

Groceries Shops Delivery System

Mohamed Khalid and Alif Ridzuan Khairuddin
 Faculty of Computing
 Universiti Teknologi Malaysia
 81310, Johor Bahru, Johor, Malaysia
 magamer@graduate.utm.my, alifridzuan@utm.my

Abstract— Generally, the pandemic makes it harder for individuals to obtain goods. Customers are either unable to enter the store to purchase the products they desire and must instead place orders from outside, or they are unable to touch the merchandise while browsing. Additionally, if the retail area is too small or narrow to ensure social isolation, there may be only one customer present at any given time. During payment, it is also difficult to build a social space between the consumer and the cashier. If grocery products were purchased online, many of these negative associations would be erased. To fix the issues, the Groceries Shops Delivery System was implemented. The project proposes to establish an online platform that enables clients to order products from local grocery stores and have them delivered or picked up online. The collected requirements will be modelled using appropriate UML diagrams. All modelling of requirements will be described in the appendix documents. Documents consist of the SRS, SDD, and STD. The SRS will represent the requirements using the appropriate UML diagrams. The database structure will be designed in the SDD. In addition, the user interface will be built within the SDD. The majority of PSM-2-developed pages will be included in the interface design part. The test cases will be designed for the STD. These test cases will be used to evaluate all the upcoming functionality. During the creation of this system, all steps of testing will be conducted. This will be done to guarantee that the performance of this system meets the expectations. This system should be completely operational and deployed at the conclusion of this project in order to resolve the current issue.

Keywords—online grocery shopping; mobile app usability; system implementation; e-commerce platform; user-friendly interfaces; development environment; testing and verification.

I. INTRODUCTION

E-commerce has been widely accepted in today's digitalized world, and many firms have taken advantage of it to expand their customer base. However, the most critical part of e-commerce delivery is often overlooked by enterprises. E-commerce is becoming increasingly popular as the number of Internet users worldwide continues to increase. Internet expansion, particularly in e-commerce, has clearly required a strong and speedy technological backdrop. Because of the rise of virtual shops, customers may now buy products and services online, which is now the quickest and cheapest choice. E-

commerce is widely considered to be the most profitable form of trading because of its low costs and ease.

Customers can buy goods from a retailer directly via the Internet using a web browser or mobile app, such as grocery online delivery. Shopping search engines let customers compare prices and availability across many e-retailers to find the best deal on a specific product. Due in part to the covid-19 pandemic, internet grocery sales increased by 300 percent in the first few months of the outbreak. 41 percent of online grocery shoppers were first-time purchasers, according to data from the National Retail Federation. The spread of covid-19 has pushed people to shop online for groceries.

II. RELATED WORKS

This section of the chapter discusses the current system analysis for the grocery delivery system utilizing existing web or mobile applications. Peapod, Walmart, and Dumpling are among the present systems on the market. Peapod is a well-known online grocery delivery service, Walmart offers online grocery delivery, and Dumpling is a personalized online grocery purchasing service. These systems enable customers to order products online and have them delivered to their doorstep with ease and adaptability.

A. Peapod

Peapod contrast to a delivery business that works in conjunction with other shops, Peapod operates its own food store. It is accessible in a limited number of cities and includes an iOS and Android app. Due to the app's real-time inventory updates, it is less probable that some groceries would be out of stock. Peapod typically offers next-day delivery; however, same-day delivery is available in certain places.

B. Walmart+

Walmart+ offers groceries delivery from its stores directly. Users can join up for Delivery Unlimited for \$13 per month or \$98 per year and receive unlimited deliveries of groceries at no additional cost. Walmart+ needs a \$35 minimum purchase for free shipping.

The amount does not include a driver gratuity. Users can plan their deliveries at a time and date of their choosing, or they can pay \$10 for expedited delivery and have their groceries

delivered within two hours or less. Express delivery do not require a minimum payment for orders. Users typically report fewer out-of-stock items since Walmart stores manage their own inventory.

