بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ

International Islamic University Chittagong

Department of Computer Science and Engineering

Project Report

Project Title: Restaurant Management System

Course Title: Computer Algorithm Lab

Course Code: CSE-2422

Semester: 4th

Submitted By

Name: Salsabil Tasnim
Student ID: C233445

Name: Rehab Binte Alamgir
Student ID: C233455

Name: Nusrat Jahan Chaity
Student ID: C233470

Submitted To

Miskatul Jannat Tuly
Lecturer
Department of Computer Science and Engineering
International Islamic University Chittagong

Submission Date: 07/08/2025

1. Abstract

This project, titled **Restaurant Management System** is a PHP-based web application developed using XAMPP to streamline the restaurant order management process. The system aims to improve efficiency in menu management, order placement, and budget-friendly food combo recommendations, offering both functional utility and algorithmic application in a web context.

The core functionality involves reading food menu data from a plain text file menu.txt and storing placed orders in orders.txt, eliminating the need for a complex database setup. Users are authenticated through a basic login system that provides session management and access control. Once logged in, users can browse a menu, place orders, and receive suggestions for food combinations that maximize satisfaction within their given budget.

The project stands out for its integration of classic computer science algorithms in a practical setting:

- **Merge Sorting**: Menu items are sorted by price using built-in PHP sorting functions to enhance user navigation.
- **Dynamic Programming (0/1 Knapsack)**: Provides the best full-item combo within the user's budget, ensuring optimal value selection without exceeding budget constraints.
- **Greedy Algorithm (Fractional Knapsack)**: Allows partial item inclusion to recommend the closest possible combo when full item selection is not feasible.
- The user interface is kept simple and accessible, focusing on functionality over aesthetics. It provides a clean environment where users can interact with the menu, place orders, and receive budget-based combo suggestions without confusion. All data interactions are handled through file input/output operations, showcasing efficient file-based data storage in the absence of a database.

Overall, this project demonstrates how algorithmic thinking and basic web technologies can be combined to solve real-world problems in a structured and meaningful way. It emphasizes hands-on experience sessions, handling, file management, and user interaction.

2. Table of Contents

3. Introduction

3.1 Background

The restaurant industry increasingly relies on technology to streamline operations and enhance customer experience. Managing menu items, taking customer orders, and handling billing are crucial daily operations that can be improved using computer systems. A web-based restaurant management system not only reduces manual errors but also improves order accuracy, inventory handling, and customer satisfaction.

3.2 Objective

The main objective of this project is to develop a file-based PHP restaurant system that:

- Displays a sorted menu from a text file.
- Allows customers to place and modify orders.
- Saves order history in a file.
- Suggests the best possible food combinations based on a given budget using **0/1 Knapsack (DP)** and **Fractional Knapsack (Greedy)** methods.

3.3 Scope

In Scope:

- File-based data handling for menu and orders.
- User logs in with session validation.
- Displaying, sorting, and selecting food items.
- Budget-based combo suggestion using algorithms.

Excludes:

- Integration with real-time databases (e.g., MySQL).
- Payment processing or inventory management.
- Multi-user login or admin dashboard.

4. Hardware and Software Requirements

4.1 Hardware

- Processor: Intel i3 or above

- RAM: 2 GB minimum
- Disk: 500 MB free space

4.2 Software

- XAMPP Server
- PHP 7.4+
- Text Editor (VS Code)
- Browser (Chrome/Firefox)

5. System Design

5.1 File Structure

- welcome.php
- index.php
- menu.txt
- orders.txt
- dp_result.txt
- greedy_result.txt
- summary.txt
- images/

5.2 Key Files

- **orders.txt**: Stores user orders.
- **menu.txt**: Stores menu items.
- **index.php**: Contains the ordering system, sorting logic, and algorithmic combos.

