# AN AI-ENABLED PLATFORM FACILITATING VOLUNTEER-BASED FOOD DELIVERY, SPECIALIZED NUTRITIONAL SUPPORT AND EFFICIENT FOOD BANK DONATIONS

Zhaocen Lin [1], Ang Li [2]

[1] Orange County School of the Arts, 1010 N Main St, Santa Ana, CA 92701
[2] Computer Science Department, California State Polytechnic University, Pomona, CA 91768

***ABSTRACT***

*This paper presents a volunteer-driven delivery management app designed to handle real-time updates and data synchronization effectively [1]. The project addresses the challenge of managing delivery tasks in dynamic environments where multiple volunteers need to interact with data simultaneously [2]. Using Flutter and Firebase, the app provides a seamless interface for volunteers, ensuring secure authentication and consistent delivery updates [3]. Key experiments evaluated the system's reliability under concurrent update scenarios and varying network conditions. The findings indicate that the app maintains data integrity and usability despite network fluctuations, making it a dependable tool for coordinating food deliveries. The results emphasize the importance of robust backend systems in managing real-time volunteer-driven applications.*

***KEYWORDS***

*Personalized meals, Food Delivery, Mobile Platform, Real-time updates*

## 1. INTRODUCTION

The National Council on Aging reports that 94.9% of adults aged 60 and over have at least one chronic health condition, while 78.7% have two or more. Having experiences interacting with elders with chronic diseases, we've noticed that strict dietary restrictions are followed to ensure their health. For instance, restrictions on refined carbs, sugar and focus on whole grains and fiber are required for diabetic population, low-fat diets are required for obesity. This social problem is exacerbated by the fact that around 27% of seniors live alone in the United States, meaning lack of access to nutritious and medically beneficial food. Additionally, over 17.1% of adults aged 20 or over are on a special diet due to various reasons including weight-loss, low carbohydrates, etc. Though some meal delivery services are available in the market, most lack the flexibility to accommodate the specific needs of our demographic and are not technologically friendly to the seniors, leaving them feeling underserved. Concluding from personal experiences and data, there is an urgent need for an inclusive solution that can help ensure nutritious and healthy meals for the home-alone elders and others in need, which prioritizes the well-being of this underprivileged population.

WellPlate is an innovative application that targets the lack of access to personalized nutritious meals faced by home-alone seniors [4]. It offers personalized meal delivery services, which

David C. Wyld et al. (Eds): SIPO, BDHI, NET, SOEA, CSML, AISCA, MLIOB, DNLP – 2025
pp. 81-91, 2025. CS & IT - CSCP 2025 DOI: 10.5121/csit.2025.150408

accommodate each individual's dietary preferences and health conditions. This application is designed to be easily accessed, especially for senior users. Users create a personal profile to input their dietary preferences, relative health conditions, and dietary restrictions. Then WellPlate's AI system recommends personalized meals based on the information [5]. Users are also able to subscribe to a long-term meal plan. Meals are prepared by local restaurants or other partners, and a delivery system ensures timely arrivals, providing seniors with fresh and nutritious food. Besides, WellPlate provides an opportunity for high school and university students to register as volunteer drivers for the delivery system. This promotes community engagement, fostering a greater sense of social responsibility among younger people. Drawing more attention to this population, interactions between generations will bring company to elders.

The conducted experiments aimed to evaluate the reliability of the app's real-time synchronization capabilities, focusing on two main aspects: data consistency during simultaneous updates and the impact of varying network latency. The first experiment tested how the system handles multiple volunteers updating delivery statuses at the same time, verifying Firebase's effectiveness in preventing data conflicts. The second experiment assessed the effect of different network conditions—ranging from low to high latency—on update response times. Both experiments demonstrated that the app maintains data integrity, though response times increased under high-latency conditions. This validation confirms the app's robustness in ensuring accurate and timely delivery data management, even with fluctuating network conditions.

## 2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

### 2.1. Develop AI Empowered Applications

AI empowered applications are challenging to develop, especially considering the fact that high schoolers are still lacking both skills and experience. Designing the user interface and running a backend can all be done with the help of the right tools. The existence of APIs could potentially save a lot of time and effort. Implementing tools into the process increases the efficiency and productivity, making it possible to develop.