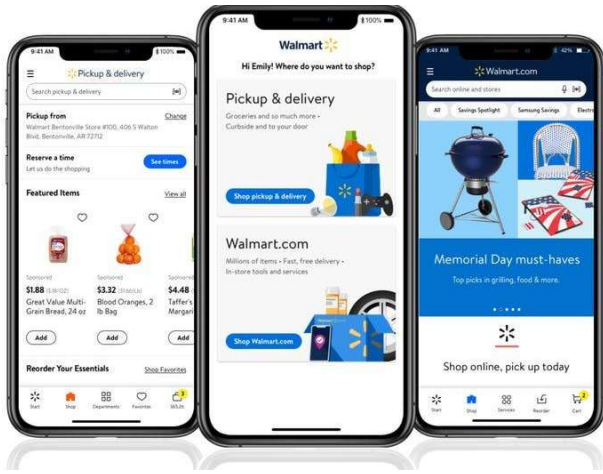


Figure 1. Walmart+ app interface.

C. Dumpling

Dumpling operates differently than conventional grocery delivery services. Instead of working with a restricted number of grocers, it employs personal shoppers. Users can select a shopper who shops at a particular store. Dumpling provides its services via its website as well as Android and Apple applications.

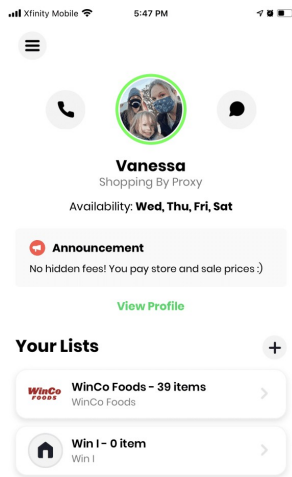


Figure 2. Dumpling app interface.

Customers can create a relationship with a shopper who is familiar with their preferences, or they can choose the shopper with the highest rating each time they place an order. Reviewers highlight that Dumpling enables them to construct a personal shopping business through fostering relationships with consumers. Therefore, users who are worried about fair pay for delivery drivers may find the app appealing. Comparison between Existing Systems

The functional comparison between the existing system and the new system is shown in Table I below.

TABLE I. COMPARISON BETWEEN SYSTEMS

Features	Peapod	Dumpling	Walmart+	Proposed System
The app has their own inventory storage	Y	N	Y	N
The app has its own store	Y	N	N	N
The app offers premium delivery annually or monthly	Y	N	Y	Y
The app is available in both ios and android	Y	Y	Y	Y
The app provides a real-time inventory update	Y	Y	N	Y
The app allows the customers to create a relationship with the shopper	N	Y	N	N

III. METHODOLOGY

This chapter will outline the methodology employed to construct the system. The chapter will also address the technologies that been be employed in the system's development and analyze the system's needs. Waterfall model stresses the assurance that, each phase must be completed before the following phase of development. Meaning there will be no scope for tasks to be left unsettled for later part. Hence, ensures the project work flow is managed fluently and organized constructively. However, errors can be fixed only during the phase since it might be affecting the subsequent phases.

The Agile-Waterfall Hybrid Model was selected for the project due to its capacity to reconcile the rigidity of Waterfall with the adaptability of Agile. The existence of predetermined deadlines and requirements was compatible with the Waterfall methodology. The hybrid model also accommodated the need for project integrations, allowing for greater flexibility in addressing dependencies. It provided a framework for consistent communication and feedback while maintaining execution teams' independence. The decision was effective in producing successful project outcomes.

