Pratibodh
A Journal for Engineering
A free and Open Access Journal
Homepage: https://pratibodh.org
RATNA & BM RANVEER Foundation

Analysis On Food Notification Web App
Design And Development of Python and Django App

Sakshi Jain[1], Akashat Jain[2], Sarthak Jain[3], Manju Vyas[4]
Department of Artificial Intelligence & Data Science
Jaipur Engineering College & Research Centre
sakshijain.ai25@jecrc.ac.in

Abstract

This project presents the development of a comprehensive notification system for online food delivery services, leveraging the robust capabilities of the Django framework. Incorporating a PostgreSQL database, Docker containerization, and Kubernetes orchestration, the system facilitates a seamless operational flow from user registration to order fulfilment. It encompasses secure user authentication, a user-friendly order management interface, dynamic real-time notifications, and a sophisticated admin dashboard for effective platform management. Additionally, the integration of a custom-trained language model enhances user interaction through an intelligent chatbot feature..

Keywords: *Python, Django, SQL, Docker, Kubernetes, Templates, Redis, Celery*

Article Status

Available online :

1. Introduction

The advent of food delivery applications has revolutionized the dining experience, offering unparalleled convenience to consumers. Despite their widespread adoption, these applications often fall short in providing efficient order tracking, necessitating multiple checks by users, which can lead to missed updates. Our proposed solution is an innovative food notification application built using Django, which ensures prompt and reliable delivery of order updates. The technology stack includes Docker, Kubernetes, Redis, Celery, and Web Sockets, all working in concert to optimize notification dispatch.

2. Literature Review:

Our review of existing literature reveals a transformative synergy between technological advancements and the food service sector. Notably, digital platforms are reshaping consumer habits, as detailed in the works of Li and Wang[2], highlighting the shift towards online food ordering. The Django framework, recognized for its adaptability, emerges as a fitting choice for developing interactive food service platforms, as discussed in existing studies. The importance of user experience in fostering loyalty is also evident, with Kim and Lee's research endorsing real-time notifications.

The unforeseen COVID-19 pandemic has only accelerated the need for contactless services, with research by Perez-Rios and Viejo illuminating technology's pivotal role during such times. Smith and Johnson's exploration into the competitive domain of food delivery elucidates how technological ingenuities, like notification systems, can secure a market advantage. Our work, grounded in this scholarly dialogue, aspires to refine the digital experience for consumers, streamline ordering processes, and contribute to the ongoing discourse on technology's role in the food industry.

Web Application Frameworks: Django, a robust Python web framework, has gained prominence in web application development. The study byshowcases the versatility and scalability of Django, making it a suitable choice for building dynamic food service platforms.
User Experience and Convenience: User experience is paramount in food service applications. Research by Kim and Lee [6] emphasizes the importance of user-centric design, including real-time notifications, to enhance user satisfaction and loyalty

3. System Architecture:

The architectural blueprint of our food notification app is multi-tiered, ensuring an optimal balance between user accessibility and backend robustness. At the forefront is a user dashboard, allowing for intuitive interaction with the service. The Django web application, the backbone of our system, orchestrates user requests and data flow, as depicted in our provided illustrations. The PostgreSQL database acts as a repository for all transactional and configurational data. Order processing modules manage the full lifecycle of an order, supplemented by a notification engine and ancillary communication services like SMS/Email. The server environment, hosted on Gunicorn, and bolstered by Kubernetes for load balancing, guarantees system resilience against surges in demand

1.User Interface (UI):

-User Dashboard: A web-based interface where users can log in, browse menus, place orders, and manage their accounts. The visual representation of new users and retained users per month are shown in figure 1:

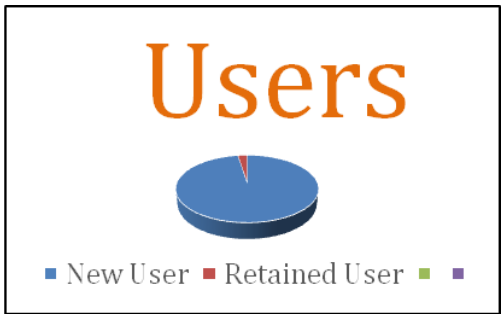

Figure 1: User Segmentation

2. Application Layer:

- Django Web Application: The core of the system, built using Django, handles HTTP requests, user authentication, and data management. Architecture of building web app is shown in figure 2

