Informatics Large Practical CW2

Exam No: B188728

Contents

1 Reflection

1.1 Where did the requirements change, why and how?

During the development of PizzaDronz in coursework-2, several changes were observed to my understanding of the initial requirements of the project. These changes necessitated adjustments to the implemented code. One of the notable changes is discussed below:

- The order must be validated with the order date passed in as an argument (while running the application) and not the current (today's) date

Consequences: My initial assumption was that the program would iterate through all orders each day, validating them for any potential errors. However, as I delved into the implementation for coursework 2, I discovered that the program receives a specific date as a command line argument. Consequently, the orders fetched from the REST API[1] are exclusively for that specified date. To accommodate this change, I had to revise the code by eliminating checks that verified whether the order date corresponded to the current date. Also, I adjusted the check for the credit card's expiration date to compare it with the order's date. Additionally, I modified the logic for checking whether a restaurant is closed by aligning it with the order's date. This adjustment ensures the program accurately validates orders based on the provided date parameter.

[1]REST API is an interface that two computer systems use to exchange information securely over the internet[17]

1.2 The Implementation of Requirements

The implementation of the PizzaDronz software involved the creation of a RestClient class to retrieve all the necessary data from the API. Despite my initial concerns about the complexity of working with REST APIs due to my limited experience, I found it to be surprisingly straightforward. There were ample online resources available on the internet for how to get started with REST [8], that guided me through the process. Following data retrieval, the system conducts order validation to ensure that only valid orders proceed to the flight path generation phase. To calculate the flight path, I have used the A-Star Algorithm[2], which guarantees a complete and optimal solution [3] that meets the project's requirements. To prevent the drone from entering the no-fly zone, I filtered out the next possible coordinates that would fall within these restricted areas.

Another project requirement was that once the drone entered the central area, it should not exit until the delivery was completed. To achieve this, I implemented a Boolean flag that activates as soon as the drone enters the central area. When the flag is set to true, it filters out any potential coordinates lying outside the central area. Thus, ensuring the drone remains within the central area until the delivery is finished. After computing flight paths for all orders on a given date, the system generates JSON files for deliveries and flight paths, along with a Geo Json file providing details about the drone's movements. To ensure the code was working as expected, I implemented rigorous unit and system testing which helped to find some bugs in the early stage of the project implementation such as null checks in RestClient.

1.3 What was hard to implement?

Implementing the A-Star Algorithm was one of the hardest parts of the whole project. During the project plan, I allocated 12 hours to implementing the flight path algorithm, however, it took me more than 20 hours to achieve a functional A-Star algorithm. Although I had studied the algorithm during the "Reasoning and Agents" course and possessed a pseudo-code to guide me, the complexity increased substantially when incorporating additional conditions for the no-fly zone and the central area constraint. Despite the time investment exceeding the initial estimate, the resulting A-Star Algorithm successfully met the project's requirements.

1.4 What needs improvement?

While I made several attempts to enhance the code leading up to the deadline, there are potential improvements that I could have implemented, as discussed below:

1. Caching Flight Paths: Upon inspecting the data fetched from the REST API, it became evident that the number of restaurants is limited, and multiple orders often originate from the same restaurant. A possible optimization could involve storing the calculated flight paths in a HashMap. By doing so, the program could reuse a previously computed flight path when handling orders from the same restaurant. Given that flight path calculations are computationally intensive, implementing this caching mechanism could significantly boost the overall efficiency of the program.

2. Validation of Drone Trajectory: Utilizing the geojson.io website to visualize the generated GeoJSON file highlighted instances where the drone occasionally traverses the edge of the no-fly zone when its movement coordinates do not fall within the restricted area. To address this, additional conditions could have been incorporated into the code. This improvement would involve considering not just the individual move coordinates but also the overall trajectory of the drone. By enhancing the condition checks, we could ensure that the drone's trajectory consistently avoids any interaction with the no-fly zone

2 PizzaDronz Client-Facing Mobile Application

2.1 Introduction

So far, our focus has been on creating a backend system capable of validating order details and generating optimal drone flight paths for valid orders. With the completion of the backend system, our focus now turns to creating a front-end mobile application. This application will empower users to choose pizzas from various restaurants and place orders seamlessly. In this essay, we will explore the design of the user interface (UI)[2] for the PizzaDronz frontend application.