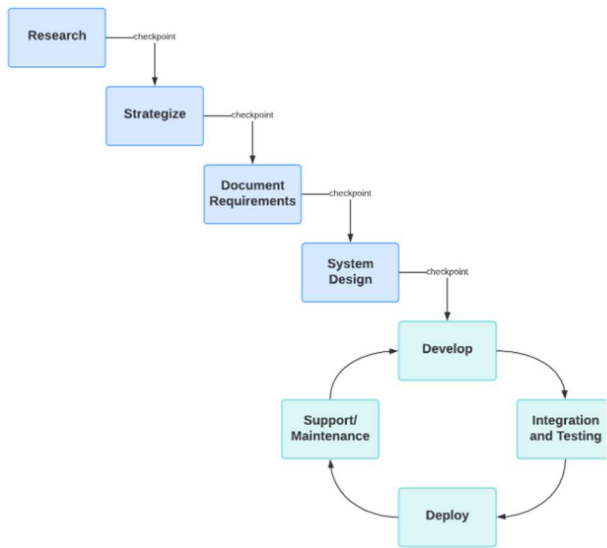


Figure 3. Agile-Waterfall Hybrid Model.

E. Requirements Gathering Phase

During the Requirements Gathering phase, I interacted with stakeholders and conducted extensive research on the market's extant grocery delivery applications to collect valuable information and insights. I interacted with stakeholders to determine their specific requirements, preferences, and anticipations for a grocery delivery system. In addition, I analyzed the features, functionalities, and user experiences of popular applications such as Peapod, Walmart, and Dumpling. This process yielded valuable insights that helped define the proposed grocery delivery system's requirements and specifications, ensuring that it meets the expectations of all stakeholders and end users.

The GSD system's functions were determined after a thorough review of the system's requirements. Prior to beginning system development, a well-documented Software Requirement Specification (SRS) should be produced. System needs can be divided into two categories: those that have a functional purpose and those that do not.

Functional Requirements are requirements that specify the system's behavior. The system's functional requirements is presented in Figure 4.

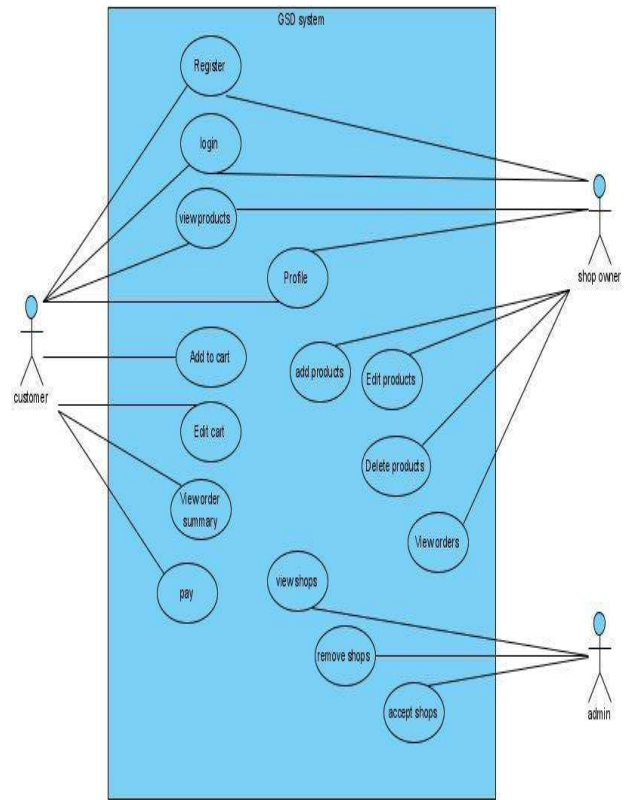


Figure 4. Use Case Diagram.

F. System Architecture Design

The MVC architecture has been selected for this system. The MVC architecture splits the system into three layers: Model, View, and Controller. The Model layer is responsible for database communication. This layer transmits and retrieves database data. The View layer is the second layer. This layer comprises the entire user interface with which the user will interact. The Controller layer is the third layer. This layer bridges the View and Model levels and houses the entire system's logic. The MVC architectural pattern facilitates the development of numerous interfaces by requiring only the replacement of the view layer. This architectural pattern produces extremely scalable and customizable applications.