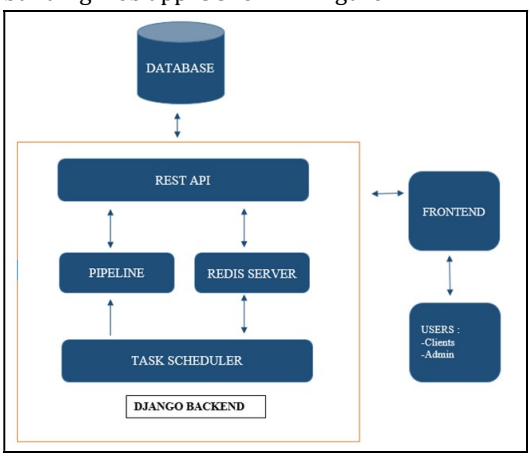

Figure 2: Basic Architecture of Web App

- Database: Stores user accounts, orders, menu items, and restaurant information. Worked on PostgreSQL. Figure 3 represents the ER diagram of database structure

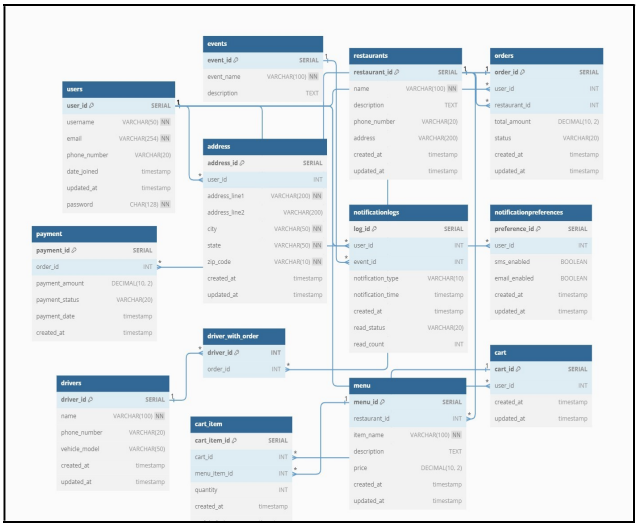

Figure 3E-R Diagram

3. Business Logic:

Order Processing: Manages the lifecycle of orders, from placement to delivery. This includes order confirmation, tracking, and notifications to users and restaurants.

User Authentication: Handles user registration, login, and account management.

Notification Engine: Generates and sends real-time notifications to users and restaurants regarding order status updates.

SMS/Email Service: Sends order confirmations, delivery updates, and promotional messages to users and restaurants.

4.Server Environment:

Web Server: Hosts the Django application. Choice here was Gunicorn

Authorization: Defines access controls for different user roles (e.g., customers, restaurant owners).

5. Scalability and Load Balancing:

Use load balancers and scaling mechanisms to handle increased traffic during peak hours.

This architecture provides a foundation for food notification Django web app, ensuring scalability, security, and efficiency in managing orders, notifications, and user interactions.

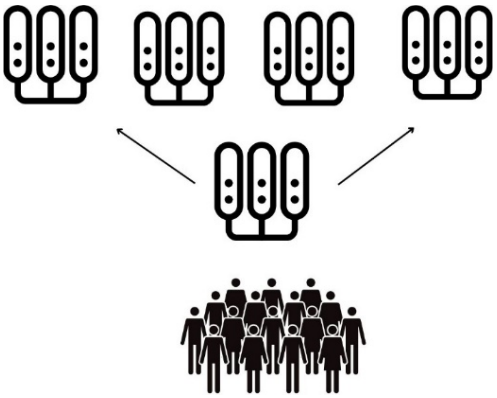

Figure 4: Load Balancing through Kubernetes

4. Implementation:

Our implementation utilizes Django's rich ecosystem, which simplifies web application development. Noteworthy Django libraries such as Django-rest-framework, Django-channels, and Django-celery enable RESTful API creation, real-time communications, and asynchronous task management, respectively. Complementing these are ancillary tools like Docker-compose for container management, Kubernetes for orchestration, and Redis for key-value storage, which together facilitate a streamlined task scheduling mechanism, as delineated in our flow diagrams.

