**Software Development Plan**

**Breakfast Delivery App (Luxor is a smart city)**

**10 / 03 / 2023**

Version1.1

Version 1.2

Version 2

Version 3

Version 4.1

Version 4.2

Version 5

Version 6

**Presented To:**

Egyptian families, Shops of breakfast and men from the age of 12 to 20.

**Submitted By:**

FCI Students.

**REVISION HISTORY**

| **Date** | **Author** | **Version** | **Description** |
| --- | --- | --- | --- |
| 19\3\2023 | Soha | Team Skills | Adding Team’s skills |
| 21\3\2023 | Ziad | 4.1 Use case | Adding use case |
| 21\3\2023 | Adham | 4.2 Use case senario | Adding Use case senario |
| 15\4\2023 | Soha | 5.Add prototype | Add prototype |
| 20/4/2023 | Soha | Gantt Chart | Make project plan |
| 26/4/2023 | Adham & Naira | FEASIBILITY STUDY VALIDATION PLAN | Risk Identification Description Prioritization Prototyping |
| 3/5/2023 | Rowan | CONFIGURATION AND VERSION CONTROL | manages different versions of configuration items |
| 12/5/2023 | Ziad | Tools & ARCHITECTURE | Tools and ARCHITECTURE |

**1. PRODUCT DESCRIPTION**

The program linked those who want breakfast (foul and falafel) with the man who prepares the breakfast, and not any other man .The man from the area surrounding you, and this man also, we will reduce pressure on him and the many requests and problems (who came before whom and so on).

The customer, of course, will not be forgotten, as our program aims to enable him to enjoy his food with the least amount of trouble and money and with the maximum amount of comfort, whether it is at home, university, work, school, or anywhere else.

And we will rely on men from the age of 12 to 20 to deliver breakfast, and this will reduce unemployment and teach children self-reliance at an early age and protect them from vagrancy:

1- Whether you're in a hurry and need a quick breakfast on the go, or you're looking for a more satisfying meal to start your day.

2- We will help you locate the nearest breakfast man so that you can enjoy your food with the least amount of trouble and money and with the utmost comfort.

3- Enjoy the authentic taste of Egyptian cuisine in the comfort of your own place!

**2. TEAM DESCRIPTION**Mr. Abu Abdullah: Owner of a breakfast shop (SME).  
  
ENG. Reham: You will rely on her excellent skill in creating a mobile application using flutter, dart.

ENG. Adham: He will take care of all the designs to create an application that suits all users, in addition to the logo and the brand.

ENG. Soha and ENG. Ziad: They will rely on their experience in documentation to prepare the work for the rest of the team.

ENG. Naira and ENG. Rawan: They will create a strong database using "firebase" to suit the application.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Team | Skills | | | | | | |
| Project  Management | Problem solving | Firebase | Flutter | Research  skills | Design | Documentation |
| Adham | **🗸** | **🗸** | **🗴** | **🗴** | **🗸** | **🗸** | **🗸** |
| Ziad | **🗴** | **🗸** | **🗴** | **🗴** | **🗸** | **🗴** | **🗸** |
| Soha | **🗸** | **🗸** | **🗴** | **🗴** | **🗸** | **🗸** | **🗸** |
| Reham | **🗴** | **🗸** | **🗴** | **🗸** | **🗸** | **🗸** | **🗴** |
| Rawan | **🗸** | **🗸** | **🗸** | **🗴** | **🗸** | **🗴** | **🗴** |
| Naira | **🗸** | **🗸** | **🗸** | **🗴** | **🗸** | **🗴** | **🗴** |

**Missing skills:**

\*The database is new to us.

\*Design and color selection.

\*Remote work.

\*Lack of experience with the environment in which the program will operate.

\*Artificial Intelligence.

**3. SOFTWARE PROCESS MODEL DESCRIPTION**

The model of this project is agile methodology (Extreme programming) to produce high-quality software and adapt to evolving and changing requirements with mid-sized team. We want to reduce project risk, especially around tight deadlines. We focus on coding and two developers’ team together on one computer.

**4. PRODUCT DEFINITION**

**Context Diagram**

The software system should provide delivery to bring the breakfast meal from the breakfast store and present it to the customer and take the price of the order, Where the user logs in to the application and discovers the breakfast shops around it, determines the suitable breakfast place for him, chooses the meals and the required number, presses to confirm the request, then gets a code, then the owner of the shop determines if he will accept or reject it, if he accepts it, he prepares it according to a specific period and the number of requests In one period, then he sends a delivery worker for each period to the customers, who sees through the application the customers’ places and goes in the shortest way, then he writes the code to finish the delivery process and confirm the arrival successfully.

**Objectives :**

Reducing crowds (the man who prepares breakfast and the client)

Reducing harassment (the girls who go to buy breakfast)

Reducing unemployment (men and children who will deliver)

Reducing noise and problems (the man who prepares breakfast and the client)

Save time and effort (customer)

Digital Transformation (Country)

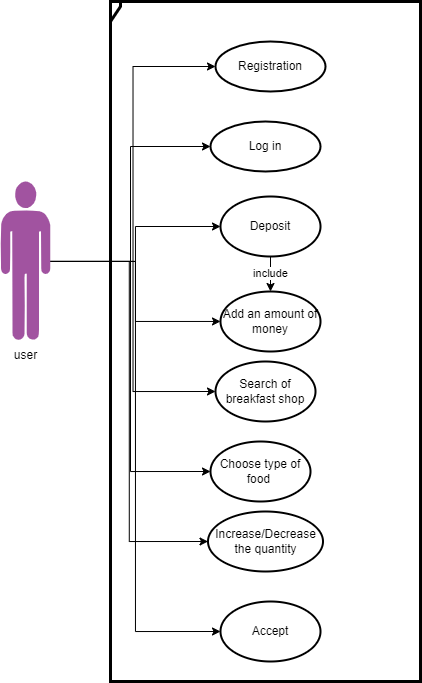
The link between culture and technology (state and tourism)