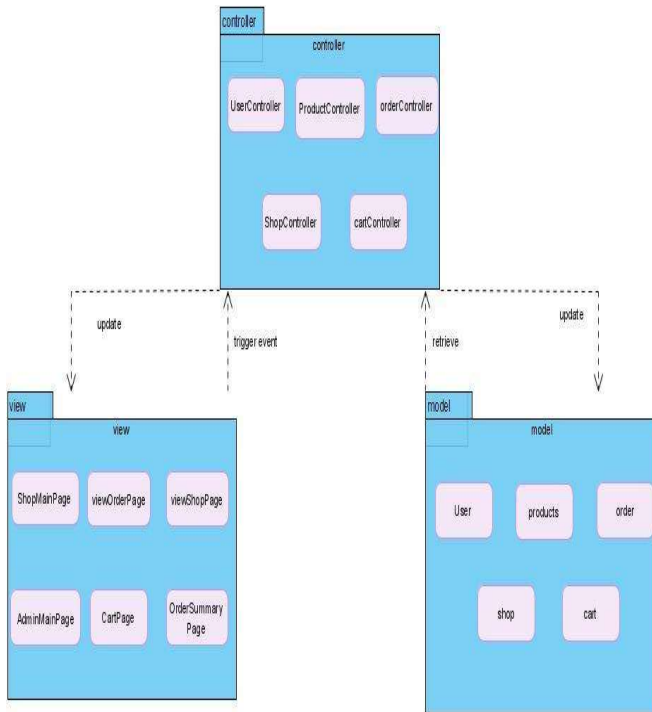


Figure 5. System Architecture Design.

IV. IMPLEMENTATION

In the development environment, Visual Studio Code (VS Code) was chosen as the source code editor due to its extensive features, multi-language support, and vibrant extension ecosystem. It provided developers with a lightweight and efficient coding environment, enabling syntax highlighting, code completion, debugging, version control integration, and customization options. Xampp, a cross-platform package, facilitated the local server environment by combining Apache, MariaDB/MySQL, PHP, and Perl. It simplified web server configuration, database management, and PHP script execution, allowing seamless development and testing of the GSD system. Postman, a widely used collaboration platform, played a crucial role in testing and validating the system's backend API. Its user-friendly interface and comprehensive functionality allowed developers to simulate HTTP requests, examine responses, and ensure the reliability and effectiveness of the API endpoints.

During the system implementation phase, the GSD system adopted the Model-View-Controller (MVC) architectural pattern with a service layer. Node.js and MySQL were utilized for the Model layer, handling data management and business logic. The View layer was developed using HTML, CSS, and JavaScript to create user interfaces, while the Controller layer, also built with Node.js, facilitated communication between the Model and View, handling user requests and system modifications. Additionally, a service layer encapsulated business logic and ensured modularity. Code fragments and examples were provided to demonstrate the implementation details, showcasing the system's ability to manage user interactions, process data, and deliver the intended functionality.

```
login: async (email) => {
  const connection = await mysql.createConnection({
    host: "localhost",
    user: "root",
    password: "",
    database: "grocery"
  });

  const [results, fields] = await connection.execute('SELECT * FROM `user` WHERE `email` = ?',
  // console.log(results)
  if (!results) {
    return null;
  }
  return results;
},
```

Figure 6. Code Snippet of the Model code using (Visual Studio Code).

```
@override
void initState() {
  super.initState();
  fetchShopData();
}

Future<void> fetchShopData() async {
  try {
    var url = Uri.http('192.168.100.43:3000', 'users/shops');
    final response = await http.get(url);
    if (response.statusCode == 200) {
      setState(() {
        _shops = json.decode(response.body);
      });
    } else {
      throw Exception('Failed to load shops');
    }
  } catch (error) {
    print('Error: $error');
  }
}
```

Figure 7. Code to fetch shop data.