5.3 Features

This Restaurant Management System is a **file-based PHP project** that uses **sessions**, **form submissions**, and basic algorithmic logic (e.g., Sorting, Greedy, and Dynamic Programming) to allow a user to:

- Authetication
- View menu
- Place orders
- View and clear orders
- See the best food combos based on budget
- Save data to files

5.4 Purpose/Description of Files

File/Folder	**Purpose/Description**
welcome.php	Login page. Starts a session and authenticates the user.
index.php	Main interface after login. Displays menu, accepts orders, shows combos, and handles session and logic.
menu.txt	Stores menu items in format: id,name,price. Loaded by index.php.
orders.txt	Stores the user's current order history. Written each time an order is placed or cleared.
dp_result.txt	Stores the result of the Dynamic Programming (Knapsack) algorithm for the best combo within a budget.
greedy_result.txt	Stores the result of the Greedy Fractional Knapsack algorithm for the best combo within a budget.
summary.txt	Created dynamically when "Finish & Exit" is triggered. Contains a session summary, including orders, best combos, and totals.
images/	Contains the background image bg-main.jpeg and other UI images.

5.5 Data & Functionalities

5.5.1 Menu Data

```
1|Burger|150
2|Fried Rice|100
3|Chicken Roll|60
```

Each line represents a food item with:

- **ID** (for selection)
- **Name** (displayed to user)
- **Price** (used in total and combo calculation)

5.5.2 Order History

When a user places an order, it's saved like:

```
1|Burger|150|2
3|Chicken Roll|60|1
```

Each line represents:

- **ID**
- **Name**
- **Unit Price**
- **Quantity**

index.php uses this file to persist orders (even after refresh).

5.5.3 Key Functionalities in index.php

1. **Login Check (Session-based)**

```
if (!isset($_SESSION['logged_in']) ||
$_SESSION['logged_in'] !== true)
```

   If not logged in, it redirects to welcome.php.

2. **Load Menu**

```
$menu = [];
$lines = file($menuFile); // loads from menu.txt
```

   Stores menu as an array of associative arrays:

```
['id'=>1, 'name'=>'Burger', 'price'=>150]
```

3. **Place Order**
   When a user selects food + quantity and clicks "Add to Order", this block runs:

```
if (isset($_POST['add_order'])) { ... }
```

   - Finds food by ID from the menu
   - Adds to $_SESSION['orders']
   - Saves to orders.txt via saveOrdersToFile()

4. **Clear Orders**

```
if (isset($_POST['clear_orders'])) { ... }
```

   - Clears the session and order file.
   - Shows empty cart.

5. **Logout & Finish Order**
   - **Logout:** clears the session.
   - **Finish Order:** clears session, orders file, and shows thank you message.

5.5.4 Combo Calculators (Budget Optimization)

1. **Dynamic Programming (0/1 Knapsack)**

```
function dpKnapsack($menu, $budget)
```

- Tries every possible combination of whole food items.
- Chooses the best combo without exceeding the budget.
- Does not allow partial items.
- Time complexity: O(n×W)

2. **Greedy Algorithm (Fractional Knapsack)**

```
function dpKnapsack($menu, $budget)
```

- Picks full items until the budget is low.
- Then picks a fraction of the next best item.
- Fast but not always optimal (approximation).
- Time complexity: O(n log n)

5.5.5 Display Sections in HTML

Each section is cleanly styled:

