National University of Computer & Emerging Sciences
Karachi Campus

Project Proposal

Computer Organization & Assembly Language

Section: BAI-9A

Restaurant Management System

Group Members:

22k-4083 Arooba Faisal

22k-8714 Amanullah Kazi

22k-4067 Moosa Memon

Restaurant Management System

1. Introduction:

This project is a **Restaurant Management System** created in **Assembly Language** as part of our course of Computer Organization and Assembly Language. Imagine walking into a fast-food restaurant, browsing through a menu, placing your order, and receiving a bill at the end—this system brings that entire experience to life, but in a digital, text-based format. It's designed to simulate the process of ordering food, selecting items from a menu, and generating a final bill, all while showcasing the power and precision of low-level programming.

At its core, the project is a demonstration of how even the most basic programming languages, like Assembly, can be used to create functional and practical applications. It highlights essential programming concepts such as **file handling** (saving and retrieving order details), **input/output operations** (interacting with the user), and **basic arithmetic** (calculating the total bill). While Assembly Language is often considered complex and challenging, this project proves that it can be used to build something as relatable and everyday as a restaurant ordering system.

2. Objectives:

- To create a functional restaurant management system that allows users to place orders and generate bills.
- To demonstrate proficiency in Assembly Language programming, including file handling, string manipulation, and user input/output.
- To simulate a real-world application using low-level programming techniques.
- To provide a user-friendly interface for ordering food items and viewing the final bill.

3. Project Scope:

The project is a console-based application that simulates a fast-food restaurant's ordering system. It includes the following features:

- **Menu Display**: The system displays a menu with various food items, including combos, drinks, and sides.
- **Order Placement**: Users can select items from the menu, specify quantities, and choose drink options.
- **Billing System**: The system calculates the total bill based on the selected items and quantities.
- **File Handling**: The system generates a text file containing the final bill, including the order number, customer name, and total amount.

- **Error Handling**: The system includes basic error handling to manage invalid user inputs.

4. System Components:

The project consists of the following key components:

- **Main Menu**: Displays the available food items and prompts the user to make a selection.
- **Order Processing**: Handles the selection of items, quantity, and drink options.
- **Billing Module**: Calculates the total cost of the order and displays the final bill.
- **File Generation**: Creates a text file with the order details, including the order number, customer name, and total bill.
- **User Input/Output**: Manages user interactions, including input validation and display of messages.

5. Technical Details:

- **Programming Language**: Assembly Language (x86 architecture)
- **Development Environment**: The project is developed using the **Irvine32 library**, which provides functions for input/output, string manipulation, and file handling.
- **File Handling**: The system reads and writes to text files to store and retrieve order details.
- **User Interface**: The interface is text-based, with prompts and messages displayed in the console.

6. Key Features:

- **Combo Meals**: Users can select from various combo meals, each with different drink options.
- **Individual Items**: Users can order individual items such as nuggets, drinks, and fries.
- **Quantity Selection**: Users can specify the quantity of each item they wish to order.
- **Drink Options**: Users can choose from different drink options for combo meals.
- **Final Bill**: The system generates a detailed bill, including the total cost, and saves it to a text file.

7. Challenges:

- **Low-Level Programming**: Assembly Language requires careful management of memory, registers, and stack operations, which can be challenging.
- **User Input Handling**: Ensuring that the system correctly handles and validates user input is critical to avoid errors.
- **File Handling**: Managing file operations in Assembly Language requires precise control over file pointers and buffers.

8. Future Enhancements:

- **Graphical User Interface (GUI)**: Implementing a GUI to make the system more user-friendly.
- **Database Integration**: Integrating a database to store and retrieve order history and customer details.
- **Advanced Error Handling**: Implementing more robust error handling to manage unexpected user inputs and system errors.
- **Multiple Payment Options**: Adding support for different payment methods, such as credit card or mobile payments.

9. Conclusion:

This project demonstrates the application of Assembly Language in creating a functional restaurant management system. It highlights the use of low-level programming concepts to build a system that simulates real-world operations. The project serves as a valuable learning experience in understanding the intricacies of Assembly Language and its potential applications in system-level programming.

10. Deliverables:

- **Source Code**: The complete Assembly Language code for the restaurant management system.
- **Documentation**: A detailed report explaining the project's design, implementation, and challenges.
- **Test Cases**: Sample inputs and outputs to demonstrate the system's functionality.
- **Final Bill File**: A sample text file generated by the system containing the order details and total bill.