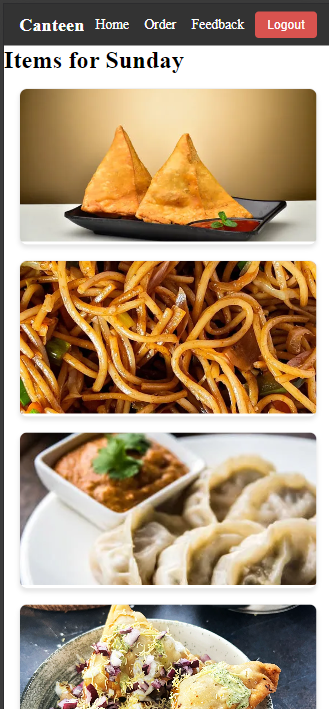
</div>

</>

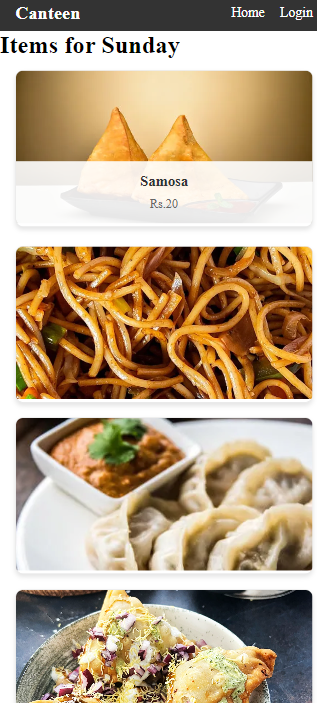
);

};

export default Home;



**Figure 4. 2 Home Page**



**Figure 4. 3 Home Page-2**

import React, { useEffect, useState } from "react";

import './order.css'; // Import your component-specific CSS file

const Order = () => {

const token = localStorage.getItem('token');

const [orders, setOrders] = useState([]);

useEffect(() => {

const fetchOrders = async () => {

try {

if (!token) {

throw new Error('Authentication token not found');

}

const response = await fetch('http://127.0.0.1:8000/order/', {

method: 'GET',

headers: {

'Authorization': `Token ${token}`,

'Content-Type': 'application/json',

},

});

if (response.ok) {

const data = await response.json();

setOrders(data);

} else {

const errorData = await response.json();

console.error('Fetch orders error:', errorData);

}

} catch (error) {

console.error('Fetch orders error:', error);

}

};

fetchOrders();

}, [token]); // Include token in dependency array to re-fetch orders when token changes

const deleteOrder = async (productId) => {

try {

const response = await fetch(`http://127.0.0.1:8000/order/delete/${productId}/`, {

method: 'DELETE',

headers: {

'Authorization': `Token ${token}`,

'Content-Type': 'application/json',

},

});

if (!response.ok) {

throw new Error('Failed to delete order');

}

// Handle successful deletion

console.log(`Order with productId ${productId} deleted successfully`);

// Update orders state by filtering out the deleted order

setOrders(prevOrders => prevOrders.filter(order => order.id !== productId));

} catch (error) {

// Handle errors

console.error('Error deleting order:', error.message);

}

};

const updateOrder = async (orderId, productId, newQuantity) => {

try {

const response = await fetch(`http://127.0.0.1:8000/order/update/${orderId}/`, {

method: 'PUT',

headers: {

'Authorization': `Token ${token}`,

'Content-Type': 'application/json'

},

body: JSON.stringify({ "product\_id": productId, "quantity": newQuantity }),

});

if (!response.ok) {

throw new Error('Failed to update order');

}

// Assuming the response contains the updated order

const updatedOrder = await response.json();

// Update the order in the state

setOrders(prevOrders => prevOrders.map(order =>

order.id === orderId ? updatedOrder : order

));

} catch (error) {

// Handle errors

console.error('Error updating order:', error.message);

}

};

return (

<div className="container-order">

<div>

<h2>Orders</h2>

<div className="order-list">

{orders.length > 0 ? (

orders.map(order => (

<div key={order.id} className="order-item">

<div className="order-info">

<p><strong>Product:</strong> {order.product.name}</p>

<p><strong>Quantity:</strong> {order.quantity}</p>

<div>

<button

style={{ cursor: 'pointer' }}

className='btn-update'

onClick={() => updateOrder(order.id, order.product.id, order.quantity + 1)}

>

+

</button>

<button

style={{ cursor: 'pointer' }}

className='btn-update'

onClick={() => updateOrder(order.id, order.product.id, order.quantity - 1)}

>

-

</button>

</div>

<p><strong>Price:</strong> {order.total\_price}</p>

<p><strong>Buyer:</strong> {order.user.username}</p>

</div>

<button onClick={() => deleteOrder(order.id)}>Remove Order</button>

</div>

))

) : (

<p>No orders available</p>

)}

</div>

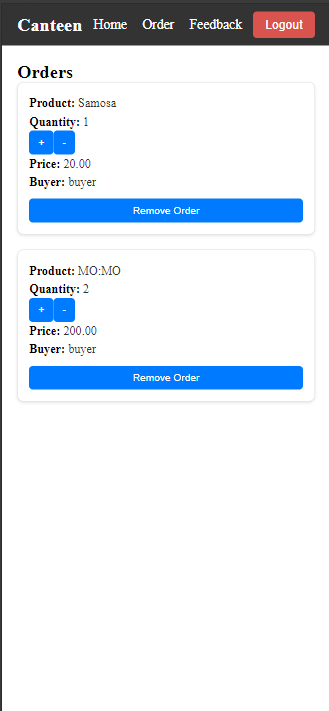
</div>

</div>

);

};

export default Order;



**Figure 4. 4 Order Page**

// Feedback.js

import React, { useState } from 'react';

import './feedack.css';

const Feedback = () => {

const [feedback, setFeedback] = useState('');

const handleChange = (e) => {

setFeedback(e.target.value);

};

const handleSubmit = (e) => {

e.preventDefault();

// Add code to submit feedback

console.log('Feedback submitted:', feedback);

// Clear the feedback input after submission

setFeedback('');

};

return (

<div className="container-feedback">

<h2>Feedback</h2>

<form onSubmit={handleSubmit} className="feedback-form">

<textarea

value={feedback}

onChange={handleChange}

placeholder="Enter your feedback here..."

></textarea>

<button type="submit">Submit Feedback</button>

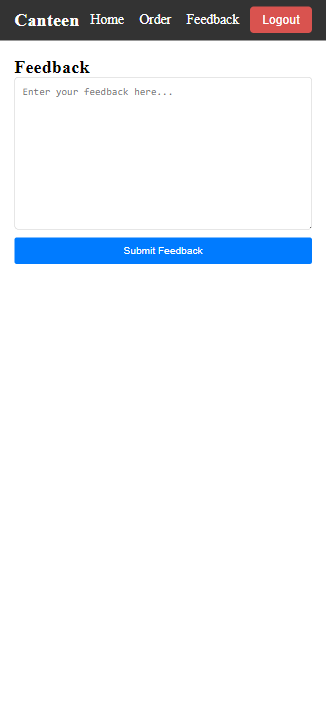
</form>

</div>

);

};

export default Feedback;



**Figure 4. 5 Feedback Page**