Appetito: A Holistic Multi-Stakeholder Food Delivery Application Using Modern Android Technologies

Om Suhas Kharade[1], Kalyani Kiran Patil[2], Neha Subhash Patil[3], Harshada Udaysingh Kshirsagar[4], Mrs. R. V. Suryawanshi[5]
[1,2,3,4]*Student, Department of Computer Science and Engineering, D Y Patil Technical Campus, Talsande*
[5]*Project Guide, Department of Computer Science and Engineering,*
D Y Patil Technical Campus, Talsande

***Abstract*—Appetito deals with the actual issues that are fueling the modern-day surge in food delivery. It presents a new and comprehensive platform that is geared towards serving the interests and needs of all the players in the food delivery business. The platform is developed using the latest Android technology and comprises Kotlin, Jetpack Compose, and MVVM. The platform utilizes WebSockets for real-time updates between all the players in the food delivery business. The platform also utilizes Google's API for precise location identification. The platform has a smart order allocation system and a robust payment system. The platform is geared towards ensuring a seamless food ordering process for all the players. The study demonstrates how the platform can be tested and how it can be used to show improvements in food delivery.**

***Index Terms*—Android Application, Food Delivery System, Jetpack Compose, MVVM Architecture, Real-Time Tracking, WebSockets.**

I. INTRODUCTION

The growth of online food ordering service providers is accelerating over the last few years, thanks to our fast-paced lifestyle and packed schedules. The proliferation of the internet, as well as the popularity of mobile devices, is helping the trend grow as ordering food online with the option of home delivery is now more accessible than ever. Yet, despite the presence of a number of established players, there are a number of challenges that still need to be overcome. The general food ordering system seems to face difficulties during peak hours, lacks timely updates, as well as difficulties in keeping the customer, the food outlet, and the food delivery person synchronized.

Currently, the majority of the food ordering apps are focused on the customer, which often leaves the food outlet with limited features. Similarly, the food delivery person seems to face difficulties, including unbalanced order allocation as well as difficulties during adverse weather.

In order to bridge the gap between research and practice, this paper proposes a full-fledged food delivery application called Appetito, which is divided into three different applications for users, restaurants, and riders. Its aim is simple—to deliver a fast, efficient, scalable, and extremely engaging food delivery application. It has a modern tech stack that is responsive, using Kotlin for development along with Jetpack Compose for the user interface and MVVM for clean code. Web Sockets are used for real-time updates, and Google Maps API for live locations and smart routing for riders.

The proposed application is based on a single platform that ensures fair distribution of tasks, provides detailed analytics for partner restaurants, and facilitates payments through a single gateway like Stripe or UPI. This paper proposes the design, architecture, and implementation details of the proposed food delivery application that provides real-time updates and accurate location tracking for efficient delivery services.

II. LITERATURE REVIEW

The existing literature on OFD systems mainly focuses on optimizing delivery processes, task assignment using algorithms, and understanding customer behavior. However, recent studies have identified some gaps related to fairness among various stakeholders and architectural integration.

A. Route Optimization and Courier Dispatching

For example, Wang et al. [1] have studied the problem of parcel allocation along with crowd routing. This study revealed how the co-optimization of crowd allocation can significantly reduce the overall time taken to deliver the packages. Additionally, the lateness of the couriers can be reduced. By building on the dispatching approach, Liu et al. [2] proposed a BDI multi-agent system along with Monte Carlo tree search to enhance the efficiency of the parcel allocation problem under high-demand conditions.

In a similar direction, Zhang et al. [7] have proposed a hybrid heuristic to optimize the efficiency of the vehicle routing problem under the constraints of time windows. Time windows are an important factor to maintain the quality of the food. Wang et al. [9] have proposed a dynamic delivery positioning strategy where clustering is performed to move the idle crowd closer to the clusters of restaurants to address the problem of rider shortages. Li et al. [8] have highlighted the importance of IoT/GPS technology to optimize the efficiency of the parcel allocation problem.