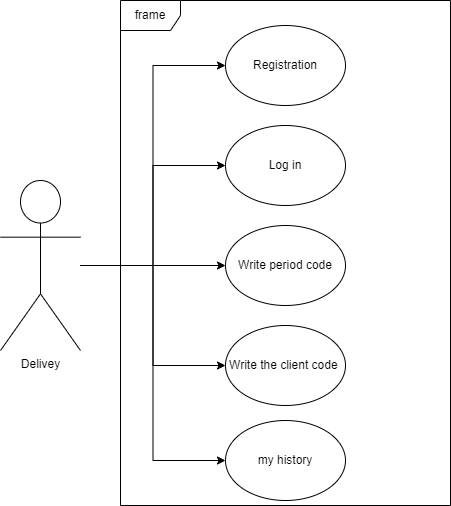
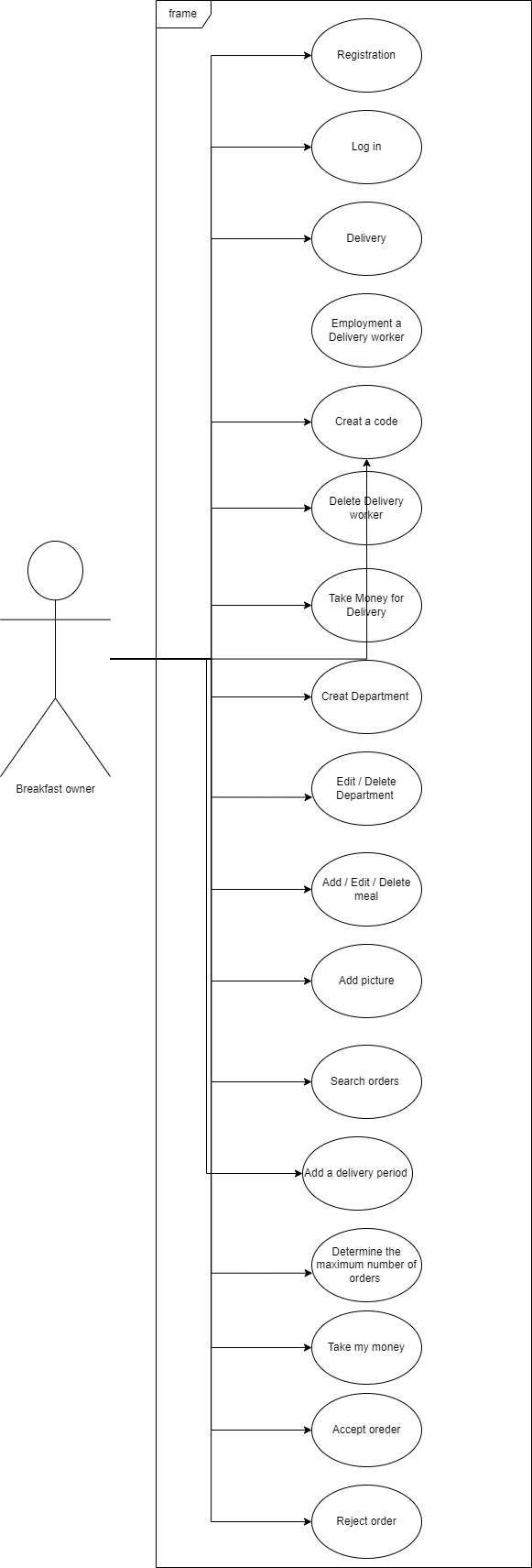
**Personas / User Characteristics**

\*The system users (Egyptian family / Delivery) should not require special training.

\*The owner of the breakfast shop does not need training, he has a tablet that only shows orders.

**Use case Diagram**





**Non-functional requirements: -**

**》Usability:**

The user can use the application easily and the application design is comfortable.

**》High performance and Efficiency**:

-The software is not affected by the number of users at the same time.

-The software works without problems when the network is 2G,3G,4G,5G and Wi-Fi even if the network is weak.

- the software system handles capacity, throughput, and response time.

**》Accessibility and Reliability.**

**》Confidentiality:**

the system protects sensitive data and allows only authorized access to the data.

**》Security and Safety:**

the system prevents harm to people or damage to the environment.

**Use Case Descriptions (Use case scenario)**

User use case

**Register:** To use app user must have an account, and enters all required information

|  |  |  |
| --- | --- | --- |
| **Use case name** | Register | |
| **Unique ID** | FD-001 | |
| **Area** | Application | |
| **Actor(s)** | User (Delivery, Owner, User) | |
| **Description** | User creates account | |
| **Triggering Event** | User click “Register” button in the application | |
| **Preconditions** | The user needs to download application then open it  The user needs to have internet access | |
| **Postconditions** | User has successfully created account | |
| **Assumptions** | A valid data | |
| **Steps Performed** | | **Information for Steps** |
| 1. Open application 2. Choose if he is a Delivery, Owner or User 3. User enters his data 4. Click on “Create Account” button 5. Validation of entered data by application | | Step 3: Username, email, Password, address, Phone number, payment method, card copy, and role |
| **Extensions**  **(Alternative Flows)** | If the downloading interrupted for any reason, use should try again and download it  If user entered a non-valid data, a warning message should appear to him | |

**Login:** To log in the app the user must enter username and password

|  |  |  |
| --- | --- | --- |
| **Use case name** | Login | |
| **Unique ID** | FD-002 | |
| **Area** | Application | |
| **Actor(s)** | User (Delivery, Owner, User) | |
| **Description** | User login to his account | |
| **Triggering Event** | User click “Login” button in the application | |
| **Preconditions** | The user needs to download application then open it  The user needs to have internet access  The user needs to have account | |
| **Postconditions** | User has successfully logged in to his account | |
| **Assumptions** | A valid data | |
| **Steps Performed** | | **Information for Steps** |
| 1. Open application 2. User enters his data 3. Click on “Login” button 4. Validation of entered data by application | | Step 2: E-mail, Password |
| **Extensions**  **(Alternative Flows)** | If user entered a non-valid data, a warning message should appear to him | |

**Deposit:** One of the ways you can buy food is to make a deposit

|  |  |  |
| --- | --- | --- |
| **Use case name** | Deposit | |
| **Unique ID** | FD-003 | |
| **Area** | Application | |
| **Actor(s)** | User (User) | |
| **Description** | This use case describes the process of deposit payment for a food delivery order. | |
| **Triggering Event** | User click “Deposit” button in the application | |
| **Preconditions** | User selects payment method. | |
| **Postconditions** | User has paid the deposit amount and received an order confirmation with the deposit amount | |
| **Assumptions** | A valid data  Availability of electronic payment methods | |
| **Steps Performed** | | **Information for Steps** |
| 1. User click “Deposit” button 2. User selects payment method. 3. User enter the deposit amount 4. confirm | | Step 2: payment method  Step 3: amount |
| **Extensions**  **(Alternative Flows)** | If user entered a non-valid data, a warning message should appear to him | |