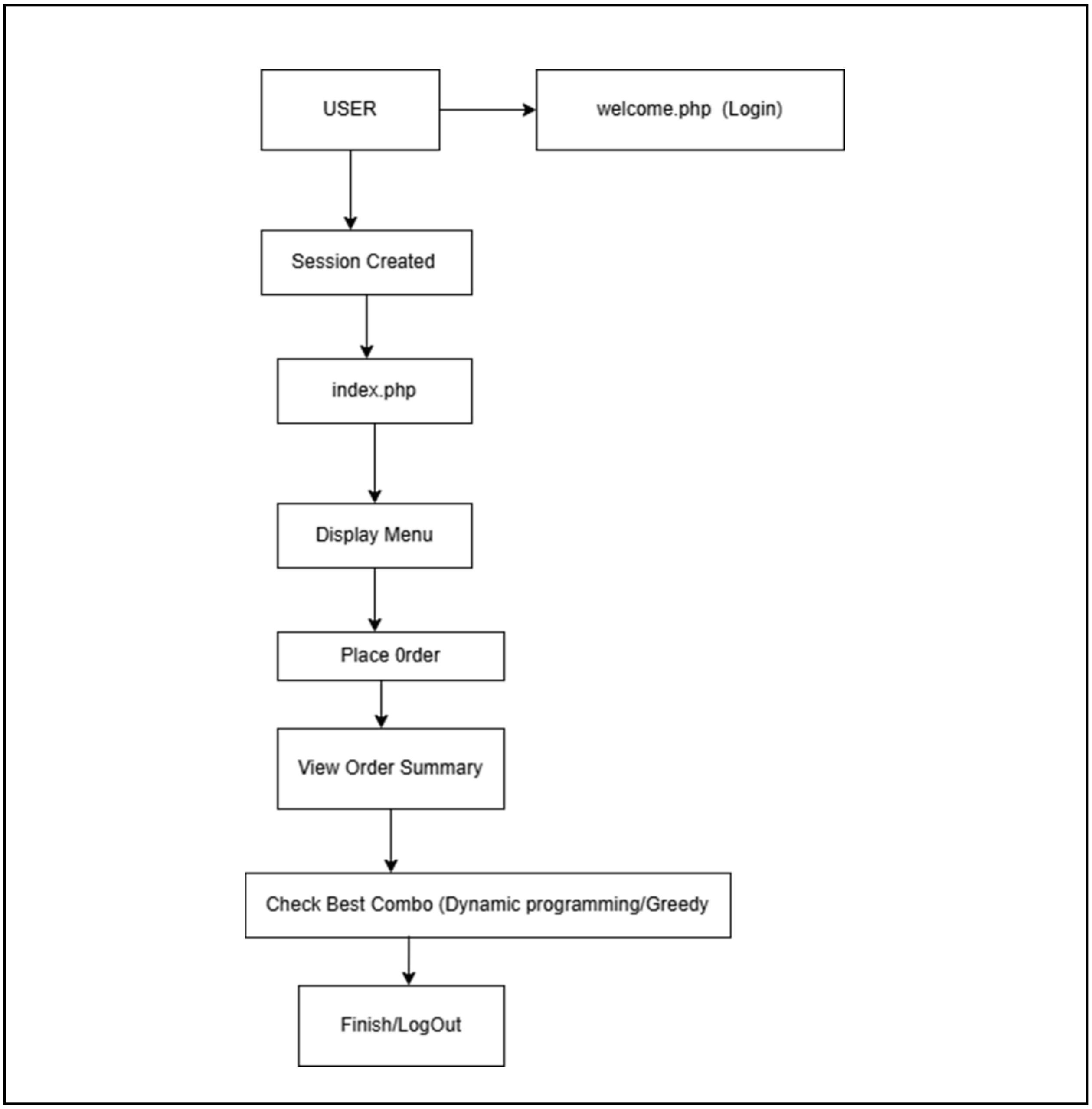
Section Title	**Purpose**
"Display Menu"	Table view of all food items sorted by ID.
"Place Order"	Form dropdown to select the item and quantity. The menu is sorted by price.
"Your Orders"	Shows ordered items, quantity, total per item, and total.
"Best Combo (Whole Items)"	Shows the result from Dynamic Programming.
"Best Combo (Partial)"	Shows the result from the Greedy algorithm.
"Finish & Exit"	Ends session with a nice thank-you message.

5.6 Algorithm Highlights

Algorithm	**Used For**	**Code Part**
Merge Sort	Menu sorting by price	mergeSortMenu()
Greedy (Fractional)	Best partial combo suggestion	greedyFractionalKnapsack()
Dynamic Programming	Best full-item combo	dpKnapsack()

6. Project Flow Diagram

6.1 Flow Diagram



6.2 Step-by-Step Explanation

1. **User accesses welcome.php**
   → Enters login info → If valid → a session is started.

2. **Redirected to index.php**
   → Main dashboard after login.
3. **Inside index.php**:
   - Menu is loaded from menu.txt.
   - User can:
     - View food list
     - Add items to the order
     - See the total order summary
     - Enter the budget to get the best combo
       - **Dynamic Programming** → Best full items
       - **Greedy** → Fast but allows partial items