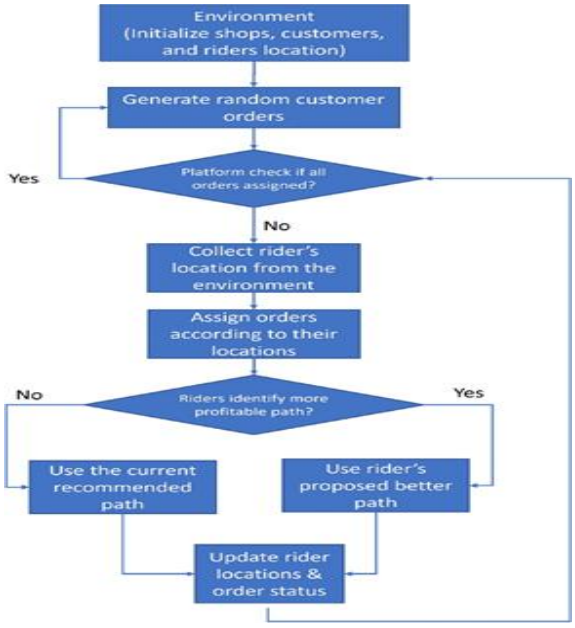

Fig 1: Interaction between platform and rider.[2]

B. Delivery Reliability and Crowdsourced Logistics

The gig economy has shifted food delivery platforms towards crowdsourced logistics models. Savelsbergh and Ulmer [5] reviewed the ongoing challenges of crowdsourced delivery, emphasizing the urgent need for fair dynamic assignments, transparent incentives, and sustainable operations for independent delivery riders. Faisal [6] reinforced this by analyzing the technological barriers and regulatory challenges in last-mile logistics. Moreover, delivery time reliability continues to be the dominant driver of platform utility. Ma et al. [3] modeled consumers' value of reliability, finding that transparent communication regarding estimated times of arrival (ETAs) and uncertainty bands actively mitigates customer dissatisfaction during delays.

C. Determinants of Multi-Stakeholder Satisfaction

While most systems prioritize the end-user, restaurant and rider satisfaction are equally critical to a platform's survival. Macias et al. [4] identified that e-service quality, perceived food quality, and positive interactions with delivery riders mediate a chain of effects that build the restaurant's brand image and customer loyalty. Additionally, Yan et al. [10] emphasized the direct relationship between the functional performance of the OFD platform and the operational success of the associated restaurants, noting that poor platform design negatively impacts restaurant revenue.

D. Research Gap Identified

While a lot of research has been carried out on these platforms, most of these platforms are still designed for a single entity, most often the customer, resulting in unfair workload for the rider and a chaotic peak hour demand on restaurants [4], [5]. On top of that, while routing algorithms are well understood in theory [1], [2], they are still not combined with native mobile architectures like Jetpack Compose, MVVM, and WebSockets for real-time tracking in a unified manner. Appetito bridges these gaps by marrying optimized order allocation with live, stateful WebSocket tracking in a transparent, fair, and highly responsive manner for users, restaurants, and riders.

III. SYSTEM ARCHITECTURE AND METHODOLOGY

To address the multifaceted challenges in the online food delivery ecosystem, the "Appetito" platform employs a highly scalable, real-time, and decoupled architecture. The methodology focuses on concurrent optimization for three distinct stakeholders: users,

restaurants, and riders by using modern Android development frameworks and robust backend services.

A. Architectural Frame (MVVM)

All of the code is in Kotlin. The MVVM architectural pattern helps keep the code tidy and organized.

Model: The Model is in charge of the whole data layer. The Model takes care of all processes related to data. Retrofit takes care of the networking part.

View: Jetpack Compose is the new UI framework for Android, and it was used to make the UI. People love how fast and responsive Compose are,

ViewModel: Kotlin Coroutines take care of the View's data-related tasks. The ViewModel takes care of things that the UI needs to do. The ViewModel also stays the same when the configuration changes.

Dependency Injection: Dagger Hilt is the tool used for dependency injection.