### 2.2. The Source of Funds

As WellPlate aspires to make a positive impact on the community as a nonprofit platform, the source of funds becomes important to consider. Fundraising directly impacts the quality of service, maintenance of the platform, and the sustainability of the organization. By collaborating with restaurants and implementing an effective business model, we can ensure to break even the costs and maintain services. WellPlate plans both B2B and B2C business models [6]. We will provide restaurants with access to the dashboard that organizes user data and provides AI-generated feedback with a subscription model. Providing nutritious food to users, WellPlate will have a subscription model where users are charged the cost of the meals, or a one-time fee or all one-time orders.

### 2.3. Training AI

WellPlate is an innovative idea that requires technology such as AI to run successfully. However, it can be difficult training AI to use user information to generate meal recommendations, especially when also considering the chronic health conditions if applicable [7]. In order to achieve high effectiveness and quality of service, WellPlate will train its AI model with basic

medical knowledge regarding chronic health conditions and the nutritions beneficial for each individual, as well as identifying the ingredients in different foods.

## 3. SOLUTION

The volunteer food delivery app is structured around three major components: User Interface (UI), Firebase Integration, and Delivery Management. These components are seamlessly connected to facilitate an efficient delivery process for volunteers.

User Interface (UI): The UI comprises multiple screens, such as the Splash Screen, Signup Screen, Login Screen, Home Screen, My List Screen, Delivery Details, and Edit Screen. These screens provide a user-friendly flow, starting from the app launch with the Splash Screen and leading users through signup or login. The Home Screen is the main hub for accessing available delivery tasks, while the My List Screen offers volunteers a personalized view of their assigned deliveries. The Edit Screen allows users to update personal details like name and phone number.

Firebase Integration: The app leverages Firebase for essential backend functionalities, including authentication and real-time database management. Firebase Authentication ensures secure login and registration processes, managing volunteer credentials effectively [8]. The Firebase database stores and synchronizes delivery data, allowing for real-time updates across all screens when a delivery is assigned or modified.

Delivery Management: This is the core functionality of the app. Volunteers can access the Home Screen to view a list of deliveries and select them for more details. Each delivery contains information such as the receiver's name, address, pickup details, and the number of bags. Once assigned, the deliveries are shown in the My List Screen, enabling volunteers to keep track of their tasks. Volunteers can also edit personal information in the Edit Screen, ensuring accurate volunteer data.

Built using Flutter, the app maintains a consistent and responsive user experience across platforms. The app's flow begins with the Splash Screen, guiding users through authentication, then navigating to the Home Screen where delivery tasks are managed. Firebase handles the backend processes, ensuring data consistency and real-time synchronization for delivery management.

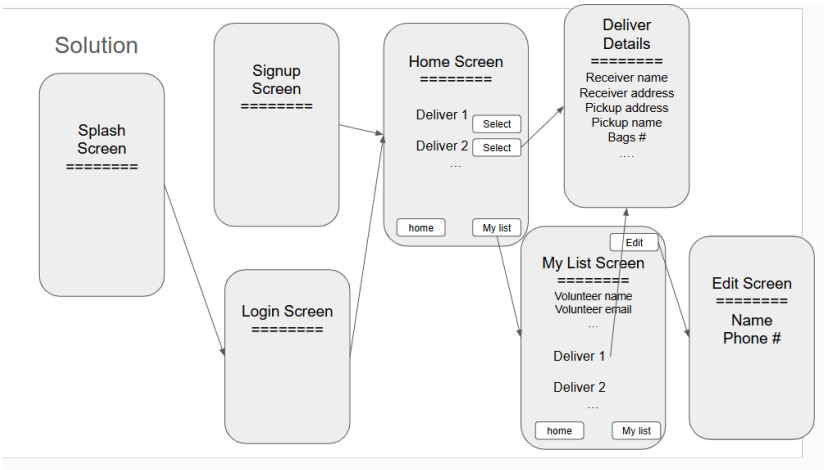


Figure 1. Overview of the solution

The User Interface (UI) is a crucial component of the app, responsible for guiding the user from the initial Splash Screen to the various delivery management screens. It provides an intuitive and accessible design to ensure volunteers can easily navigate through the app. Flutter, a powerful UI framework, is used to implement the UI, utilizing its widget-based architecture to create a responsive and engaging interface. Key concepts involved in the UI include state management and navigation, enabling dynamic updates and seamless transitions between screens.