2.2 Why are we considering UI?

It is important to acknowledge that the UI acts as the crucial link between the user and the backend system. According to a study, a staggering 94% of users rejected using an application solely based on its poor UI design [12]. A meticulously designed interface not only improves user experience (UX)[3] – but also plays a pivotal role in user engagement and retention [13]. The presence of clear and intuitive design elements contributes to a positive impression, helping users navigate the application seamlessly. This is particularly essential when users need to select their desired pizzas and place orders effortlessly.

2.3 Assumptions

Before implementing the UI, we are adhering to the following assumptions:

1. The application's user base is limited to University of Edinburgh students. We assume the university's MyEd[4] login would be used to create an account and login user.
2. Once the proposed user interface design has been presented, the additional backend requirements, including the Payment System and Order Tracking, will be implemented.
3. User can only order a maximum of 4 pizzas and the delivery address is by default Appleton Tower.

2.4 Characteristics and Requirements

To ensure our frontend application has a meaningful impact and is valuable to users, we conducted thorough research to identify key characteristics and requirements essential for the success of a food delivery application. The following requirements are accompanied by justifications:

1. **User-Friendly Interface:**
   - Ensure a simple and intuitive design to facilitate easy navigation.
   - Use clear and concise language with visually appealing graphics.

   Justification - A user-friendly interface is crucial for an intuitive and seamless experience, enhancing user satisfaction and engagement[13].
2. **Simplified Ordering System:**
   - The app should focus on a minimalist design with a streamlined ordering process.
   - Allow users to quickly place orders within a few taps.

   Justification - Simplifying order placement minimizes friction, letting users swiftly proceed to check-out, and enhancing productivity and convenience [10].
3. **Accessible:**

[2]Anything a user may interact with to use a digital product or service[15]
[3]The emotions and perceptions users have while interacting with the application[16]
[4]It is a gateway to web-based services within and beyond the University of Edinburgh[9]

- Implement colour blind-friendly colour scheme.
- Provide an option to easily adjust the application font size.
- Include a text-to-speech option for reading the on-screen text.

Justification - Implementing this feature, as per the disability statistics report from Edinburgh University for the academic year 2022-23, will provide accessibility to approximately 17.5% of students with disabilities [14].

4. **Real-time order Tracking:**

   - Prioritize the development of a robust real-time tracking system.
   - Provide live updates on order status, from preparation to delivery.

   Justification - Real-time order tracking instils trust in the system, creating a belief in the process and fostering increased user engagement [11].

5. **Efficient Payment Integration:**

   - Integrate a secure and efficient payment system.
   - Supports various payment methods, with an emphasis on a one-click checkout process.

   Justification - To meet the user demand for inputting details and ensuring a seamless payment experience, integration is necessary, as it is required by the backend credit card validation system [7].

6. **Feedback Loop and Rating:**

   - Incorporate a user feedback system for rating experience and providing comments.
   - Display aggregated ratings for each restaurant and pizza option.

   Justification - Research on Grubhub's food delivery system indicates that user feedback has significantly contributed to an 80% increase in revenue [4] [6].

7. **Quick Reorder Functionality:**

   - Implement a "Quick Reorder" feature that allows users to repeat their previous orders with minimal effort.

   Justification - Allowing the user to reorder their previous orders is demanded by the public in other food-based applications, such as Starbucks [18] [5].