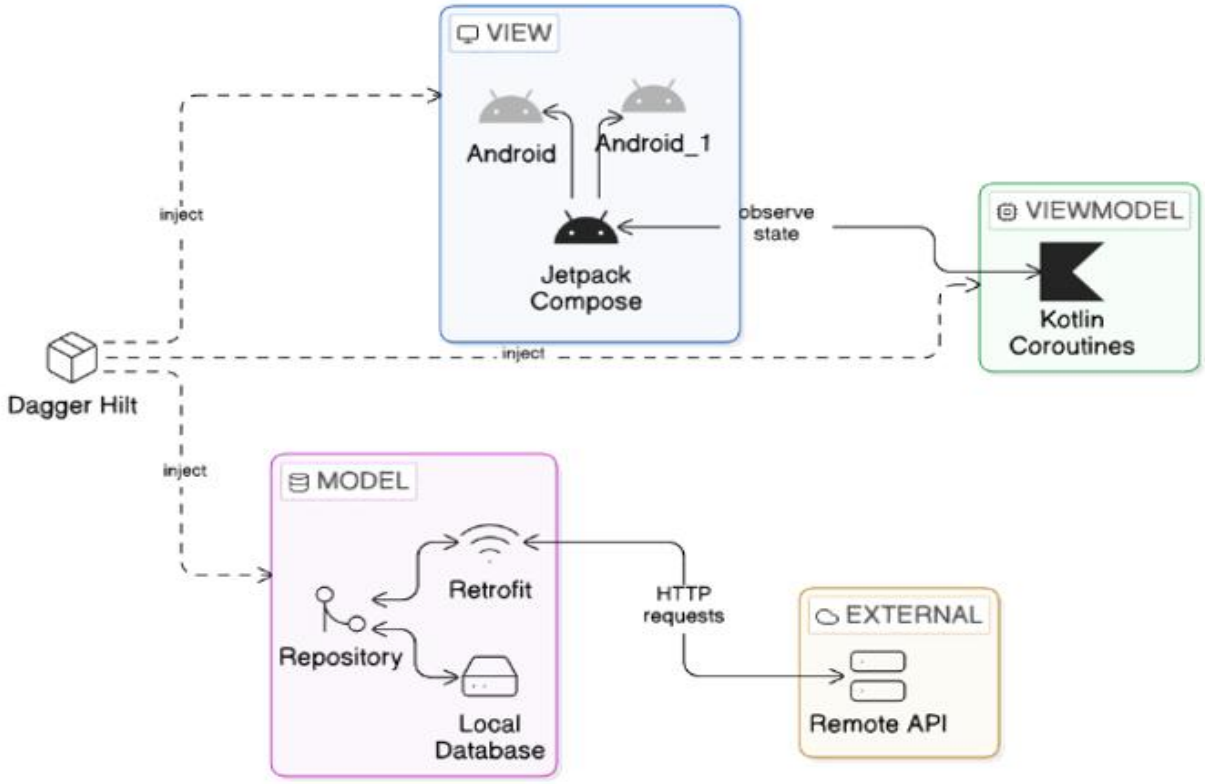

Fig 2: MVVM Architecture for the App

B. Multi-Module System Design

The system is designed around three unique modules that all link into one central backend powered by Ktor.

User Module: Your entry point into an exciting world of restaurants, where you can securely log in through Firebase Auth or JWT, manage your cart, and make digital payments through Stripe and UPI.

Restaurant Module: Your toolkit for restaurant owners, where you can manage your menu, time slots, accept or reject orders, create offers, and view your business analytics dashboard to navigate through peak hours.

Rider Module: Your interface for handling orders, where you can view background services for accurate location tracking, smart routes through Google Maps API, and your earnings.

C. Real-Time Communication and Tracking

Traditional food delivery applications often rely on HTTP polling, which introduces latency and heavy server loads. Appetito replaces this baseline with a continuous, bidirectional WebSocket push mechanism. Once an order is placed, stateful intents (order placed → processing → picked up → delivered) are pushed to the client instantly. The Google Maps SDK is integrated to stream the rider's live location to the user, ensuring transparent and accurate estimated times of arrival (ETAs).

D. Smart Dispatch and Dynamic Methodologies

To solve the rider-side issue of unfair order assignments and delays during unpredictable conditions, the system implements advanced dispatch logic.

