Analysis of a Food Delivery Website Project Using Django and Python

Checklist:

Funcional Requirements:

User Authentication Implemented

Choose from a menu and submit order to the restaurant.

Create a user-friendly landing page, about page and ordering page,

Create an ordering system and have the ability to capture user information.

Create a dashboard for employees to be able to track the order status of the meal.

Integrate a payment system using paypal's api.

Have the ability to send email confirmations to users.

Admins have the ability to change menu items.

Introduction:

This analysis focuses on the software development project that centres around a food delivery website built using the Django framework and implemented in the Python programming language. The aim of this project was to create an efficient and user-friendly platform for customers to order food from this website. Django, with its rapid development capabilities and robust features, was chosen to streamline the development process and ensure a scalable and maintainable solution.

1. Project Overview:

The Dala delivery website is a comprehensive platform that integrates various modules, including user authentication, restaurant management, order processing, and payment handling. The system was geared towards connecting the customer to the restaurant to enable them order a meal remotely.

2. Django Framework:

The Django framework was chosen because enables the development of the food delivery website by providing a structured approach to web application development. Models define the data structures, Views handle user interactions and presentation logic, and Controllers manage data flow. The project utilizes Django's ORM for efficient database interactions, ensuring seamless integration between the application and the underlying data.

Django's built-in features, such as the authentication system, were essential to the project's security because of the ability to manage user roles and access control. Additionally, Django's templating engine was effective when performing HTML rendering, and creating dynamic, responsive user interfaces.

3. Python:

Python's versatility and readability made it an ideal choice for this project. The language's extensive ecosystem of libraries and frameworks enhances development speed and allows for the integration of third-party components. Python's simplicity also promotes collaboration among developers, ensuring a coherent and maintainable codebase.

4. Database Design and Management:

The Django server instance was designed to capture information related to users, orders, and payments. Migrations were employed to manage database schema changes efficiently, ensuring data consistency and integrity. The restaurant menu was integrated via Django admin page.

5. Testing and Quality Assurance:

Testing was an integral part of the development process, with the project implementing various testing levels, including integration tests, and code quality tests. Django's testing framework aids in the creation of comprehensive test suites, ensuring the reliability and correctness of the application. Continuous integration practices were employed to automate testing, contributing to the overall quality of the code. The use of SonarCloud and SonarQube build was also instrumental in analyzing the cleanliness of the code.

6.Challenges:

Integration with various api’s for tedious and created many errors.

Sometimes the pages would not switch seamlessly because of errors in creating views or having the various classes not imported into the web page.

Integration with SonarQube remotely proved to be challenging as the file format containing the private key(.pem) was difficult to decode.

7.Future Plans:

Integration of external database to allow for more restaurants be featured.

Improve the reliability of the code.