2.5 User Flow

The following diagrams 1 and 2 show the main success scenario and alternate success scenario to be considered in designing the UI:

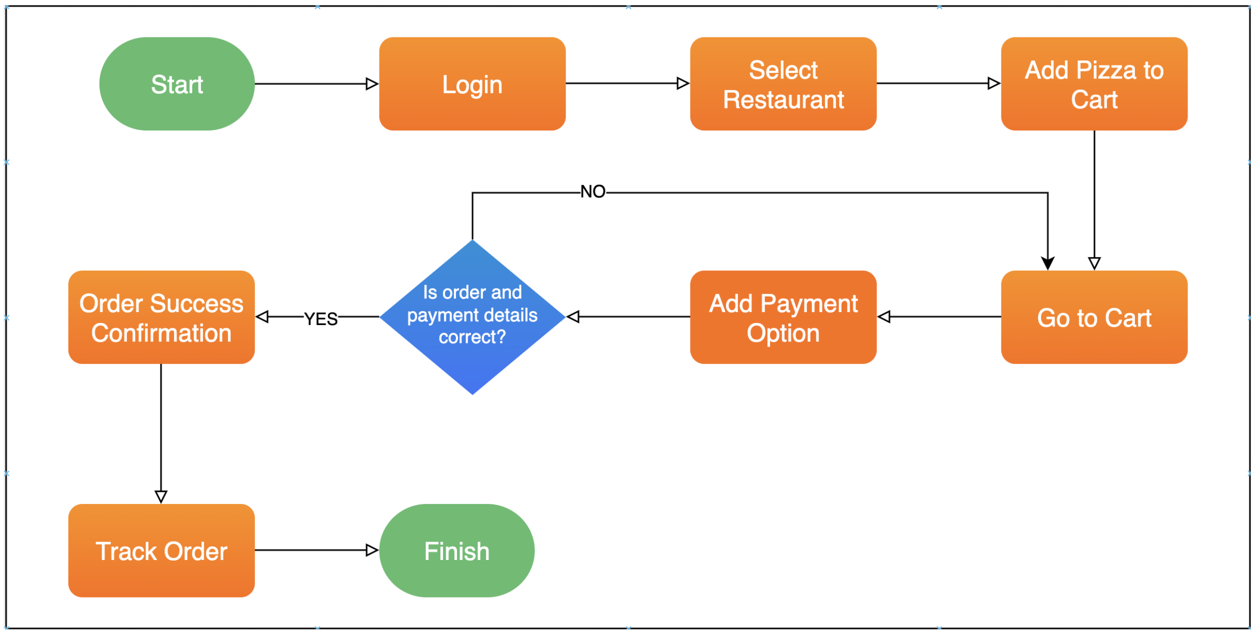


Figure 1: The main user flow success

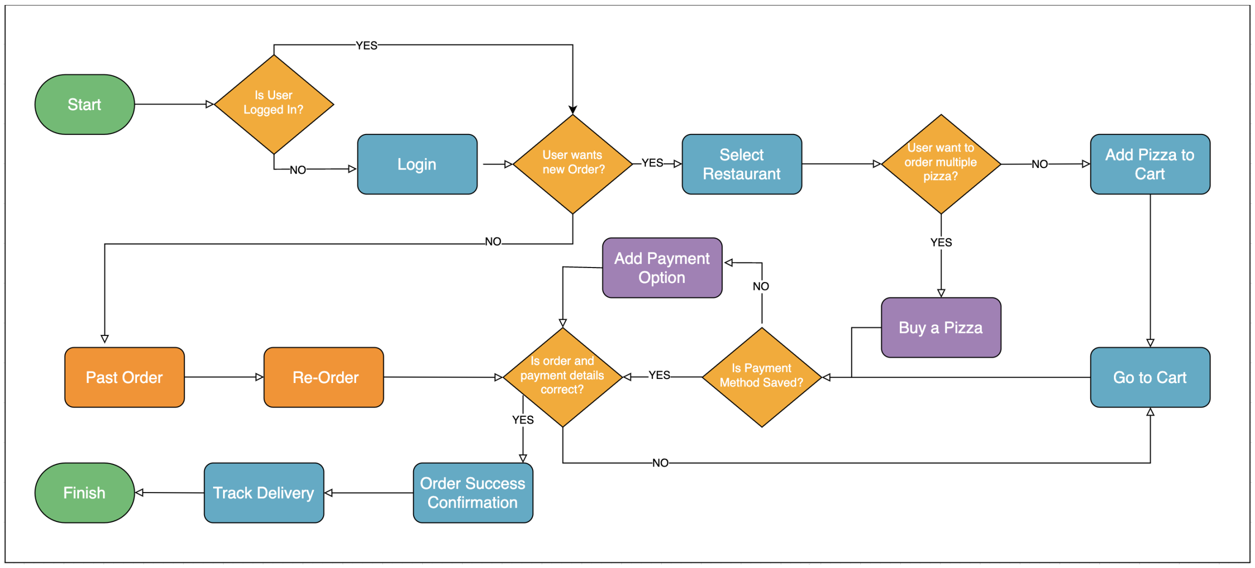
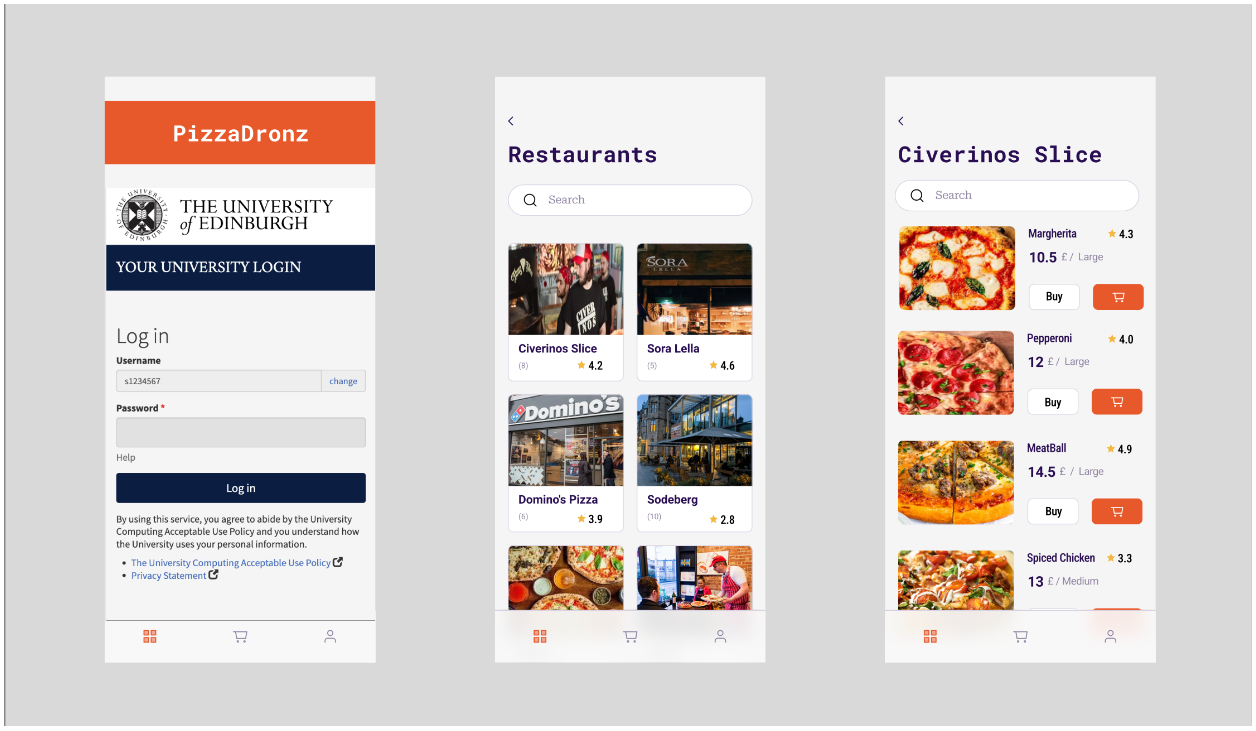