**Search for nearest owner:** To find the nearest food delivery man.

|  |  |  |
| --- | --- | --- |
| **Use case name** | Search for nearest owner | |
| **Unique ID** | FD-004 | |
| **Area** | Application | |
| **Actor(s)** | User (User) | |
| **Description** | This use case describes the process of searching for the nearest food delivery man to place an order. | |
| **Triggering Event** | User selects “Search Now” button. | |
| **Preconditions** | User add his location on the app. | |
| **Postconditions** | User has selected food items from the nearest food delivery. | |
| **Assumptions** | User has an active internet connection and location services are enabled. | |
| **Steps Performed** | | **Information for Steps** |
| 1. User click “Search Now” button 2. User choice one of the nearest food delivery mans | | n/a |
| **Extensions**  **(Alternative Flows)** | If there are no available delivery persons nearby, the user is notified that this location is empty. | |

**Choose type of food:** choose your favorite food.

|  |  |  |
| --- | --- | --- |
| **Use case name** | Search for nearest owner | |
| **Unique ID** | FD-005 | |
| **Area** | Application | |
| **Actor(s)** | User (User) | |
| **Description** | This use case describes the process of searching for the nearest food delivery man to place an order. | |
| **Triggering Event** | User selects “Search Now” button. | |
| **Preconditions** | User add his location on the app. | |
| **Postconditions** | User has selected food items from the nearest food delivery. | |
| **Assumptions** | User has an active internet connection and location services are enabled. | |
| **Steps Performed** | | **Information for Steps** |
| 1. User click “Search Now” button 2. User choice one of the nearest food delivery mans | | n/a |
| **Extensions**  **(Alternative Flows)** | If there are no available delivery persons nearby, the user is notified that this location is empty. | |

**Owner use case**

|  |  |  |
| --- | --- | --- |
| **Use case name** | Delivery | |
| **Unique ID** | FD-006 | |
| **Area** | Application | |
| **Actor(s)** | User (Breakfast shop owner) | |
| **Description** | This use case describes the next action about the delivery mans. | |
| **Triggering Event** | User selects “Delivery” button. | |
| **Preconditions** | The Owner has logged in | |
| **Postconditions** | The owner selects the appropriate action. | |
| **Assumptions** | Decide on an employee | |
| **Steps Performed** | | **Information for Steps** |
| 1. User click “Delivery” button | | n/a |
| **Extensions**  **(Alternative Flows)** | .................. | |

**Add delivery man**

|  |  |  |
| --- | --- | --- |
| **Use case name** | Add delivery man | |
| **Unique ID** | FD-007 | |
| **Area** | Application | |
| **Actor(s)** | User (Breakfast shop owner) | |
| **Description** | This use case describes the process of hiring a delivery man. | |
| **Triggering Event** | User selects “Add delivery man” button. | |
| **Preconditions** | Create a unique code for the new employee | |
| **Postconditions** | The delivery worker is ready to work for the application employer. | |
| **Assumptions** | The shop owner needs to hire a new worker | |
| **Steps Performed** | | **Information for Steps** |
| 1. User click “Add delivery man” button 2. Write the delivery man 's name 3. Create a unique code for the delivery man 4. User click “Accept” button to confirm the operation | | Step 2, 3: man’s name and code. |
| **Extensions**  **(Alternative Flows)** | delivery man will have unique code and no one can use it. | |

**Delete delivery man**

|  |  |  |
| --- | --- | --- |
| **Use case name** | Delete delivery man | |
| **Unique ID** | FD-008 | |
| **Area** | Application | |
| **Actor(s)** | User (Breakfast shop owner) | |
| **Description** | This use case describes the process of Delete a delivery man. | |
| **Triggering Event** | User selects “Delete delivery man” button. | |
| **Preconditions** | The client is on the application and has a unique code | |
| **Postconditions** | The delivery man has no money and no unique code. | |
| **Assumptions** | The shop owner needs to delete a new worker | |
| **Steps Performed** | | **Information for Steps** |
| 1. User click “Delete delivery man” button 2. Choose the delivery man 's name 3. User click “Accept” button to confirm the operation | | Step 2: man’s name and code. |
| **Extensions**  **(Alternative Flows)** | IF delivery man has unique code, the code will delete. | |

**Withdraw money to the customer**

|  |  |  |
| --- | --- | --- |
| **Use case name** | Withdraw money to the customer | |
| **Unique ID** | FD-009 | |
| **Area** | Application | |
| **Actor(s)** | User (Breakfast shop owner) | |
| **Description** | This use case describes the process of Withdraw money to the customer. | |
| **Triggering Event** | User selects “Take money for Delivery” button. | |
| **Preconditions** | Write the name and code of the delivery man | |
| **Postconditions** | The delivery man money will decrease. | |
| **Assumptions** | The delivery man has money and work with this owner | |
| **Steps Performed** | | **Information for Steps** |
| 1. User click “Take money for Delivery” button 2. Write the delivery man 's name and code 3. User click “Accept” button to confirm the operation | | Step 2, 3: man’s name and code. |
| **Extensions**  **(Alternative Flows)** | Money more than 1 pound | |

**Create Department**

|  |  |  |
| --- | --- | --- |
| **Use case name** | Create Department | |
| **Unique ID** | FD-010 | |
| **Area** | Application | |
| **Actor(s)** | User (Breakfast shop owner) | |
| **Description** | This use case describes the process of creating new department. | |
| **Triggering Event** | User selects “Create Department” button. | |
| **Preconditions** | A new type has added and this type did not exist before | |
| **Postconditions** | The owner will add a new meal. | |
| **Assumptions** | The new type has not existed before | |
| **Steps Performed** | | **Information for Steps** |
| 1. User click “Create Department” button 2. Write the Department 's name 3. Add picture 4. User click “Accept” button to confirm the operation | | n/a |
| **Extensions**  **(Alternative Flows)** | If there two Department with the same name, the application sends error message | |

**Delete Department**

|  |  |  |
| --- | --- | --- |
| **Use case name** | Delete Department | |
| **Unique ID** | FD-011 | |
| **Area** | Application | |
| **Actor(s)** | User (Breakfast shop owner) | |
| **Description** | This use case describes the process of deleting department. | |
| **Triggering Event** | User selects “Delete Department” button. | |
| **Preconditions** | A type has already existed | |
| **Postconditions** | This type will be deleted and the meals that contain it will be deleted from it. | |
| **Assumptions** | The type has already existed | |
| **Steps Performed** | | **Information for Steps** |
| 1. User click “Delete Department” button 2. choose Department 's name 3. User click “Accept” button to confirm the operation | | n/a |
| **Extensions**  **(Alternative Flows)** | If there is no Department with this name, the application sends error message | |