During the system implementation phase, several key codes were implemented to ensure the proper functioning of the GSD system. For instance, in the Model layer, code snippets were created to handle database operations, such as retrieving and updating product information, managing user authentication and authorization, and processing order transactions. These codes utilized SQL queries to interact with the MySQL database, ensuring efficient data management and retrieval.

In the View layer, Flutter was utilized to design and develop the user interfaces of the GSD system. Flutter, a cross-platform framework, allowed for the creation of visually appealing and interactive interfaces using Dart programming language. With Flutter, the UI components and screens, including product listings, shopping carts, and order placement, were implemented. Flutter's rich set of pre-built widgets and customizable UI elements facilitated the creation of engaging and responsive user interfaces. Additionally, Flutter's hot-

reload feature enabled quick iteration and instant feedback during the UI development process. By leveraging Flutter's capabilities, the GSD system achieved a consistent and aesthetically pleasing user experience across multiple platforms.

In the Controller layer, Node.js codes were implemented to handle the routing and logic behind user requests. These codes defined the routes and corresponding functions to be executed when a specific URL was accessed. They processed incoming requests, retrieved and manipulated data from the Model layer, and rendered the appropriate views from the View layer to provide users with the requested information and functionality.

Overall, these implemented codes were crucial in bringing the GSD system to life, enabling seamless interaction, data management, and system operations for both users and administrators. This feature not only eases the workload for the developer but also reduces the time typically spent waiting for the application to compile, thereby enhancing productivity and efficiency.

```
Future<UserObject?> login() async {
  Map<String, String> customHeaders = {"content-type": "application/json"};

  var url = Uri.http('192.168.100.43:3000', 'users/login');
  var response = await http.post(url,
    headers: customHeaders,
    body: jsonEncode({'email': '$_email', 'password': '$_password'}));
  print('Response status: ${response.statusCode}');
  print('Response body: ${response.body}');
  if (response.statusCode == 200) {
    UserObject userObject = UserObject.fromJson(json.decode(response.body));
    return userObject;
  } else {
    return null;
  }
}
```

Figure 8. Post request to the backend.

The user interface (UI) of the GSD system was carefully designed to provide an intuitive and seamless shopping experience for users. The UI elements were thoughtfully arranged to ensure easy navigation and efficient access to different functionalities. The use of clean and visually appealing layouts, along with appropriate color schemes and typography, helped create an aesthetically pleasing interface that users could engage with effortlessly.

The UI incorporated various features to enhance user interactions. User-friendly forms were implemented to enable easy input of information, such as user registration details and product quantities. Interactive elements, such as buttons and icons, provided clear indications of actions that users could take, such as adding items to the cart or proceeding to checkout. Visual feedback, such as loading indicators and success/error messages, kept users informed about the progress of their actions and any relevant updates.

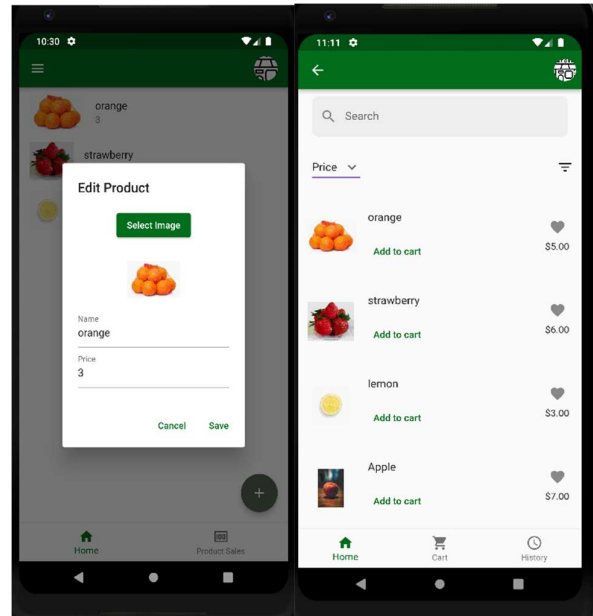


Figure 9. User Interface of Edit product page and User Interface of <Edit product page>

V. TESTING

This section defines the test cases for a particular use case. In addition, defining test cases enables developers and Quality assurance personnel to maintain and test the functionality of every use case within the system. Having test cases that cover the majority of the system's functionalities is an effective method to ensure that the system is in good condition.