Figure 2: All alternate success flow

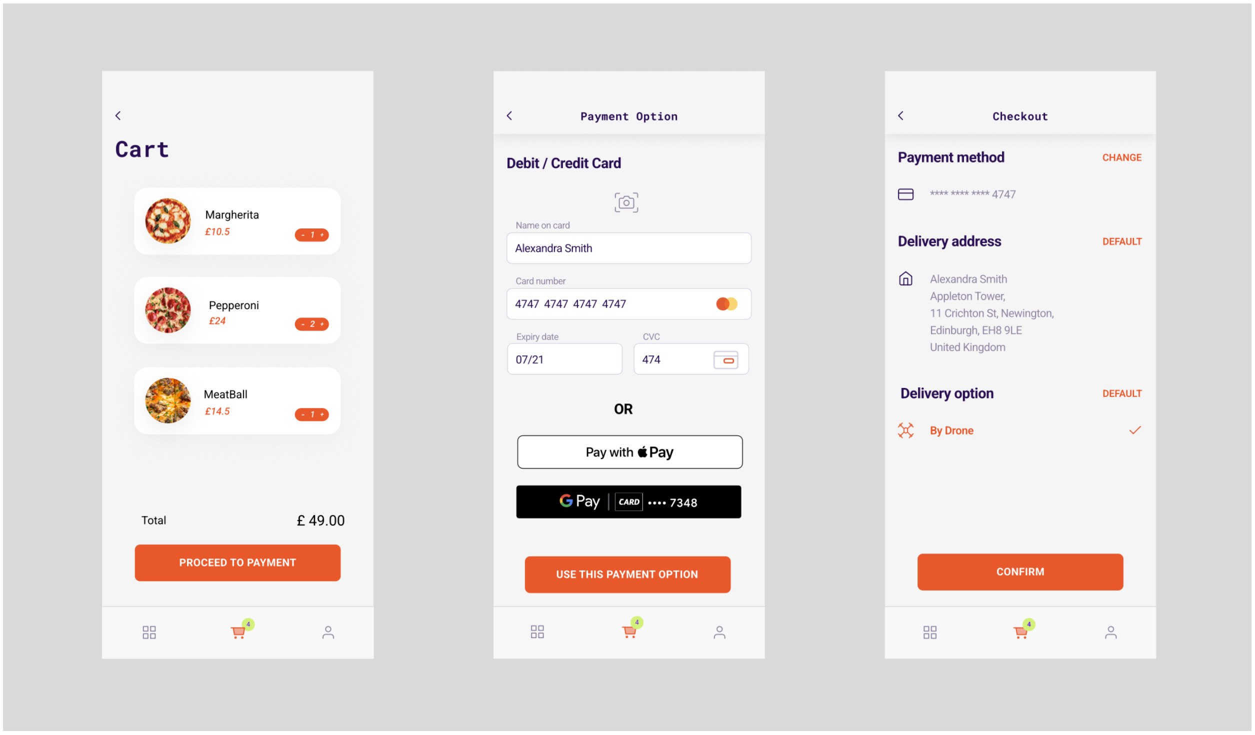


s1234567

By using this service, you agree to abide by the University Computing Acceptable Use Policy and you understand how the University uses your personal information.

- The University Computing Acceptable Use Policy
- Privacy Statement

Figure 3: The user interface for Login (using MyEd as mentioned in the Assumption), Restaurants List and Pizza Menu.



Alexandra Smith

4747 4747 4747 4747

07/21

474

**** **** **** 4747

Alexandra Smith
Appleton Tower,
11 Crichton St, Newington,
Edinburgh, EH8 9LE
United Kingdom

Figure 4: The user interface for Cart, Payment Option and Checkout Screen.