**Delete** **Meal**

|  |  |  |
| --- | --- | --- |
| **Use case name** | Delete Meal | |
| **Unique ID** | FD-012 | |
| **Area** | Application | |
| **Actor(s)** | User (Breakfast shop owner) | |
| **Description** | This use case describes the process of deleting meal. | |
| **Triggering Event** | User selects “Delete meal” button. | |
| **Preconditions** | A meal has already existed | |
| **Postconditions** | This meal will be deleted from this type. | |
| **Assumptions** | The meal has already existed | |
| **Steps Performed** | | **Information for Steps** |
| 1. User click “Delete meal” button 2. choose meal 's name 3. User click “Accept” button to confirm the operation | | n/a |
| **Extensions**  **(Alternative Flows)** | If there is no meal with this name, the application sends error message | |

**Add meal**

|  |  |  |
| --- | --- | --- |
| **Use case name** | Add meal | |
| **Unique ID** | FD-013 | |
| **Area** | Application | |
| **Actor(s)** | User (Breakfast shop owner) | |
| **Description** | This use case describes the process of adding new meal. | |
| **Triggering Event** | User selects “Adding meal” button. | |
| **Preconditions** | A new meal has added and this meal did not exist before | |
| **Postconditions** | Every meal has price and (increase/decrease) the quantity. | |
| **Assumptions** | The new meal has not existed before | |
| **Steps Performed** | | **Information for Steps** |
| 1. User click “Add meal” button 2. Write the meal 's name and price 3. Add picture 4. User click “Accept” button to confirm the operation | | n/a |
| **Extensions**  **(Alternative Flows)** | If there two meal with the same name, the application sends error message | |

**Edit Meal**

|  |  |  |
| --- | --- | --- |
| **Use case name** | Edit Meal | |
| **Unique ID** | FD-014 | |
| **Area** | Application | |
| **Actor(s)** | User (Breakfast shop owner) | |
| **Description** | This use case describes the process of editing meal. | |
| **Triggering Event** | User selects “Edit meal” button. | |
| **Preconditions** | A meal has already existed | |
| **Postconditions** | This meal will be deleted from this type. | |
| **Assumptions** | The meal has already existed | |
| **Steps Performed** | | **Information for Steps** |
| 1. User click “Edit meal” button 2. choose meal 's name and edit the name, price or description 3. User click “Accept” button to confirm the operation | | n/a |
| **Extensions**  **(Alternative Flows)** | If there is no meal with this name, the application sends error message | |

**Add picture**

|  |  |  |
| --- | --- | --- |
| **Use case name** | Add picture | |
| **Unique ID** | FD-015 | |
| **Area** | Application | |
| **Actor(s)** | User (Breakfast shop owner) | |
| **Description** | This use case describes the process of adding picture for new meal or department. | |
| **Triggering Event** | User selects “Add picture” button. | |
| **Preconditions** | Owner add a new meal or new department from application’s picture | |
| **Postconditions** | Every meal / department has picture. | |
| **Assumptions** | The new meal has not picture | |
| **Steps Performed** | | **Information for Steps** |
| 1. User click “Add picture” button 2. choose the meal 's name /department 3. Add picture 4. User click “Accept” button to confirm the operation | | n/a |
| **Extensions**  **(Alternative Flows)** | If their meal/ department with the same picture, the application send error message | |

**Withdraw my money**

|  |  |  |
| --- | --- | --- |
| **Use case name** | Withdraw my money | |
| **Unique ID** | FD-016 | |
| **Area** | Application | |
| **Actor(s)** | User (Breakfast shop owner) | |
| **Description** | This use case describes the process of taking money from application. | |
| **Triggering Event** | User selects “Take my money” button. | |
| **Preconditions** | Write the quantity and code of the owner | |
| **Postconditions** | The Owner money will decrease. | |
| **Assumptions** | The Owner delivery meals and earn money | |
| **Steps Performed** | | **Information for Steps** |
| 1. User click “Take my money” button 2. Write the quantity and code 3. User click “Accept” button to confirm the operation | | Step 2: code and quantity. |
| **Extensions**  **(Alternative Flows)** | Money more than 1 pound | |

**Order history**

|  |  |  |
| --- | --- | --- |
| **Use case name** | Order history | |
| **Unique ID** | FD-017 | |
| **Area** | Application | |
| **Actor(s)** | User (Breakfast shop owner) | |
| **Description** | This use case describes the process of searching about orders. | |
| **Triggering Event** | User selects “Order history” button. | |
| **Preconditions** | The owner has some orders | |
| **Postconditions** | The Owner can see his order and prepare them. | |
| **Assumptions** | The Owner has orders from his customers | |
| **Steps Performed** | | **Information for Steps** |
| 1. User click “Order history” button | | n/a |
| **Extensions**  **(Alternative Flows)** | If There is not any order, the application send “It’s empty” message | |

**Add delivery period**

|  |  |  |
| --- | --- | --- |
| **Use case name** | Add delivery period | |
| **Unique ID** | FD-018 | |
| **Area** | Application | |
| **Actor(s)** | User (Breakfast shop owner) | |
| **Description** | This use case describes the process of adding a delivery period. | |
| **Triggering Event** | User selects “Add delivery period” button. | |
| **Preconditions** | Add a limit delivery period | |
| **Postconditions** | When last period end, the next period time start. | |
| **Assumptions** | The period is suitable | |
| **Steps Performed** | | **Information for Steps** |
| 1. User click “Add delivery period” button 2. Add period and Maximum number of orders in this period 3. User click “Accept” button to confirm the operation | | Step 2: period time and Maximum number of orders. |
| **Extensions**  **(Alternative Flows)** | If orders more than Maximum number of orders, this order will be in next period | |

**Delivery use case**

**Write a period code:** to see the delivery path

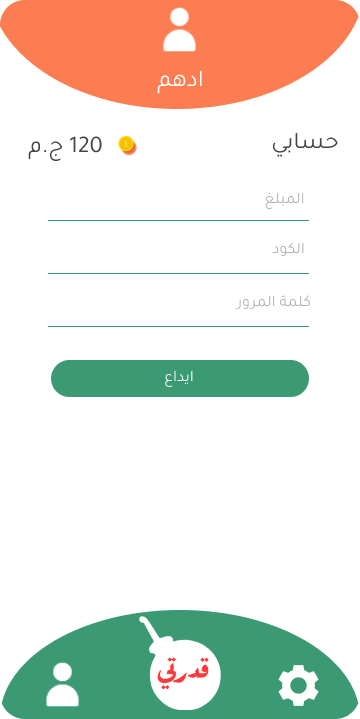
|  |  |  |
| --- | --- | --- |
| **Use case name** | Write a period code | |
| **Unique ID** | FD-019 | |
| **Area** | Application | |
| **Actor(s)** | User (Delivery) | |
| **Description** | To see the delivery path | |
| **Triggering Event** | Delivery enter the period code | |
| **Preconditions** | Th Delivery should take the code from the owner | |
| **Postconditions** | The delivery can see the path | |
| **Assumptions** | A valid period code | |
| **Steps Performed** | | **Information for Steps** |
| 1. Take the code from the owner 2. Write the code on app 3. See the path | | Step 1: E-mail, Password  Setp3: period code |
| **Extensions**  **(Alternative Flows)** | If Delivery entered a non-valid code, a warning message should appear to him | |