7. Implementation Details

- **Login System**: The login system is implemented using PHP sessions. User credentials are hardcoded (user as username and 12345 as password). Upon successful login via welcome.php, users are redirected to index.php.
- **Menu Display**: Menu data is read from menu.txt, which contains comma-separated values for item ID, name, and price. This data is parsed and displayed in a styled HTML table.
- **Order System**: Users select menu items and quantities through a form. The selected order is stored in PHP $_SESSION variables to retain state during the session. When the order is submitted, it is saved permanently to orders.txt using file handling operations.
- **Sorting**: The menu items are sorted by price in ascending order using merge sort, ensuring that cheaper items appear first for better combo optimization.
- **Divide and Conquer**: Merge sort is used to sort the menu items by price in ascending order. Merge sort is a divide-and-conquer algorithm, which means it breaks down a problem into smaller subproblems, solves them independently, and then combines their results to solve the original problem.
- **Dynamic Programming (0/1 Knapsack)**: This algorithm is used to suggest the best combination of full menu items such that the total price does not exceed the given budget. It considers all possible combinations and selects the one with the highest total value within constraints.
- **Greedy Algorithm (Fractional Knapsack)**: This method is used when partial items can be considered. It sorts items by value-to-cost ratio and selects items greedily until the budget is exhausted. This may include fractional selection (e.g., half a pizza) for optimal usage of the budget.

8. Output Analysis

8.1 Feature Analysis

- **Login Page**: Prevents unauthorized access to the system and ensures session-based security.
- **Menu Display**: A clear, sorted list of available menu items is presented to the user, improving ease of selection.
- **Order Summary**: Once an order is placed, a summary showing item names, quantities, unit price, and total price is displayed.
- **Best Combo Suggestions**:
  - DP Result: Shows a combination of full items that maximizes value under the given budget.
  - Greedy Result: Displays partial selections if the budget cannot cover all the items.
- **Example Output for Budget TK 300**:
  - DP: Burger, Fries, Ice Cream — Total = 290 TK
  - Greedy: Pizza (80%), Fries (100%) — Approx. 300 TK