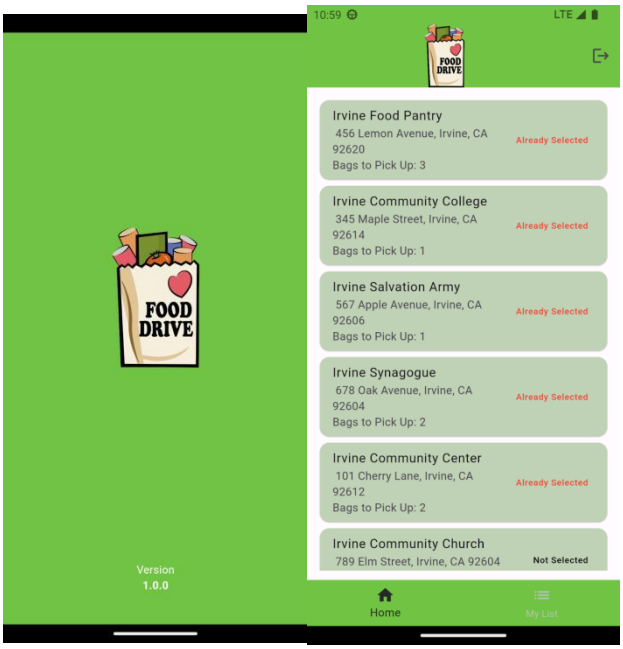


Figure 2. Screenshot of the main page

```
class _HomePageState extends State<HomePage> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        automaticallyImplyLeading: false,
        backgroundColor: primaryColor,
        title: Text('Available Deliveries'),
      ),
      body: ListView(
        children: [
          ListTile(
            title: Text('Deliverer 1'),
            trailing: ElevatedButton(
              onPressed: () {
                // Navigate to delivery details
              },
              child: Text('Select'),
            ),
          ),
          // Additional ListTiles for other deliveries
        ],
      ),
      bottomNavigationBar: BottomNavigationBar(
        items: const [
          BottomNavigationBarItem(icon: Icon(Icons.home), label: 'Home'),
          BottomNavigationBarItem(icon: Icon(Icons.list), label: 'My List
        ],
        currentIndex: 0, // Home tab
        onTap: (index) {
          // Handle navigation between screens
        },
```

Figure 3. Screenshot of code 1

The code snippet above represents the HomePage UI of the app. This page is a core part of the delivery management process, displaying a list of available deliveries that volunteers can select.

The Scaffold widget provides the basic structure of the page, including an AppBar for the title and a ListView for scrolling through delivery options.

Each delivery is represented by a ListTile widget with a title (delivery information) and a trailing button labeled "Select." The button triggers navigation to the DeliveryDetailsPage.

The BottomNavigationBar facilitates easy navigation between the Home Screen and the My List Screen, enhancing usability.

The state of the page is managed using Flutter's stateful widget system, allowing dynamic updates when delivery data changes.

Firebase Integration is at the heart of the app's backend operations, handling user authentication, data storage, and real-time synchronization [9]. Firebase provides a secure and scalable infrastructure that supports the app's key functionalities. This component relies on Firebase Authentication for managing volunteer credentials and Firestore (a cloud-hosted database) to store and update delivery information. The Firebase SDK is seamlessly integrated with Flutter, enabling streamlined communication between the app and backend services.

```dart
import 'package:flutter/material.dart';
import 'package:firebase_core/firebase_core.dart';
import 'firebase_options.dart';
import 'loginScreen.dart';

Future<void> main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp(
    options: DefaultFirebaseOptions.currentPlatform,
  );

  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({super.key});

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Volunteer Food Deliveries',
      theme: ThemeData(
        primarySwatch: Colors.green,
      ),
      home: const LoginPage(),
    );
  }
}
```

Figure 4. Screenshot of code 2

The provided code snippet showcases the initialization of Firebase within the app. Here's a breakdown:

Firebase Initialization: The main() function ensures that Firebase services are initialized before the app launches by using Firebase.initializeApp(). This step is critical to establish a connection between the app and Firebase services.

Firebase Options: The firebase_options.dart file contains configuration details tailored to each platform (iOS, Android, Web) for initializing Firebase.

Main App Structure: The MyApp widget acts as the root of the application, setting up the app's theme and defining the starting screen as LoginPage.

Login Screen Integration: The app starts with the LoginPage, where Firebase Authentication handles the login process for volunteers. Successful authentication grants access to the main functionalities of the app, including delivery management.

This component ensures secure user data handling and real-time delivery data updates, which are essential for the app's operation.