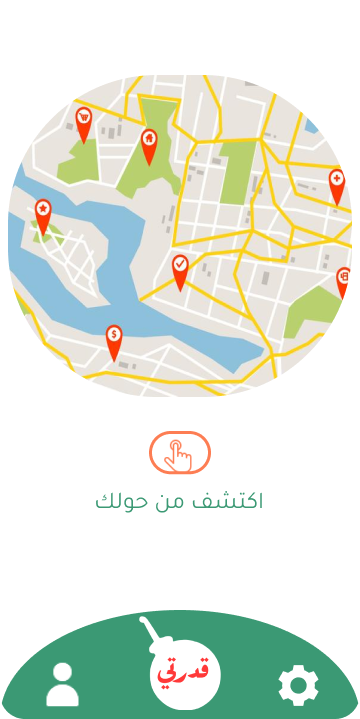
**Write the client code:** to confirm the order has been delivered.

|  |  |  |
| --- | --- | --- |
| **Use case name** | Write the client code | |
| **Unique ID** | FD-020 | |
| **Area** | Application | |
| **Actor(s)** | User (Delivery) | |
| **Description** | to confirm the order has been delivered. | |
| **Triggering Event** | Delivery enter the client code | |
| **Preconditions** | The delivery can see the path | |
| **Postconditions** | The order has been confirmed | |
| **Assumptions** | A valid client code | |
| **Steps Performed** | | **Information for Steps** |
| 1. The Delivery take the code from the user 2. The order is confirmed | | Setp1: client code |
| **Extensions**  **(Alternative Flows)** | If user entered a non-valid code, a warning message should appear to him | |

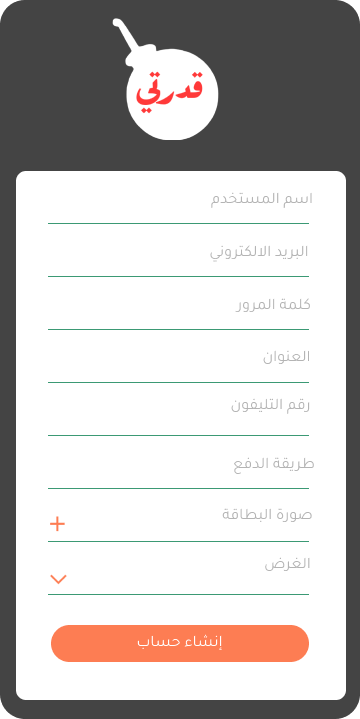
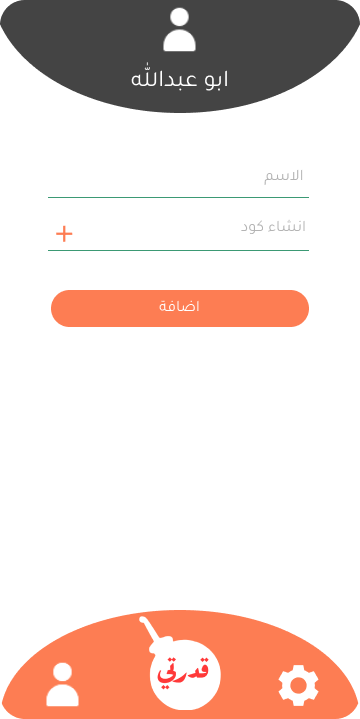
**History:** to see delivery history.