8.2 Screenshots

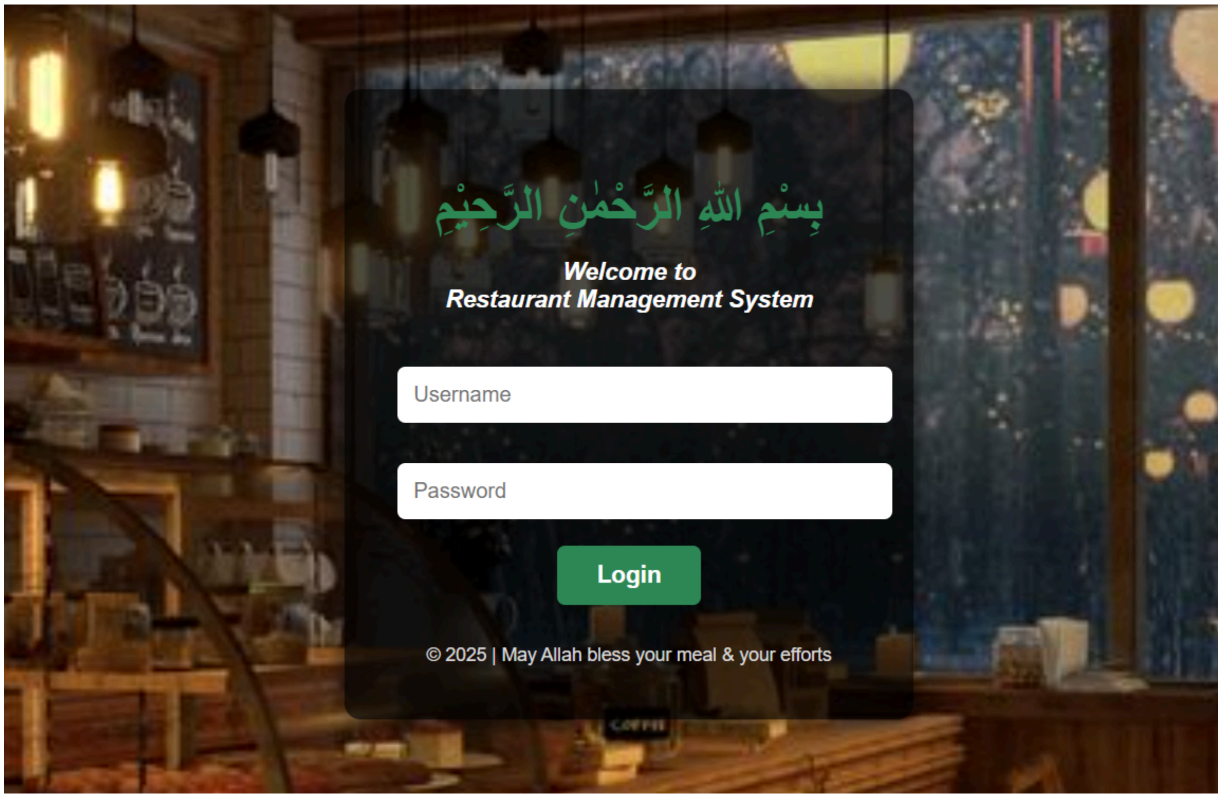
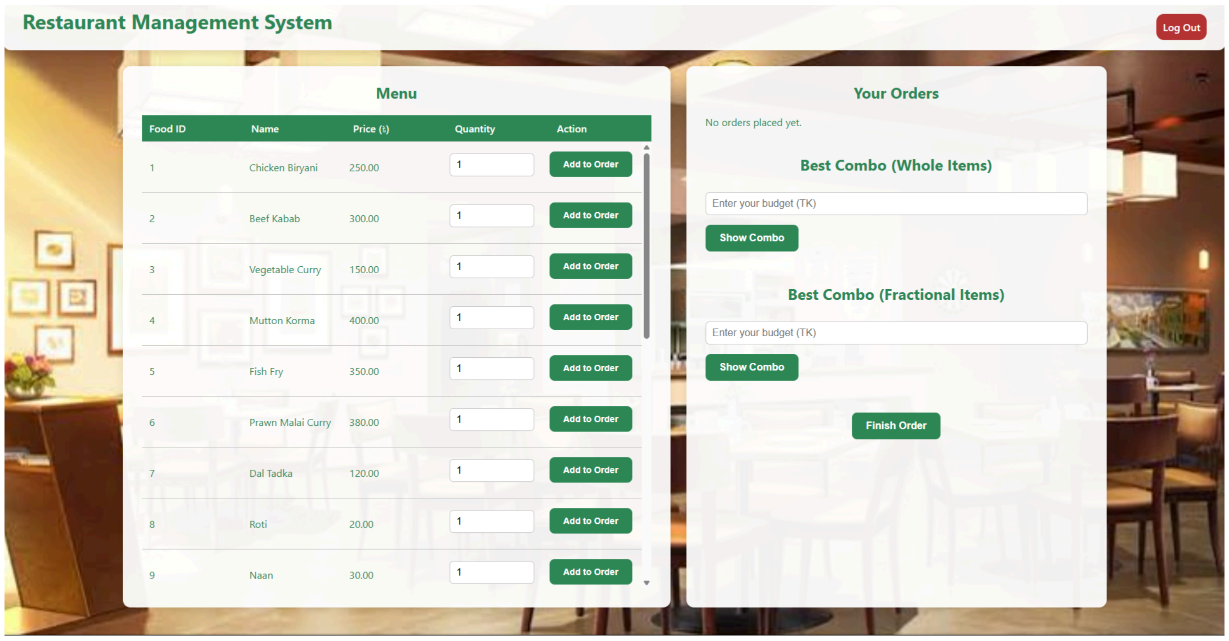

بِسْمِ اللّٰهِ الرَّحْمٰنِ الرَّحِيْمِ
Welcome to
Restaurant Management System
Username
Password
Login
© 2025 | May Allah bless your meal & your efforts

Fig 1: Login Page



Menu

Food ID	Name	Price (৳)	Quantity	Action
1	Chicken Biryani	250.00	1	Add to Order
2	Beef Kabab	300.00	1	Add to Order
3	Vegetable Curry	150.00	1	Add to Order
4	Mutton Korma	400.00	1	Add to Order
5	Fish Fry	350.00	1	Add to Order
6	Prawn Malai Curry	380.00	1	Add to Order
7	Dal Tadka	120.00	1	Add to Order
8	Roti	20.00	1	Add to Order
9	Naan	30.00	1	Add to Order

Your Orders

No orders placed yet.