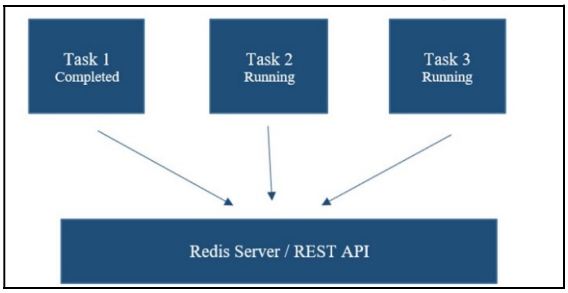

Figure 5: Flow Diagram of Task Scheduler

5. Project Snippets:

This is the landing page of the food notification app:

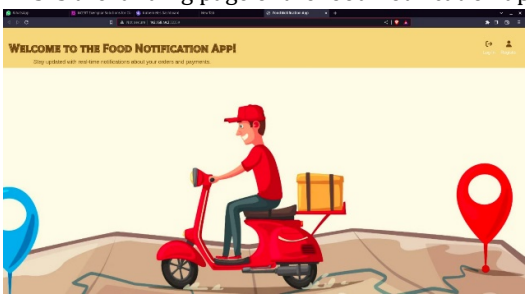

Figure 6:Landing Page

This is the restaurants page of the project where the names of all the restaurants are listed:

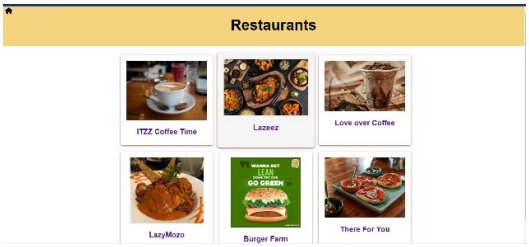

Figure 7Restaurants Page

6. Future Enhancements:

Looking ahead, the system's evolution could embrace AI-driven personalization, advanced security measures, multilingual support, interactive voice response, augmented reality features, environmental sustainability initiatives, predictive analytics, and blockchain for enhanced transparency. These forward-looking enhancements aim to enrich the platform's user-friendliness and global appeal.

The user interface of the food notification application is crafted to provide a seamless and intuitive experience. Upon arrival, the landing page greets users with a clear and engaging interface, as depicted in the accompanying visuals. Additionally, the restaurant directory page presents a comprehensive list of dining options, enabling users to effortlessly navigate through their choices.

7. Conclusion:

Our project culminates in a Django-based notification system that significantly enhances the end-user experience by providing immediate and precise updates throughout the food delivery process. This technological solution, empowered by Docker and Kubernetes, ushers in a new era of customer satisfaction and engagement, where each delivery is an exemplar of efficiency and convenience.

8. References:

The foundation of our project is built upon a diverse array of scholarly works and technical resources. From the theoretical underpinnings provided by key studies in the realm of software engineering to the practical guides on Python and Django, each reference has contributed to the robust development of our application. These sources, which range from academic journals to technical tutorials, have been meticulously documented to reflect their influence on our work.

[1] The Physics Classroom. (n.d.). Retrieved from https://www.physicsclassroom.com/.

[2] Kelly, L., & Breault, K. (2006). Developing Educational Websites: Investigating Internet Use by Students and Teachers. In Proceedings of Thinking, Evaluating, Rethinking, ICOM-CECA Conference, Rome.

[3] AglaSem Admission. (n.d.). Retrieved from https://admission.aglasem.com/.

[4] Sachan, N. (2019, February 20). Welcome to BHU Student Club, BHU Student Club. Retrieved from http://bhustudentclub.in/.

[5] Jalote, P. (2003). An Integrated Approach towards Software Engineering. Narosa Publishing House.

[6] Thakur, M.S. (2017). Review on Structural Software Testing Coverage Approaches. International Journal of Advance Research, Ideas and Innovations in Technology, 281-286.

[7]MKdos, Encode -OS LTD 2011-presenthttps://www.django-rest-framework.org/

[8]SpringGuide -Github : https://github.com/spring-guides/gs-messaging-stomp-websocket

[9] Akash Shrivastava(2022)-Published in ScaleRealMedium - https://medium.com/scalereal/push-notifications-through-django-db528c303b91

[10] Musciano, C., & Kennedy, B. (1996). HTML, The Definitive Guide. O'Reilly & Associates.

[11] Powell, T. A. (2010). HTML & CSS: The Complete Reference. The McGraw-Hill Companies.

[12] Flanagan, D. (2006). JavaScript: The Definitive Guide. O'Reilly Media, Inc.

[13] Shenoy, A., & Sossou, U. (2014). Learning Bootstrap. Packt Publishing Ltd.