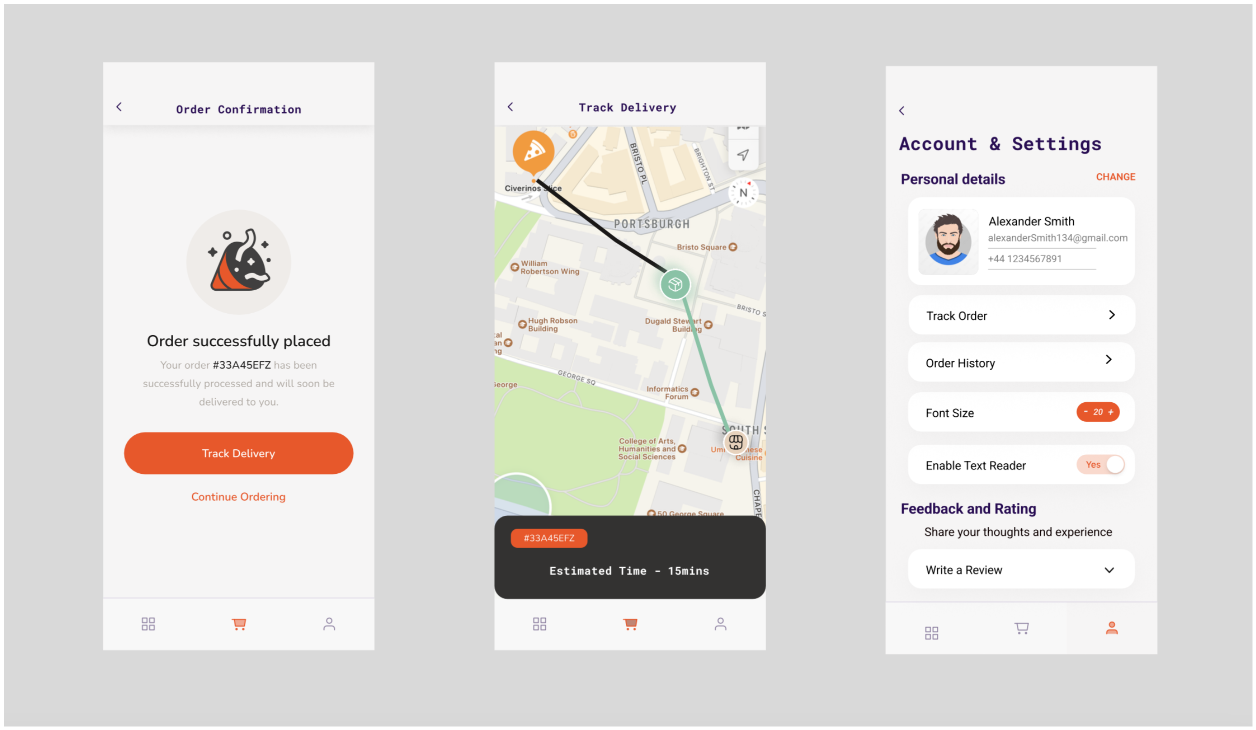


Figure 5: The user interface for Order Confirmation, Delivery Tracking and Account & Settings screen.

2.6 UI Design

We used the Figma platform to design the 9 UI screens which meet all the requirements as mentioned above as shown in Figure- 3, 4 and 5:

2.7 Design Implementation

In our Figma UI design, we've prioritized user-friendly elements by employing a consistent colour scheme and font size. Each pizza and restaurant is accompanied by appealing visuals, enhancing the overall aesthetic. For a simplified ordering system, new users follow a straightforward process spanning a maximum of four screens to select a pizza and complete their order. Existing users benefit from a streamlined experience, requiring only three screens as the payment details are securely stored. In terms of accessibility, the account and settings screen offers options to adjust font size and enable screen text-reading. The colour scheme of the UI is orange and dark blue generated from an accessible colour palette generator [1], which is friendly to a majority of users with colour blindness.

Real-time order tracking is a key feature. Upon order placement, the confirmation screen provides the option to track the live order. The order tracking screen displays the drone's real-time position and the estimated time remaining. The track order option is conveniently accessible in the account and settings section, allowing users to resume tracking even after temporarily closing the app. Efficient payment integration is ensured through a user-friendly payment options screen. Users can input credit/debit card details manually or use the mobile camera for quick scanning. Saving card details for one-tap payments is supported. Additionally, Google Pay and Apple Pay options facilitate seamless and swift payments.

The app incorporates a robust feedback loop and rating system. Aggregate ratings for each restaurant and menu option are prominently displayed. Users can contribute feedback and ratings through the "Write a Review" option in the account and settings section. A quick re-order function is provided in the account section, where recent orders are conveniently stored under "Order History." Users can effortlessly re-order their preferred items with a single tap.