Best Combo (Whole Items)

Enter your budget (TK)

Show Combo

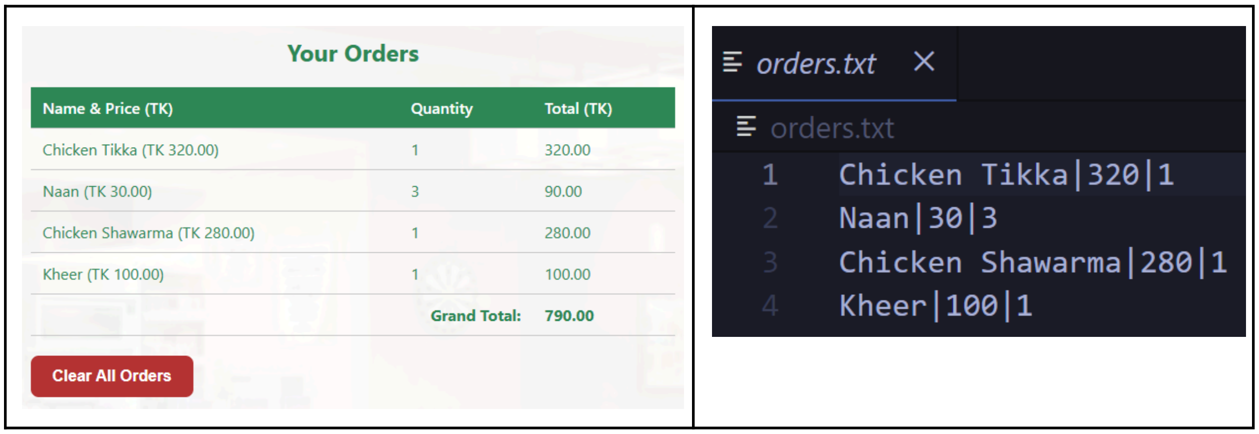
Best Combo (Fractional Items)

Enter your budget (TK)

Show Combo

Finish Order

Fig 2: Main Dashboard



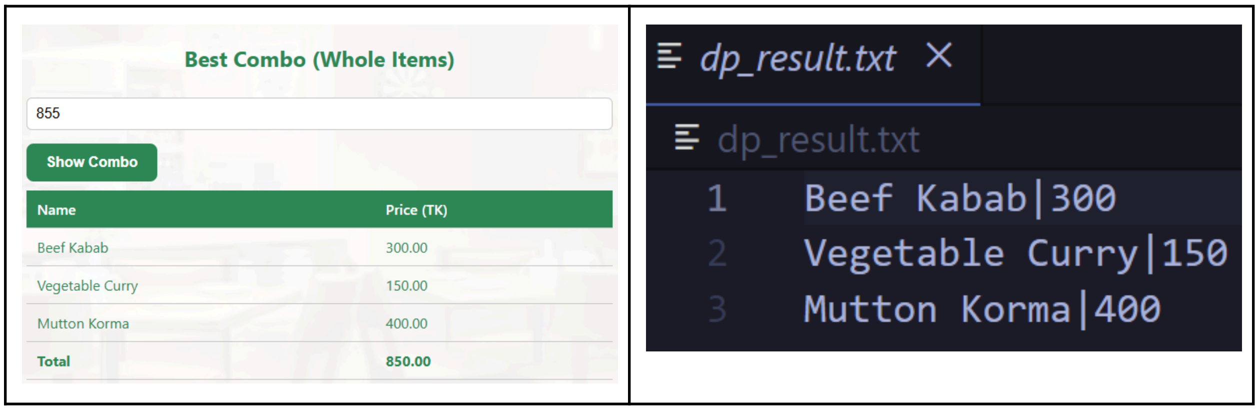
Your Orders

Name & Price (TK)	Quantity	Total (TK)
Chicken Tikka (TK 320.00)	1	320.00
Naan (TK 30.00)	3	90.00
Chicken Shawarma (TK 280.00)	1	280.00
Kheer (TK 100.00)	1	100.00
	Grand Total:	790.00

orders.txt

```
Chicken Tikka|320|1
Naan|30|3
Chicken Shawarma|280|1
Kheer|100|1
```

Fig 3: Adding Orders (including **orders.txt**)



Best Combo (Whole Items)