Defining test cases for each use case is a crucial aspect of the testing process, as it helps ensure the functionality, reliability, and quality of the system. Test cases serve as a set of instructions and expected outcomes that guide developers and quality assurance personnel in verifying that the system performs as intended. By defining comprehensive test cases, the system can be thoroughly tested, identifying any issues or bugs that need to be addressed.

Test cases encompass various scenarios and inputs to cover different paths and functionalities within the system. They include positive test cases, which validate that the system behaves correctly under normal conditions, and negative test cases, which verify that the system handles unexpected or invalid inputs gracefully. Additionally, edge cases are considered to test the boundaries of the system and ensure it handles exceptional situations correctly.

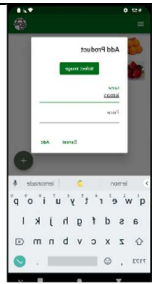
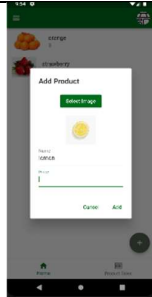
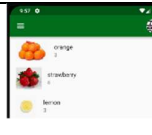
G. Black-Box Testing

During the testing phase of this system, Black Box testing was the primary technique utilized. This testing technique focuses on testing the system by inputting specific data and examining the output that would be generated based on that data.

In addition, this testing method was chosen because it does not require the tester to understand the system's underlying architecture. All of the results of the system's Black Box testing can be found in the STD document included in the appendix of

this report. Here is an example of Blackbox testing as shown in the Table I below.


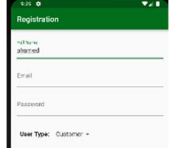
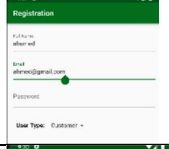
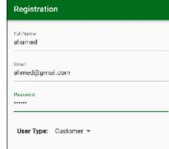
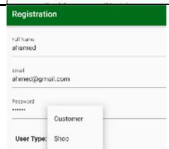

TABLE I. BLACK-BOX TESTING

Test Case ID	Input data/Action	Expected result	Actual result	Pass / Fail
TC003_01_01	Action: User add product name	Name field highlights		Pass
TC003_01_02	Action: User add images	Image inserted		Pass
TC003_01_03	Action: User clicks add	Product added to the shop		Pass

H. User Acceptance Testing (UAT)

During the testing phase of this system, User Acceptance testing was implemented as the chosen form of Black Box testing. During this phase, relevant consumers who will interact with the system were selected in order to implement the test cases designed in earlier phases of this project. Consequently, relevant users will evaluate this system according to the test cases that have been designed. The STD contains the results of the User Acceptance testing performed on this developed system. In addition, this project's appendix contains the STD document. Here is an example of UAT testing as shown in Table II below.