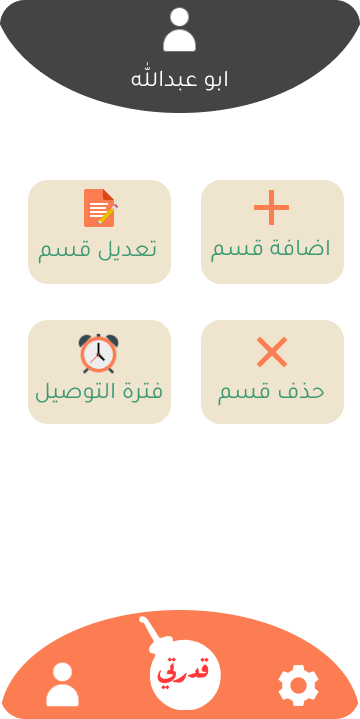
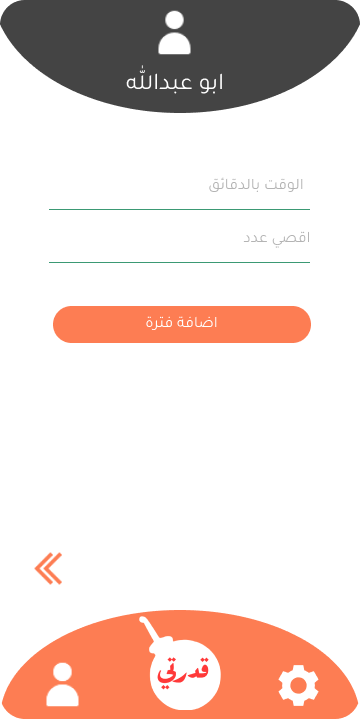
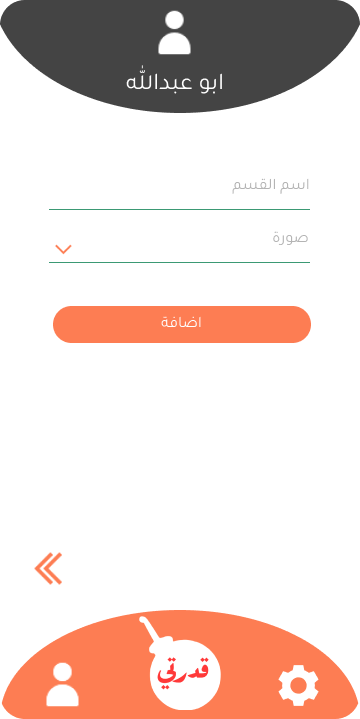
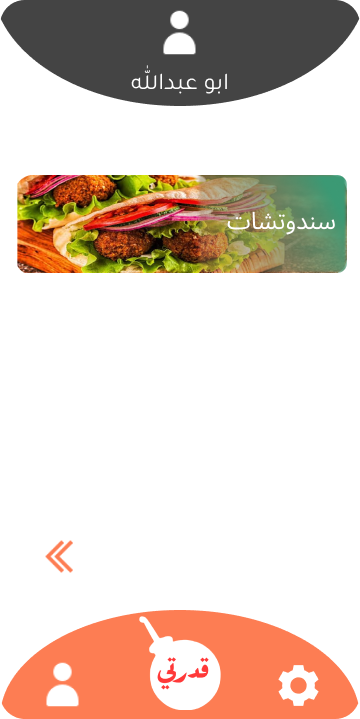
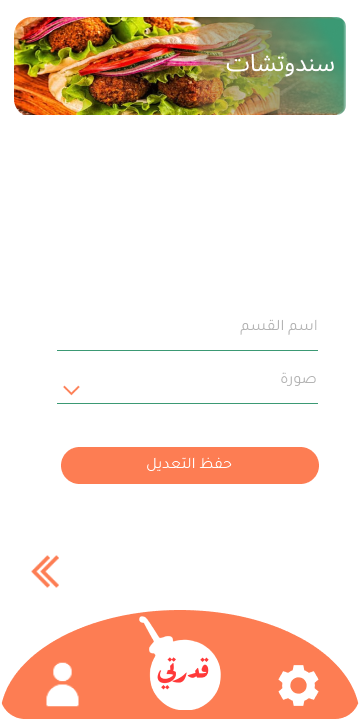
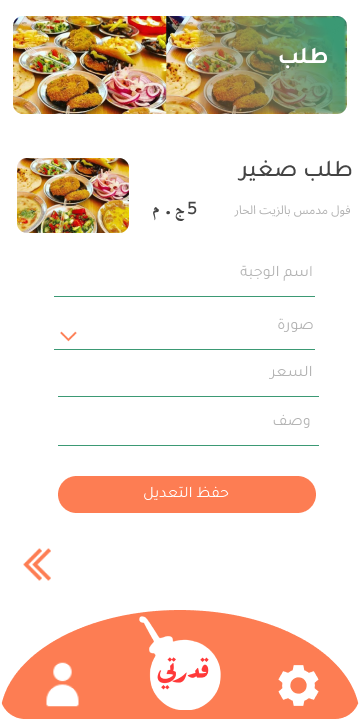
|  |  |  |
| --- | --- | --- |
| **Use case name** | Write the client code | |
| **Unique ID** | FD-021 | |
| **Area** | Application | |
| **Actor(s)** | User (Delivery) | |
| **Description** | to see delivery history | |
| **Triggering Event** | Delivery click "My History" | |
| **Preconditions** | There is delivered orders | |
| **Postconditions** | Delivery can see his history | |
| **Assumptions** | ……………….. | |
| **Steps Performed** | | **Information for Steps** |
| Delivery click "My History" | | n/a |
| **Extensions**  **(Alternative Flows)** | If there is no history, show message that there is no history. | |

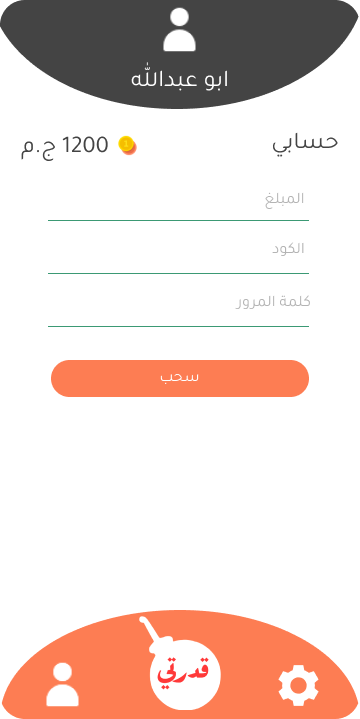
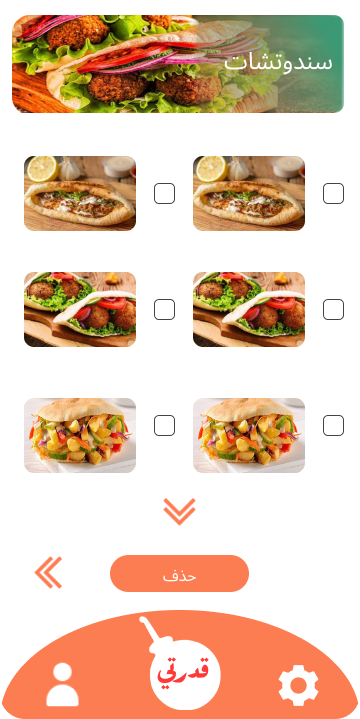
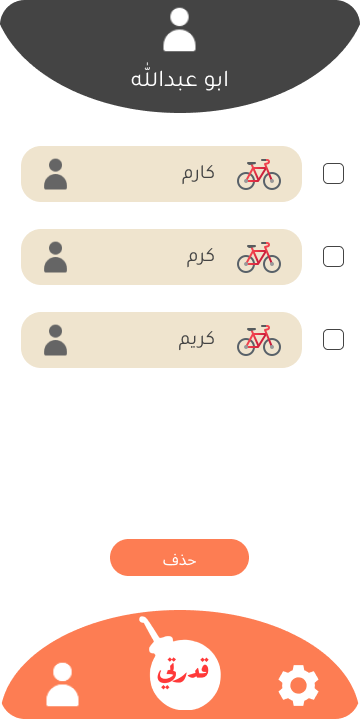
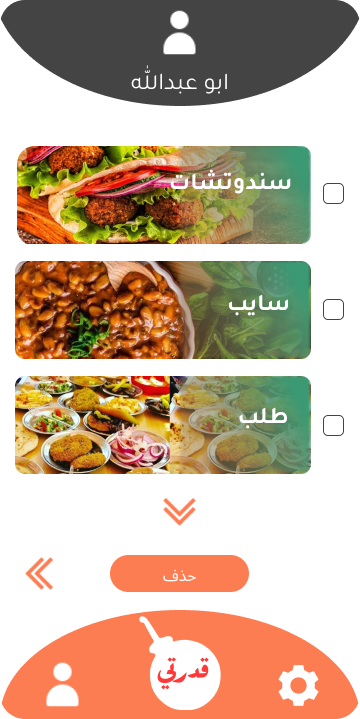
**5. USER EXPERIENCE WIREFRAMES**