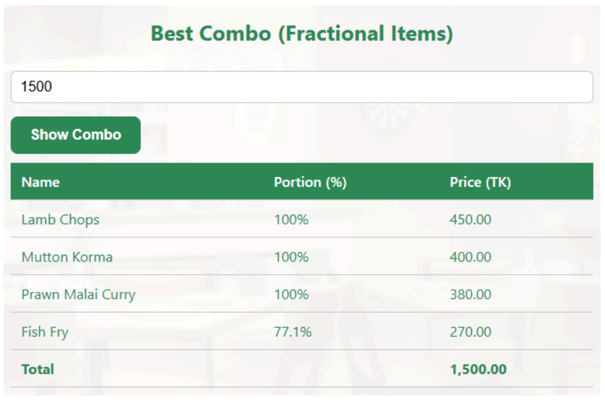
855

Name	Price (TK)
Beef Kabab	300.00
Vegetable Curry	150.00
Mutton Korma	400.00
Total	**850.00**

dp_result.txt

```
Beef Kabab|300
Vegetable Curry|150
Mutton Korma|400
```

Fig 4: Best Combo - Whole Items (including **dp_result.txt**)

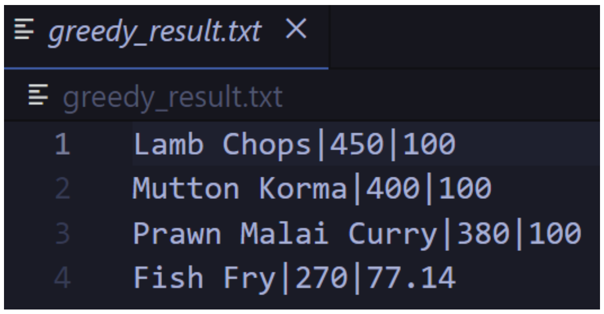


Best Combo (Fractional Items)

1500

Show Combo

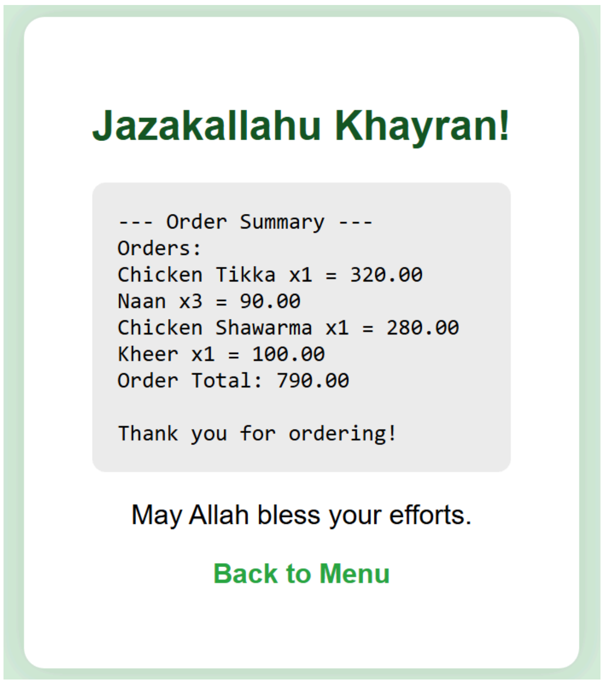
Name	Portion (%)	Price (TK)
Lamb Chops	100%	450.00
Mutton Korma	100%	400.00
Prawn Malai Curry	100%	380.00
Fish Fry	77.1%	270.00
Total		1,500.00



greedy_result.txt

```
Lamb Chops|450|100
Mutton Korma|400|100
Prawn Malai Curry|380|100
Fish Fry|270|77.14
```

Fig 5: Best Combo - Fractional Items (including **greedy_result.txt**)



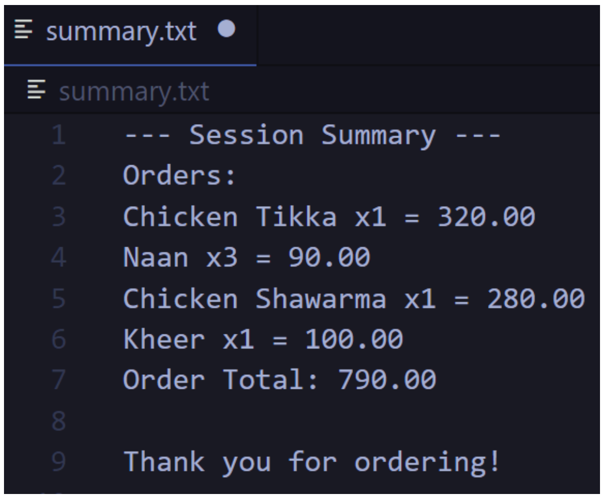
Jazakallahu Khayran!

```
--- Order Summary ---
Orders:
Chicken Tikka x1 = 320.00
Naan x3 = 90.00
Chicken Shawarma x1 = 280.00
Kheer x1 = 100.00
Order Total: 790.00

Thank you for ordering!
```

May Allah bless your efforts.