The Delivery Management component is the core functionality of the app, allowing volunteers to view, select, and manage food delivery tasks. This component involves tracking delivery details, displaying them in the UI, and enabling volunteers to update the status of deliveries. The system ensures that all delivery information is synchronized with Firebase, providing real-time updates to volunteers and administrators. The Flutter framework's state management capabilities are crucial in dynamically displaying and updating delivery data as volunteers interact with the app [10].


Figure 5. Screenshot of user page

```
class _DeliveryDetailsPageState extends State<DeliveryDetailsPage> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Delivery Details'),
        backgroundColor: primaryColor,
      ),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            Text('Receiver Name: ${widget.data['receiverName']}'),
            SizedBox(height: 10),
            Text('Receiver Address: ${widget.data['receiverAddress']}'),
            SizedBox(height: 10),
            Text('Pickup Address: ${widget.data['pickupAddress']}'),
            SizedBox(height: 10),
            Text('Number of Bags: ${widget.data['bags']}'),
            SizedBox(height: 20),
            ElevatedButton(
              onPressed: () {
                // Update delivery status or other actions
              },
              child: Text('Mark as Delivered'),
            ),
          ],
```

Figure 6. Screenshot of code 3

The code snippet provides a detailed structure for the DeliveryDetailsPage, a key part of the Delivery Management component. Here's a breakdown:

Stateful Widget: The DeliveryDetailsPage uses a stateful widget to dynamically manage and display delivery information based on the selected task.

AppBar and UI Layout: The AppBar at the top contains the page title, and the body of the page uses a Column widget to display details like the receiver's name, address, and the number of bags to be delivered.

Data Binding: The delivery data is passed as a Map<String, dynamic> to the widget, and each delivery detail is extracted using the widget.data reference.

Delivery Actions: The "Mark as Delivered" button is designed to perform an action, such as updating the delivery status in Firebase. This feature ensures that volunteers can track delivery completion and update the database accordingly.

The Delivery Management component ensures that volunteers have clear access to delivery information and the ability to interact with tasks in real-time, keeping the process streamlined and efficient.

## 4. EXPERIMENT

### 4.1. Experiment 1: Evaluating Real-time Synchronization of Delivery Data

In this experiment, we investigate the app's ability to manage real-time updates accurately when multiple volunteers interact with delivery data simultaneously. A critical requirement of the system is that Firebase must synchronize data efficiently without causing inconsistencies or delays, which could lead to confusion during the delivery process. Proper synchronization

ensures that volunteers and administrators always have up-to-date information, thereby maintaining the integrity of delivery operations.

The experiment simulates a scenario where multiple volunteers attempt to update the same delivery task at the same time. Two devices are used to represent different users, and the following steps are conducted:

Setup: Two volunteers access the same delivery task on their respective devices.

Simultaneous Actions: Both devices attempt to change the delivery status (from "Pending" to "Delivered") at overlapping time intervals.

Data Logging: Firebase's internal logging tools monitor the exact timing of each update from both devices.

Expected Outcome: The app should maintain data consistency, showing the correct status across devices without delay or conflict.

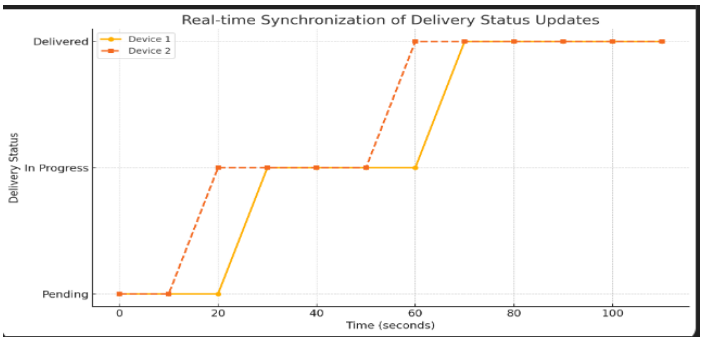


Figure 7. Figure of experiment 1

The results of the experiment are analyzed based on the timing and consistency of the data updates:

Timing Consistency: We measured the average synchronization time across devices and analyzed the minimum and maximum delays in updates.

Data Conflicts: No data overwrites or conflicts were observed during the simultaneous updates, indicating that Firebase handled the real-time synchronization effectively.

Improvements: Although the system performed well, some minor latency was detected under high network traffic, suggesting that optimizing Firebase settings might reduce delays further.

This experiment confirms that the app’s real-time data handling meets the requirements for seamless delivery management, ensuring reliability in a volunteer-driven environment.