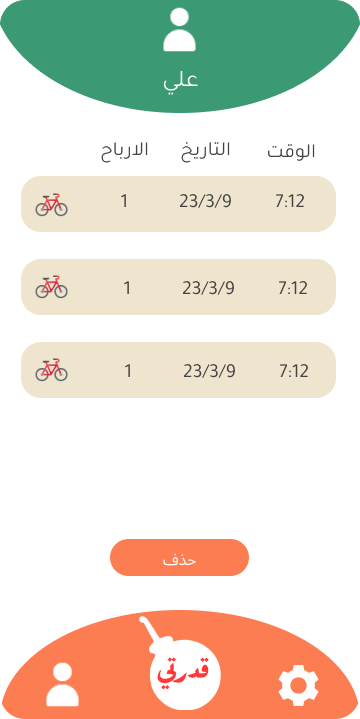
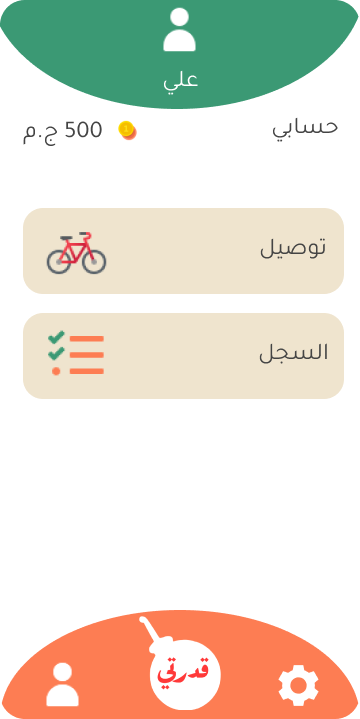


****

****

****

****

****

**6. PROJECT ORGANIZATION**

Breakdown of major tasks and schedule

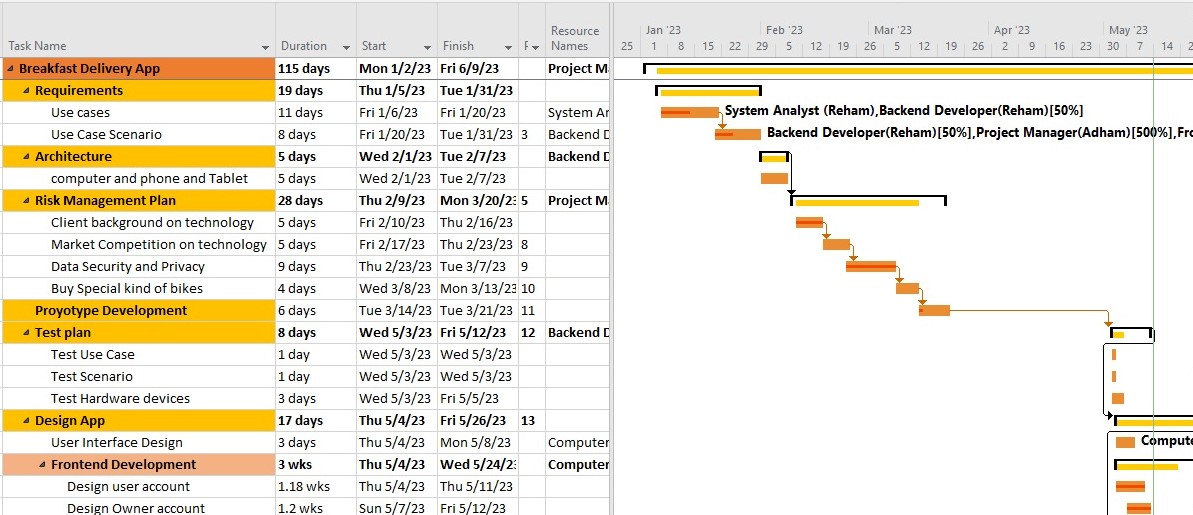
**# Reham** is responsible for Mob-App and **Adham** responsible for Design.

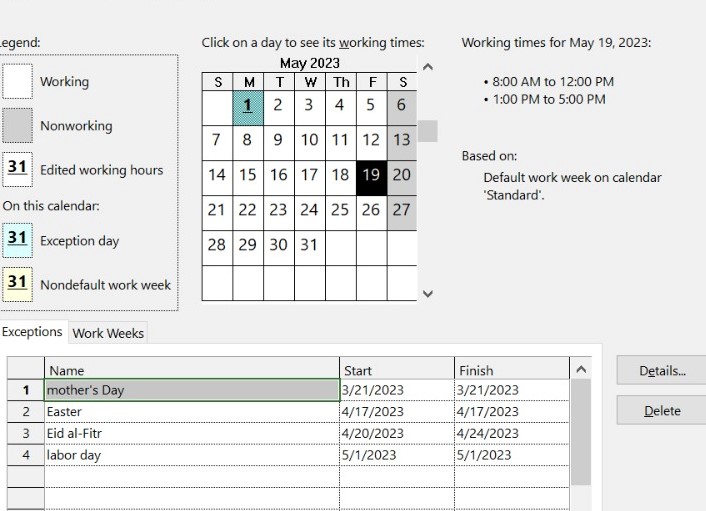
\*Missing experience is the Firebase, and **Rowan** learns it well. Where you search for educational courses and train more and more.

We use agile methodology, Reham is responsible for Mob-App, Adhamresponsible for Design and manager and System Analyst, Soha developer and Quality Assurance (QA) doucs and Naira, Rowan and Ziad are developers.

Defines the high level which team members are responsible for which tasks

**PERT / Gantt Chart**

****

****

**7. VALIDATION PLAN**

More than 15 use cases have been implemented which is about 80% of the application as a whole (Design, ddocumentation and 75% of the Mob app)

We expect it to be 99.5% successful after its release, as it will be used by many Egyptian families in popular areas, as well as workers in the workplace. We also expect that it will spread very quickly. As it will save a lot of time and effort for many people

**8. FEASIBILITY STUDY**

**1. Risk Identification:**

1. Client background on technology
2. Customer satisfaction and retention
3. Market competition
4. Data security and privacy

**2. Description of the risks:**

**1. Client background on technology:**

Risk: The client may have limited knowledge of technology, which could lead to unrealistic expectations or difficulty in understanding the app's features and functionalities.

Mitigation: Provide clear documentation and training materials to help the client understand the technology behind the app. Maintain open communication channels to address any questions or concerns the client may have. Set realistic expectations and ensure that the client is involved in the development process to avoid misunderstandings.

**2. Customer satisfaction and retention:**

Risk: Poor user experience, app performance issues, or unmet customer needs can lead to dissatisfaction and loss of customers.