Back to Menu



summary.txt

```
--- Session Summary ---
Orders:
Chicken Tikka x1 = 320.00
Naan x3 = 90.00
Chicken Shawarma x1 = 280.00
Kheer x1 = 100.00
Order Total: 790.00

Thank you for ordering!
```

Fig 6: Order Summary (including **summary.txt**)

9. Challenges Faced and Future Scope

9.1 Challenges

- Implementing and integrating algorithm logic (knapsack) within procedural PHP code structure.
- Handling PHP session variables across multiple pages while ensuring session timeout and security.

- Performing consistent file read/write operations without data loss, especially under concurrent access.
- Avoiding database usage while maintaining efficiency and scalability.

9.2 Future Scope

- Migrate to a MySQL database for better scalability, integrity, and query efficiency.
- Introduce multiple user roles, such as Admin (can add/edit menu) and Customer (can place orders).
- Implement real-time inventory and order status tracking using AJAX or WebSocket.
- Develop a mobile-friendly UI using responsive design (CSS Media Queries / Bootstrap).

10. Conclusion

This project serves as a comprehensive demonstration of how algorithmic problem-solving techniques can be seamlessly integrated into a real-world restaurant management system using PHP and basic file handling mechanisms. Unlike traditional static systems, this project brings a dynamic flavor by incorporating computational logic directly into the workflow of a restaurant scenario, including features like menu sorting, order placement, login validation, and most notably, budget optimization using well-known algorithms like Greedy (Fractional Knapsack) and Dynamic Programming (0/1 Knapsack).

From a UI/UX perspective, the project employs a simple yet functional interface that allows users to interact with the system through login forms, menus displayed from text files, and real-time order submissions. The use of file-based storage (menu.txt, orders.txt) not only simplifies data persistence but also keeps the system lightweight and easy to understand for beginners.

What sets this system apart is its **integration of core algorithmic logic into everyday tasks**, such as selecting affordable food combos based on a user's budget. The Greedy algorithm quickly identifies cost-effective choices, while Dynamic Programming ensures optimal combinations when dealing with fixed budget constraints.

These approaches reflect real-world decision-making scenarios and show how theoretical algorithms can solve practical problems.

This educational model offers a hands-on learning platform for students and newcomers to both web development and algorithm design. By engaging with the project, learners gain exposure to:

- Real-life application of sorting and optimization algorithms.
- Data handling through PHP and file I/O operations.
- Designing systems that balance usability with intelligent backend logic.

Ultimately, this project bridges the gap between theoretical computer science concepts and their **practical implementation**, making it an ideal introduction to **systems development**, **problem-solving**, and **algorithmic thinking** in the context of web-based applications.

11. References

1. PHP Manual, "Official Manual," The PHP Group. [Online]. Available: https://www.php.net/manual. [Accessed: Aug. 5, 2025].
2. W3Schools, "PHP Tutorial," W3Schools. [Online]. Available: https://www.w3schools.com/php/. [Accessed: Aug. 5, 2025].
3. GeeksforGeeks, "Knapsack Problem," GeeksforGeeks. [Online]. Available: https://www.geeksforgeeks.org/knapsack-problem/. [Accessed: Aug. 5, 2025].
4. XAMPP Documentation. [Online]. Available: https://www.apachefriends.org/docs/. [Accessed: Aug. 5, 2025].
5. Stack Overflow, "Discussions on PHP File Handling," Stack Overflow. [Online]. Available: https://stackoverflow.com/tags/php/info. [Accessed: Aug. 5, 2025].

12. Appendix

12.1 Setup Instructions

1. Clone the Project (https://github.com/nusratjahanchaity/Restaurant-Management-System)
2. Install XAMPP
3. Move the Project folder to ‘xampp/htdocs/’
4. Start Apache in XAMPP
5. Run the Project (http://localhost/Restaurant-Management-System)