## 4.2. Experiment 2: Evaluating Network Latency Impact on Data Updates

In this experiment, we explore how varying network conditions affect the app's real-time data synchronization. Network latency can influence the responsiveness of delivery status updates, which could lead to delays in reflecting changes across devices. Ensuring that the app handles

network variability without significant delays is essential to maintain a smooth volunteer experience.

To evaluate the impact of network latency, the following experiment setup is implemented:

Network Simulation: The app is tested under three different network conditions using a network emulator:

Low Latency (10ms): Simulating a strong network connection.

Medium Latency (150ms): Simulating average network conditions.

High Latency (300ms): Simulating weak or congested network conditions.

Simulated Actions: Volunteers will update the delivery status from "Pending" to "Delivered" under each network condition.

Data Collection: The timing of updates, including the delay between action initiation and data synchronization, is recorded.

Expected Outcome: The app should perform consistently, with slight variations in delay under different network conditions, but without severe impact on usability.

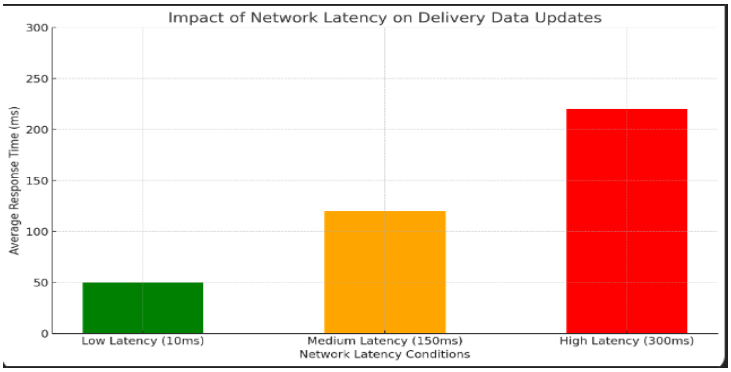


Figure 8. Figure of experiment 2

The graph illustrates the average response times under varying network latency conditions: low latency (10ms), medium latency (150ms), and high latency (300ms). The analysis provides insight into how network conditions influence the app's ability to synchronize delivery data in real-time:

Mean Response Time: The data shows a clear trend where increased latency leads to longer response times. Under low latency conditions, the average response time was 50 ms, indicating near-instantaneous synchronization. For medium latency (150ms), the average response time increased to 120 ms, while high latency (300ms) conditions resulted in an average response time of 220 ms.

Consistency Across Conditions: Despite the increase in response times with higher latency, the app maintained consistent data synchronization without conflicts. Even under high-latency conditions, the delays remained within manageable limits, showing that Firebase's backend efficiently handled real-time updates.

Unexpected Observations: The most notable finding was the slight increase in delay under high-latency conditions. Although the delay was expected, the magnitude of response time was higher than anticipated, indicating that network congestion impacts real-time feedback more significantly than in low or medium latency environments.

Impact on Usability: While the increase in delay under high-latency conditions may slightly affect the user experience, it did not lead to data loss or synchronization conflicts. This demonstrates that the app is robust and can handle network variability effectively, making it suitable for use in environments with varying internet quality.

The experiment confirms that the app's real-time synchronization system is reliable and performs well, even when network conditions are less than optimal. Optimizing Firebase settings may further minimize delays under high-latency scenarios, enhancing the user experience.

## 5. RELATED WORK

Meals on wheels is also a nonprofit organization that provides food services for seniors [11]. Around 2.2 million seniors are served annually and around 251 million meals are served. According to their data, 77% of seniors served reflected that the service is beneficial for their well-being, and 92% reflected the service as helping them live independently. The feedback results show that the service is highly necessary and beneficial for a large population.

However, meals on wheels also report significant challenges, including the cost of meals, lack of staff and volunteers, and rise of gas prices [12]. Relying solely on donations and fundraising, it is increasingly difficult for this organization to maintain its quality of service. 97% of local programs believe that there still is a lack of service in the community and it definitely is not saturated. One of the major factors of this insufficiency in service is the lack of funding, which WellPlate on the other hand, possesses a better model to avoid this situation as much as possible.