Mitigation: Conduct thorough market research to understand customer needs and preferences. Design the app with a user-centric approach, focusing on usability, accessibility, and performance. Regularly gather user feedback and make necessary improvements to enhance customer satisfaction. Implement a robust customer support system to address any issues promptly.

**3. Market competition:**

Risk: Competitors may offer similar services, leading to a saturated market and difficulty in differentiating the app.

Mitigation: Identify unique selling points (USPs) that set the app apart from competitors, such as exclusive partnerships with restaurants, innovative features, or competitive pricing. Continuously monitor market trends and competitor activities to stay ahead of the curve. Invest in marketing and promotional activities to increase brand awareness and attract new customers.

**4. Data security and privacy:**

Risk: Unauthorized access, data breaches, or misuse of customer data can lead to legal issues, loss of customer trust, and damage to the app's reputation.

Mitigation: Implement strong security measures, such as encryption, secure authentication, and access control mechanisms. Regularly update and patch the app to address any security vulnerabilities. Develop a comprehensive privacy policy and ensure compliance with relevant data protection regulations. Educate users on the importance of data security and privacy, and provide them with tools to manage their personal information.

**3. Risk Prioritization:**

Risk prioritization helps allocate resources and focus on the most critical risks first.

**1. Data security and privacy:**

Given the sensitive nature of customer data and the potential legal and reputational consequences of a data breach, this risk should be prioritized. Ensuring robust security measures and compliance with data protection regulations is crucial to maintain customer trust and avoid potential legal issues.

**2. Customer satisfaction and retention:**

The success of a food delivery app largely depends on its ability to meet customer needs and provide a seamless user experience. Prioritizing customer satisfaction and retention is essential to ensure the app's long-term viability and growth. Focus on user-centric design, performance optimization, and continuous improvement based on user feedback.

**3. Market competition:**

In a competitive market, differentiating the app from its competitors is vital to attract and retain customers. Prioritizing market competition helps ensure that the app remains relevant and innovative. Identify unique selling points, monitor market trends, and invest in marketing and promotional activities to stay ahead of the competition.

**4. Client background on technology:**

While it's important to manage client expectations and ensure they understand the app's features and functionalities, this risk can be addressed through effective communication and documentation. Prioritizing the other risks will indirectly contribute to managing this risk, as a secure, user-friendly, and competitive app will likely meet the client's expectations.

**4. Risk Mitigation:**

**1. Client background on technology:**

Mitigation: Educate the client on the technology behind the app and provide clear documentation and training materials. Maintain open communication channels to address any questions or concerns. Involve the client in the development process to set realistic expectations and ensure their understanding of the app's features and functionalities.

**2. Customer satisfaction and retention:**

Mitigation: Conduct market research to understand customer needs and preferences. Design the app with a user-centric approach, focusing on usability, accessibility, and performance. Regularly gather user feedback and make necessary improvements to enhance customer satisfaction. Implement a robust customer support system to address any issues promptly.

**3. Market competition:**

Mitigation: Identify unique selling points (USPs) that set the app apart from competitors, such as exclusive partnerships with restaurants, innovative features, or competitive pricing. Continuously monitor market trends and competitor activities to stay ahead of the curve. Invest in marketing and promotional activities to increase brand awareness and attract new customers.

**4. Data security and privacy:**

Mitigation: Implement strong security measures, such as encryption, secure authentication, and access control mechanisms. Regularly update and patch the app to address any security vulnerabilities. Develop a comprehensive privacy policy and ensure compliance with relevant data protection regulations. Educate users on the importance of data security and privacy, and provide them with tools to manage their personal information.

**5. Risk Mitigation Timeline:**

a. **Client background on technology**: 1-3 months

b. **Customer satisfaction and retention**: Ongoing

e. **Market competition**: 1-3 months

d**. Data security and privacy**: 2-5 months

**1. Client background on technology (Pre-development and Development phases):**

* Begin educating the client on the technology during the pre-development phase.
* Provide clear documentation and training materials as the development progresses.
* Maintain open communication channels and involve the client in the development process to set realistic expectations.

**2. Customer satisfaction and retention (Development, Testing, and Post-launch phases):**

* Conduct market research during the development phase to understand customer needs and preferences.
* Design the app with a user-centric approach, focusing on usability, accessibility, and performance.
* Gather user feedback during the testing phase and make necessary improvements.
* Implement a robust customer support system post-launch to address any issues promptly.

**3. Market competition (Development, Testing, and Post-launch phases):**

* Identify unique selling points during the development phase.
* Monitor market trends and competitor activities throughout the project.
* Invest in marketing and promotional activities during the testing phase and continue them post-launch to increase brand awareness and attract new customers.

**4. Data security and privacy (Development, Testing, and Post-launch phases):**

* Implement strong security measures during the development phase, such as encryption, secure authentication, and access control mechanisms.
* Regularly update and patch the app during the testing phase to address any security vulnerabilities.
* Develop a comprehensive privacy policy and ensure compliance with relevant data protection regulations throughout the project.
* Educate users on the importance of data security and privacy post-launch and provide them with tools to manage their personal information.

**6. Prototyping:**

1. Define the app's core features and functionalities based on market research and client requirements.

2. Create a low-fidelity prototype, such as wireframes or mockups, to visualize the app's layout and user interface.

3. Develop a high-fidelity prototype that includes basic functionality and interactivity, allowing users to navigate through the app and test its features.

4. Gather user feedback on the prototype and use this feedback to refine the app's design and functionality.

5. Iterate on the prototype as needed, making improvements based on user feedback and testing results.

6. Once the prototype meets the desired level of quality and user satisfaction, proceed with full-scale development.

**9. CONFIGURATION AND VERSION CONTROL**

Version control manages different versions of configuration items that are created during the software engineering process.

It is the process of managing and tracking changes to code and documentation by all members of teams. The attributes for version control for all projects are that

* you can revert changes.
* You can collaborate to fix issues.
* You can solve conflicts.
* You can organize features.

Team members are allowed to log in at the same time and take a copy for work, thus saving time.   
Any document, model, or design related to the project are artifacts, which means they evolve to reflect changes in the project.   
As projects progress, the team will create many artefacts that are not static.   
Project artefacts need configuration version control applied to them.   
This is so changes can be tracked and information can be retrieved at any time for your project's needs or to share with other projects.

**10. TOOLS**



[This Photo](https://damiandeluca.com.ar/que-es-flutter) by Unknown Author is licensed under [CC BY-NC-ND](https://creativecommons.org/licenses/by-nc-nd/3.0/)

1- Flutter 2-Figma

3-Android studio  
4-Firebase

5 Dart 6- Visual Studio Code

**11. ARCHITECTURE**

Mobile and smart phone required to install our application.