2.8 Conclusion

In conclusion, the PizzaDronz mobile app's UI design focuses on simplicity and accessibility, with a MyEd login for University of Edinburgh students. Crafted in Figma, the visually appealing screens follow a streamlined ordering system, real-time tracking, and efficient payments. The orange and dark blue colour scheme ensures inclusivity for colour blindness. As development advances, the well-designed frontend, coupled with robust backend features, promises a seamless and engaging pizza-ordering experience for the university community.

References

[1] *Accessible Color Palette Generator | WCAG Compliant.* [Online; accessed 1. Dec. 2023]. Dec. 2023. URL: `https://venngage.com/tools/accessible-color-palette-generator`.

[2] *AI | Search Algorithms | A∗ Search | Codecademy.* [Online; accessed 2. Dec. 2023]. Dec. 2023. URL: `https://www.codecademy.com/resources/docs/ai/search-algorithms/a-star-search`.

[3] Muhammad Muchson Attoyibi, Faradila Emma Fikrisa, and Anik Nur Handayani. "The Implementation of A Star Algorithm (A*) In the Game Education About Numbers Introduction". In: *2nd International Conference on Vocational Education and Training (ICOVET 2018).* Atlantis Press. 2019, pp. 234–238.

[4] Alex Egg. "Online Learning for Recommendations at Grubhub". In: *Proceedings of the 15th ACM Conference on Recommender Systems.* 2021, pp. 569–571.

[5] *Grocery and Food Delivery Site UX: Allow Users to Add "Past Purchases" to the Cart from the Homepage – Articles – Baymard Institute.* [Online; accessed 1. Dec. 2023]. Dec. 2023. URL: `https://baymard.com/blog/grocery-food-delivery-orders`.

[6] Matthew Haruyama and Kazuyoshi Hidaka. "What influences users to provide explicit feedback? A case of food delivery recommenders". In: *User Modeling and User-Adapted Interaction* (2023), pp. 1–44.

[7] *ILP coursework Specification 4.1.* [Online; accessed 25. Nov. 2023]. Nov. 2023. URL: `https://drive.google.com/file/d/1VJ7-GFU6WPsu1omxuAeOjlBGgeD8kM-7/view?usp=drive_link`.

[8] Gayanga Kuruppu. "Getting JSON Data From a RESTful API Using JAVA - The Startup - Medium". In: *Medium* (Dec. 2021). URL: `https://medium.com/swlh/getting-json-data-from-a-restful-api-using-java-b327aafb3751`.

[9] *MyEd Portal.* [Online; accessed 2. Dec. 2023]. Dec. 2023. URL: `https://www.ed.ac.uk/information-services/computing/comms-and-collab/myed-portal`.

[10] Chinedu Niicholas Ogeleka. "Online Food Ordering System". In: (2019).

[11] AHM Shamsuzzoha and Petri T Helo. "Real-time tracking and tracing system: Potentials for the logistics network". In: *Proceedings of the 2011 international conference on industrial engineering and operations management.* 2011, pp. 22–24.

[12] Elizabeth Sillence et al. "Trust and mistrust of online health sites". In: *Proceedings of the SIGCHI conference on Human factors in computing systems.* 2004, pp. 663–670.

[13] S Sridevi. "User interface design". In: *International Journal of Computer Science and Information Technology Research* 2.2 (2014), pp. 415–426.

[14] *Statistics.* [Online; accessed 1. Dec. 2023]. Dec. 2023. URL: `https://www.ed.ac.uk/student-disability-service/about/statistics`.

[15] *UI vs UX | Difference Between UI and UX | What is UX or UI.* [Online; accessed 2. Dec. 2023]. Dec. 2023. URL: `https://www.usertesting.com/resources/topics/ui-vs-ux`.

[16] *User Experience*. [Online; accessed 2. Dec. 2023]. Apr. 2020. URL: `https://www.productplan.com/glossary/user-experience`.

[17] *What is RESTful API? - RESTful API Explained - AWS*. [Online; accessed 2. Dec. 2023]. Dec. 2023. URL: `https://aws.amazon.com/what-is/restful-api`.

[18] *Yelowsoft*. [Online; accessed 1. Dec. 2023]. Nov. 2023. URL: `https://www.yelowsoft.com/blog/important-factors-for-food-delivery-app-development`.