Documentation

- ## Problem Statement –

  Self-ordering systems are fast becoming popular in restaurants. Customers can browse the entire menu at their own pace, an option preferred by many customers, rather than being hurried by a cashier. Customers can use the system to submit food orders autonomously.
  This allows business to manage orders easily and reduces customers wait time.

- ## Requirements

  Operating System: Windows 98, Windows XP, Windows7 and above, Linux
  Browser: Any of Mozilla, Opera, Chrome etc
  Hardware Requirements: Processor Pentium III 630MHz RAM 128 MB Hard disk 20 GB Monitor 15" color monitor Keyboard 122 keys

- ## Technologies Used

  Language: Python, HTML, CSS, MD Bootstrap
  Development Kit: Django Scripting Language Enable Json
  Database: SQL Lite3

- ## Project Description

  The structure of the system can be divided into 3 main logical components:
  - Web Ordering System- provides the functionality for customers to place their order and supply Necessary details.
  - Menu Management-allows the restaurant to control what can be ordered by the customers
  - Order Retrieval System-This is a final logical component. Allows restaurant to keep track of all Orders placed. This component takes care of order retrieving and displaying order information.

- **Self Ordering System Module**

  This module provides the functionality for customers to place their order and supply necessary details. Users of the system, namely restaurant customers, must be provided the following functionality:
  - Manage their account.
  - Log in to the admin page.
  - Navigate the restaurant's menu.
  - Select an item from the menu.
  - Add an item to their current order.
  - Provide payment details.
  - Place an order to dine-in or take-away.
  - Receive confirmation in the form of an order number.
  - View order placed.
  - Mark as completed in restaurant billing page.

- ## Scope

In a nutshell, it can be summarized that the future scope of the project circles around maintaining information regarding:
●We can add printer in future.
●We can give more advance software for Self-Food Ordering System including more facilities.
●We will host the platform on online servers to make it accessible worldwide.
●Integrate multiple load balancers to distribute the loads of the system
●Create the master and slave database structure to reduce the overload of the database queries
●Implement the backup mechanism for taking backup of codebase and database on regular basis on different servers.

- ## Challenges Faced

The software presents a broad range of options to its users some intricate options could not be covered into it; partly because of logistic and partly due to lack of sophistication. Paucity of time was also major constraint; thus, it was not possible to make the software fool proof and dynamic.

Considerable efforts have made the software easy to operate even for the people not related to the field of computers but it is acknowledged that a layman may find it a bit problematic at the first instance. The user is provided help at each step for his convenience in working with the software.

- ## Conclusion

At the end it is concluded that we have made effort on following points…
●A description of the background and context of the project and its relation to work already done in the area.
●We define the problem on which we are working in the project.
●We describe the requirement Specifications of the system and the actions that can be done on these things.
●We understand the problem domain and produce a model of the system, which describes operations that can be performed on the system.
●We included features and operations in detail, including screen layouts.
●We designed user interface and security issues related to system.
●Finally, the system is implemented and tested according to test cases.