System Design Document

Kitchen Inventory Management System

D & J Tech Solutions

# Architecture

## Web Framework

The Django framework is designed around the model-template-view. It is designed to separate concerns and make less code do more with consistent quality.

The model layer is a class that represents the data structure. This allows the user to configure the models to the database and then Django allows us to manipulate our data structures by accessing them via the models.

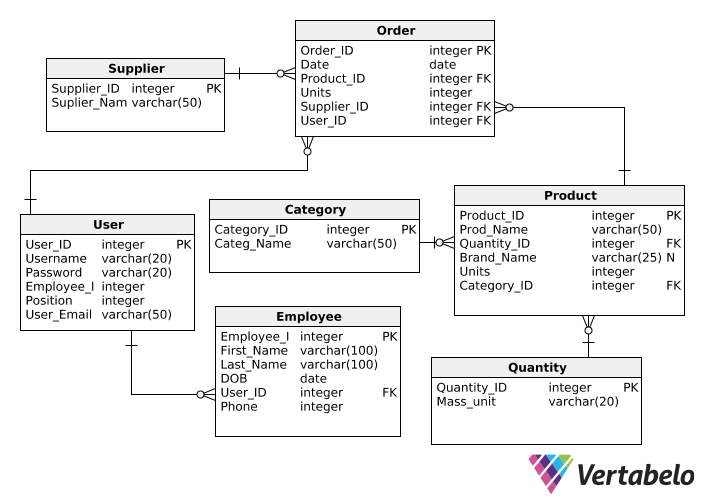
The view layer handles requests with data from a model. Accessing a specific value of data from an entity is possible by the view layer which communicates with the model layer to access that value from an entity in the database.

The template is what allows us to see our database values by rendering them into HTML and displaying it as a consistent template. Each time new data is called it is turned into html and presented in the same manner as determined by the HTML structure of the template layer.

The URL dispatcher matches with a view by matching the requested URL pattern that represents a specific page in the website.

# Data Access Design

## Model Design (Data Model)

**

## Functional Decomposition

User Stories:

1. Look up employee by last name.

* Employee table must include first and last\_name, as well as date of birth, contact details and position. This allows the employee to be found by the userID unique value PK.

1. View all products in category “dry goods”

* The entity Category can have several Products but each product can only have one category. This allows searching for a product to be done more quickly as you can have a large number of products.

1. User logs into account
   * User needs a username and password to access the web application

## Database Investigation

SQLite – We chose to use SQL as our database management system simply because this is a low traffic application that has minimal functionality. With there only being a as many users as there are staff in the restaurant, we do not require a more robust DBMS while still providing us with the functionality we need.

# Security Design

## Framework Security

Our code will be outside of the Web servers foot to ensure it cannot be viewed as plain text.

## Security Mechanisms

Access Control

* User authentication and Authorization – To log-in to the application. UserID and passwords are required. A user must be an Admin to access employee data. All users are able to access product data.

Application uses other access security principles such as separation of duties, limited attack vectors and least privilege.

## User Roles and Permissions

**Admin (Headchef)** – in our case the Head Chef and Owner will take on the admin role. The Admin is able to view the employee data base for management purposes. All users including the admin are able to view and manipulate the product database.

**Employee** – Employees are only able to view the product database as it is necessary for their job. The are not authorized to view the employee database.

# User Interface Design

## View Design (UI List)

*Home*

*Order*

*Employee*

### User Story 1

The sous chef is running uses carrots. They edit the quantity in stock and then add more carrots to the next order. Home.html will allows the sous chef to edit the quantity in stock and add more to the next order. Order.html will display everything that has been added to the order so they can confirm their order has been added to the order list.

### User Story 2

Admin account wants to call a staff member because they have not shown up for their shift. The admin will use employee.html to view staff members and click their name for their information to pop up in a modal.

## UI Design

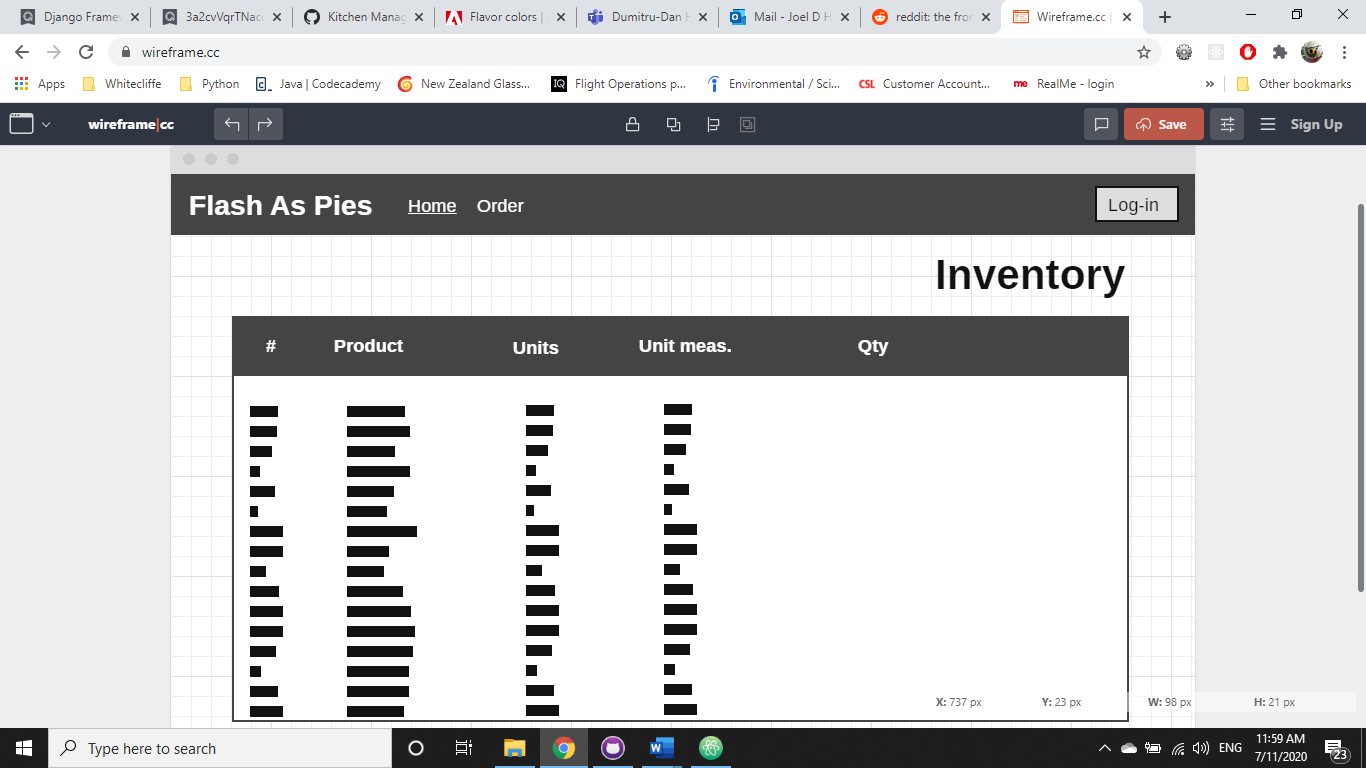
High contrast, easy to read user interface

Obvious functionality

Bootstap used in HTML.

Black, White and Blue color scheme.

* Creates easy readability
* Blue is commonly used for log-in buttons, creates familiar look and feel previous used web apps for users.

Landing page wireframe