TABLE II. USER ACCEPTANCE TESTING

Test Title		Priority #	Test Case ID	Test Execution Date		
Test for UC001: Test user registration		1	TC01	20/6/2023		
Test Description		Test Designed By		Test Executed By		
Test for UC001: Test to ensure users able to register in the application		Mohamed khalid		Stakeholder		
Step ID	Step Description	Test Date	Expected Result	Actual Result	Pass (P)/ Fail (F)	Additional Notes
001	Click on "register now"	20/6/2023	Redirected to registration page		P	NIL
002	Click on "Full name"	20/6/2023	Redirect to fill names		P	NIL
003	Click on "Email"	20/6/2023	Redirect to fill email		P	NIL
004	Click on "Password"	20/6/2023	Redirect to fill password		P	NIL
005	Click on "user type"	20/6/2023	Option popup		P	NIL
006	Click on "register"	20/6/2023	Redirect to main page		P	NIL

VI. CONCLUSION

The Grocery Shop Delivery (GSD) system has achieved its defined objectives by effectively gathering and discussing system requirements, implementing a well-designed solution, and conducting thorough testing. The system provides a user-

friendly online grocery delivery platform that meets the needs of its users. Suggestions for future improvements, such as incorporating a steps counter, joint workout sessions, and a communication forum, offer opportunities for further enhancing the system's functionality and user experience. Overall, the GSD system has demonstrated its capability to deliver a reliable and efficient online grocery shopping solution, with potential for continuous growth and adaptation to meet evolving user demands.

ACKNOWLEDGMENT

We would like to express our sincere gratitude to Faculty of Computing and Universiti Teknologi Malaysia (UTM) for their invaluable support and resources, which played a crucial role in the successful research.

REFERENCES

- [1] Kvalsvik, F. (2022). Understanding the role of situational factors on online grocery shopping among older adults. *Journal of Retailing and Consumer Services*, 68, 103009. <https://doi.org/10.1016/j.jretconser.2022.103009>
- [2] Morganosky, M. A., & Cude, B. J. (2000). Consumer response to online grocery shopping. *International Journal of Retail & Distribution Management*, 28(1), 17–26. <https://doi.org/10.1108/09590550010306737>
- [3] Hanus, G. (2016). CONSUMER BEHAVIOUR DURING ONLINE GROCERY SHOPPING. *CBU International Conference Proceedings*, 4, 010–013. <https://doi.org/10.12955/cbup.v4.737>
- [4] Fox, M. A. (2006). Reflections on: Online grocery shopping. *First Monday*. <https://doi.org/10.5210/fm.v0i0.1582>
- [5] The City Cook. (2016, February 12). *Why Grocery Shopping Matters*. <https://www.thecitycook.com/articles/2016-02-11-grocery-shopping-matters#:~:text=Grocery%20shopping%20is%20where%20the,you're%20about%20to%20eat>
- [6] Browne, M. (2020, October 16). *Retention of online growth looks strong, but a COVID-19 recession looms ahead*. Supermarket News. <https://www.supermarketnews.com/online-retail/retention-online-growth-looks-strong-covid-19-recession-looms-ahead>
- [7] Shoemaker, M. S. S. (2022, March 21). *Walmart Plus Review 2022: A Dietitian's Personal Take*. Healthline. <https://www.healthline.com/nutrition/walmart-grocery-delivery-review#:~:text=Walmart%2B%20could%20be%20a%20good,their%20groceries%20from%20Walmart%20stores>
- [8] GroceryHackers. (2022, May 7). *Dumpling Review – Is It Any Good?* <https://groceryhackers.com/grocery-delivery/companies/dumpling/>
- [9] Zee Krstic and Amina Lake Abdelrahman, Good Housekeeping Institute. (2020, April 14). *10 Best Grocery Delivery Services to Use in 2022*. Good Housekeeping. <https://www.goodhousekeeping.com/food-products/g28039081/best-grocery-delivery-services/>
- [10] P. (2018, June 26). *Peapod Makes Grocery Shopping Easier Than Ever With New Text To Order Launch*. Peapod. <https://www.pnewswire.com/news-releases/peapod-makes-grocery-shopping-easier-than-ever-with-new-text-to-order-launch-663700733.html>
- [11] Jain, A. (2022, June 9). *How To Build An Online Grocery Store Like Peapod*. Oyelabs - Driving Business Value. <https://oyelabs.com/build-online-grocery-store-like-peapod/>