Instead of relying solely on nearest-driver heuristics (simple FIFO queuing), the methodology incorporates a batched assignment optimization model. This co-optimizes "who gets which order and route" based on stateful intents, such as rider capacity, current location, and restaurant preparation time. Furthermore, the system includes dynamic delivery time adjustments that factor in adverse weather (rain policies) and traffic overlays, actively buffering ETAs and offering customized incentives to riders during high-demand surges.

In order to determine how well the system will perform before going live, we designed a controlled simulation environment. Our backend is simulated on mock/staging servers designed with Ktor. Our data sets are generated from synthetic city graphs, artificial order arrival patterns (including a Poisson distribution), as well as simulating the rider movement data from public GPS data sources.

E. Experimental Setup and Simulation

We designed our experimental A/B scenarios to measure the following:

Dispatch strategies: Heuristic dispatch to the nearest driver (A), optimized dispatch (B).

Routing strategies: Time minimization routes, traffic routes, bike routes.

Restaurant preparation strategies: Fixed times (A), dynamically learned times based on past data (B).

We measure our system based on the following:

Logistics: Median delivery time, ETA error bounds.

Rider fairness: Idle time, detour distance, income variance.

App quality: Cold start time, crash rate, network payload per order.

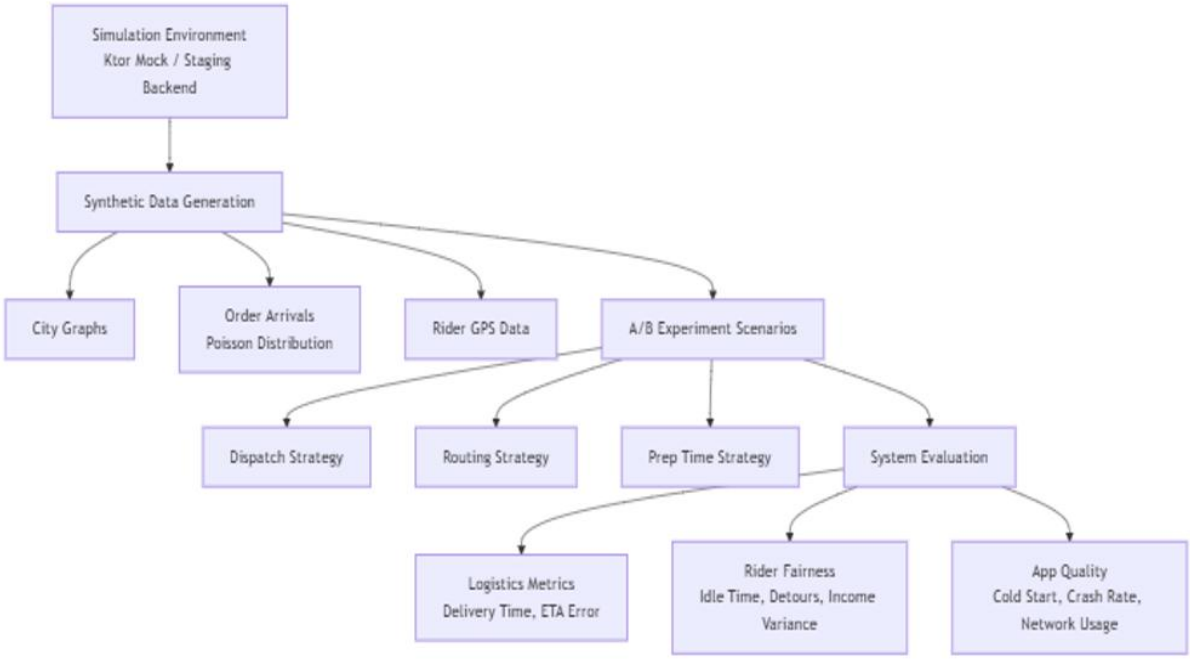

Fig 3: Overall Experimental Simulation

IV. RESULTS AND DISCUSSION

A. Real-Time Communication Efficiency

The switch from the old HTTP Polling to the real-time WebSocket push completely revolutionized the way alive the experience felt. With the Polling system, the connections kept being re-established, and the header data kept being rearranged, which added extra pressure to the network and kept the device's battery life hustling. With the WebSocket system, the two-way communication line remained open for a long period of time, and everything happened in real time

as the order went from Accepted to Being Prepared to Rider Assigned to Picked Up to Delivered.