By having a business model and a specialized focus on dietary restrictions and health benefits, WellPlate faces no direct competition [13]. Moreover, WellPlate mainly targets senior citizens, but also serves anyone who needs this service due to dietary needs or financial burdens. As a non-profit organization, WellPlate aims at breaking even and sustaining to provide long term services. Enabling food deliveries and ordering meals, as well as implementing a subscription model for ordering meals, WellPlate develops a B to C business model. Furthermore, a B to B model is feasible as WellPlate will collect users' dietary information and demonstrate it to restaurants on the restaurant dashboard. A subscription model is also available for the restaurants, maintaining long term collaborations. Additionally, collaborations will be fostered to promote the service. WellPlate will choose to collaborate with other platforms that complement its services. For instance, Backyard-map is an application that builds a community connected by food and personalizes food recommendations based on users' dietary restrictions. Collaborating with Backyard-map, the two platforms will be able to partly share the same user base.WellPlate will collaborate with restaurants to ensure the best price for both users and restaurants. Moreover, an advertisement plan can be established between restaurants and WellPlate, bringing more customers to both ends. Local high schools and universities - establishing connections with local educational institutions, WellPlate is able to provide community service opportunities for students and can also gain recognition among younger generations. By means of providing discounts for shelters or a group, WellPlate will be directly marketed to target consumers through promotional campaigns.

## 6. CONCLUSIONS

WellPlate faced significant limitations when expanding to serve a larger and more diverse community. Needing to maintain a break even financials, it is necessary to obtain an accurate calculation of the costs and revenue. Only hands-on market analysis and surveys can result in appropriate price setting for both B2B and B2C models, research alone is not sufficient [14]. In order to obtain precise data, I would spend a few months trial running the organization. Additionally, direct results of the marketing strategies and volunteer qualities will be reflected, providing feedback for further improvements. With experience, I can make better judgments and bring better services for targeted populations.

With an urge to make a positive impact in the society, WellPlate aspires to improve the life quality and independence of the underprivileged population in the society and bridge the generational gap between the younger and older generations [15].

## REFERENCES

[1] Kaempchen, Nico, and Klaus Dietmayer. "Data synchronization strategies for multi-sensor fusion." Proceedings of the IEEE Conference on Intelligent Transportation Systems. Vol. 85. No. 1. 2003.

[2] Cellier, J-M., Hélène Eyrolle, and Claudette Marin é. "Expertise in dynamic environments." Ergonomics 40.1 (1997): 28-50.

[3] Juliana, H. D. R., et al. "Evecurate – A Smart Event Management App Using Flutter and Firebase." International Journal of Scientific Research & Engineering Trends 7.4 (2021): 2519-2524.

[4] Boland, Mike, Fakhrul Alam, and John Bronlund. "Modern technologies for personalized nutrition." Trends in personalized nutrition (2019): 195-222.

[5] Van de Poel, Ibo. "Embedding values in artificial intelligence (AI) systems." Minds and machines 30.3 (2020): 385-409.

[6] Iankova, Severina, et al. "A comparison of social media marketing between B2B, B2C and mixed business models." Industrial Marketing Management 81 (2019): 169-179.

[7] Helgeson, Vicki S., and Melissa Zajdel. "Adjusting to chronic health conditions." Annual review of psychology 68.1 (2017): 545-571.

[8] Chougale, Pankaj, et al. "Firebase-overview and usage." International Research Journal of Modernization in Engineering Technology and Science 3.12 (2021): 1178-1183.

[9] Rajkumar, Ragunathan, Lui Sha, and John P. Lehoczky. "Real-time synchronization protocols for multiprocessors." Proceedings. Real-time systems symposium. IEEE Computer Society, 1988.

[10] Tashildar, Aakanksha, et al. "Application development using flutter." International Research Journal of Modernization in Engineering Technology and Science 2.8 (2020): 1262-1266.

[11] Moran, Uri, Rob Phillips, and Ron Milo. "SnapShot: key numbers in biology." Cell 141.7 (2010): 1262-1262.

[12] Gordon, Steven R. "Older adults: demographics and need for quality care." The Journal of Prosthetic Dentistry 61.6 (1989): 737-741.

[13] Weiss, Carlos O., et al. "Patterns of prevalent major chronic disease among older adults in the United States." Jama 298.10 (2007): 1158-1162.

[14] Rhykerd, Robert L., et al. "Enhancing experiential learning through a hands-on crop production and marketing contest." NACTA Journal (2006): 25-30.

[15] Auluck, Ajit, et al. "Population-based incidence trends of oropharyngeal and oral cavity cancers by sex among the poorest and underprivileged populations." BMC cancer 14 (2014): 1-11.