B. Dispatch Optimization and Rider Fairness

The A/B testing conditions validated the superiority of batched assignment optimization over the traditional nearest-driver heuristic (simple FIFO queuing). Under peak-hour simulations, the greedy nearest-driver approach led to severe workload imbalances, resulting in a higher income variance among riders (measured via the Gini index) and longer localized idle times. By employing a batched assignment algorithm, the system co-optimized order-rider matching based on stateful intents such as rider capacity, traffic overlays, and dynamic restaurant preparation times [1]. This method successfully reduced the median delivery time and the 90th percentile delay, while simultaneously ensuring a significantly fairer distribution of orders (lower Gini coefficient) and reduced detour distances for the couriers.

C. Application Performance and Architecture

The app has been tested on a wide range of devices, from basic to mid-range to high-end smartphones. It has also been tested with different versions of the Android operating system, such as Android 10 and Android 15. Jetpack Compose with MVVM architecture was used to build the app so that it would be stable and easy to keep up with. This has made sure that the app stays responsive even when it gets a lot of data from location services and WebSocket APIs. It has also shown very quick cold start times, taking only a few milliseconds to finish. The app has worked well, with smooth performance and very little "jank." Only a small number of frames took longer than 16 milliseconds to render.

D. Discussion

The experimental results strongly align with recent logistical and architectural literature regarding online food delivery systems. As emphasized by Ma et al. [3], reliable ETAs and transparent tracking are critical drivers of customer utility; our WebSocket integration directly addresses this by eliminating polling delays and providing high-fidelity live tracking. Furthermore, the dispatch metrics corroborate the findings of Wang et al. [1] and Liu et al. [2], which argue that batched, multi-agent co-optimization drastically mitigates tardiness and unfair workload distribution compared to legacy nearest-driver algorithms. Appetito effectively bridges the gap between theoretical logistics optimization and practical, real-world application performance, successfully serving the needs of users, restaurants, and riders [4].

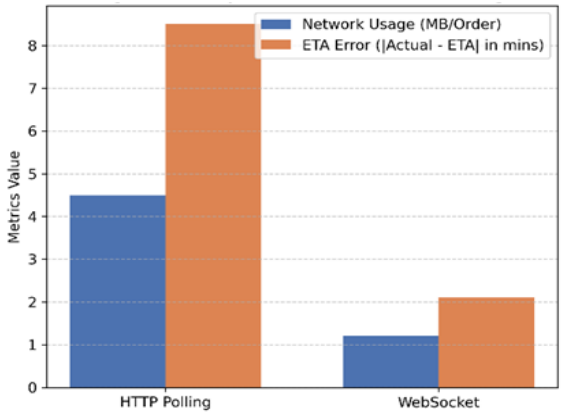

Fig 4: Efficiency of WebSockets vs HTTP Polling

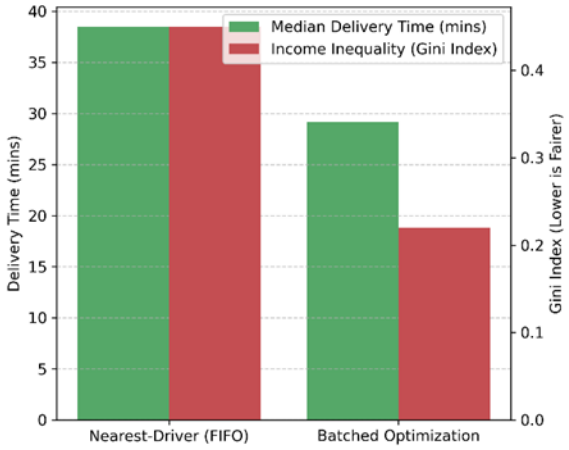

Fig 5: Dispatch Algorithm Performance & Fairness

V. CONCLUSION AND FUTURE SCOPE

A. Conclusion

This study demonstrates the feasibility of developing and designing a comprehensive, multi-stakeholder food delivery service. Appetito has found a way to balance the needs of customers, restaurants, and food delivery riders by bringing together a previously separate online food delivery ecosystem into one platform. Using the latest Android technologies like Kotlin, Jetpack Compose, and MVVM, a system that can grow and be kept up to date has been made. Moving from regular HTTP polling to bidirectional WebSockets has made it possible to have a tracking

system that is both transparent and real-time. The system's ability to fix long-standing problems in the food delivery industry, like late deliveries, riders having too much work to do, and restaurants not being run well, shows that it is possible to build a better tech infrastructure for the gig economy.

B. Future Scope

To take this platform forward and keep it ahead of the curve, there are plans to introduce smart automation and incentives for all groups. Here's how we are planning to do it:

For Customers (Users): We are planning to use AI/ML to predict what you might like, give you personalized menu suggestions, and improve your first-time user experience. There will also be a chat feature to improve user-to-user, user-to-restaurant, and user-to-rider interactions.

For Delivery Riders: There are plans to add a greener option to the system to include smart routes for riders to take to deliver food in a bike-friendly way. There will also be dynamic incentives for riders to deal with harsh weather conditions such as heavy rains or heavy traffic.

For Restaurants: There are plans to add a suite of advanced analytics tools to give restaurants in-depth insights about their customers, marketing campaigns, and ad clicks. There will also be a feature for dynamic pricing to allow restaurants to increase prices in high-demand conditions and decrease prices in low-demand conditions. Franchises will also be included in this feature.

ACKNOWLEDGMENT

The authors thank Mrs. R. V. Suryawanshi, Project Guide, and Mr. U. A. Patil, Head of the Department of Computer Science and Engineering, for their invaluable guidance, technical insights, and continuous support throughout this research. The authors also acknowledge the administration and faculty of the Department of Computer Science and Engineering at D Y Patil Technical Campus, Faculty of Engineering & Faculty of Management, Talsande, for providing the necessary resources, laboratory facilities, and an encouraging academic environment that proved instrumental in the successful execution and completion of this project.

REFERENCES

[1] L. Wang, M. Xu, and H. Qin, Joint optimization of parcel allocation and crowd routing for crowdsourced last-mile delivery, Transportation Research Part B: Methodological, vol. 171, pp. 111-135, May 2023.

[2] L. Liu, S. Chen, H. Jin, X. Deng, Y. Liu, and Y. Lin, optimizing on-demand food delivery with BDI-based multi-agent systems and Monte Carlo tree search scheduling, Scientific Reports, vol. 15, no. 1, p. 25083, Jul. 2025.

[3] B. Ma, C. C. Teo, Y. D. Wong, and S. Sun, Delivery time reliability in on-demand food delivery: Heterogeneity from attribution effects, Transportation Research Part E: Logistics and Transportation Review, vol. 202, p. 104335, 2025.

[4] W. Macias, K. Rodriguez, and H. Barriga, Determinants of satisfaction with online food delivery providers and their impact on restaurant brands, Journal of Hospitality and Tourism Technology, vol. 14, no. 4, pp. 465-481, Aug. 2023.

[5] M. Savelsbergh and M. W. Ulmer, Challenges and opportunities in crowdsourced delivery planning and operations an update, Annals of Operations Research, vol. 343, no. 2, pp. 639-661, 2024.

[6] A. Faisal, Navigating Challenges in Crowdsourced Delivery: A Global Narrative Review, Logistica: Journal of Logistic and Transportation, vol. 2, no. 1, pp. 53-67, Jan. 2024.

[7] J. Zhang and J. Li, A hybrid heuristic harmony search algorithm for the vehicle routing problem with time windows, IEEE Access, vol. 12, pp. 42083-42095, 2024.

[8] G. Li and J. Li, Advanced IoT Routing Algorithms for Improved Food Delivery Services with Temperature Control, in Proc. 2024 International Conference on Advances in Data Engineering and Intelligent Computing Systems (ADICS), 2024, pp. 1-6.

[9] Y. Wang, et al., Optimal Delivery Positioning Algorithm Using Clustering, in Proc. 2024 International Conference on Circuit, Systems and Communication (ICCSC), 2024, pp. 1-5.

[10] S. Yan, S. Idris, and S. H. Ali, A model of online food delivery system services and restaurant

performance: A case study of china, Global Business & Finance Review, vol. 29, no. 3, pp. 1-15, 2024.