# Data-Engineering project Documentation


**Group Members**

| | |
|---|---|
| Bassant Tarek | ID: 120200244 |
| Rewan Yehia | ID: 120200013 |
| Nancy El-sherbeny | ID: 120200247 |
| MennaAllah Ashraf | ID: 120190066 |
| Abdelrahman Khalifa | ID: 120200024 |


**To**
Dr. Mohamed Abbassy

**And**
Eng. Ahmed Hesham

# 1 Overview

Our team embarked on the development of a comprehensive Grocery Shop application that integrates database functionality with mobile technologies. The primary goal was to create a user-friendly platform that offers multiple features for managing shopping carts, browsing available groceries, and enhancing the shopping experience for our users.

# 2 Implementation Approach

For the Grocery Shop application, our team opted to create an Android mobile app using Ionic Angular for the frontend and PHP for the backend, adhering to RESTful API principles. To handle data storage, we utilized a MySQL database hosted on Clever Cloud, storing user credentials, shopping carts, and product details.

## 2.1 Deployment Strategy

- **Backend (RESTful API using PHP):** We deployed the PHP-based backend, housing the RESTful API, on an AWS EC2 instance. This instance facilitated the hosting and accessibility of the API endpoints.
- **Frontend (Android Mobile App):** The Android mobile app, developed using Ionic Angular, was packaged and deployed as an APK file. Users could download and install this APK file on their Android devices to access the Grocery Shop application.

## 2.2 Infrastructure Utilization

- **AWS EC2 Instance:** Primarily utilized for hosting the backend PHP application, ensuring the availability and accessibility of the RESTful API endpoints.
- **Clever Cloud (MySQL Database):** Leveraged to store and manage the data concerning user credentials, shopping carts, and product details, providing a reliable database solution.

## 2.3 Project Repository

The codebase for the Grocery Shop application is hosted on GitHub at the following repository:
**GitHub-Repository:** https://github.com/Data-Project-Team/grocery-app

# 3 Database Structure Overview

The database for our grocery app comprises multiple tables, each serving a specific purpose in organizing, storing, and managing various aspects of the application. These tables are designed to efficiently store data related to users, products, orders, categories, user interactions, and more.

Below is a brief overview of the main tables and their intended functionality within the grocery app:

- **app_cart:** Stores information related to user shopping carts including product quantities and timestamps.

- **app_ctg:** Contains categories for organizing products, along with image references and status indicators.
- **app_init:** Contains initial data and configurations important for the app's functioning.
- **app_orders:** Manages user orders, associating users with products and order timestamps.
- **app_products:** Stores comprehensive product details such as names, descriptions, prices, stock quantities, and user interactions like likes and views.
- **app_users:** Manages user profiles, including login information, status, and contact details.
- **app_wishlist:** Stores the wishlist items of users, linking users with products they wish to save.
- **app_addresses:** Stores the addresses of users, linking users with their addresses.
- **app_payments:** Stores the payment card information of users, linking users with their payment card information.

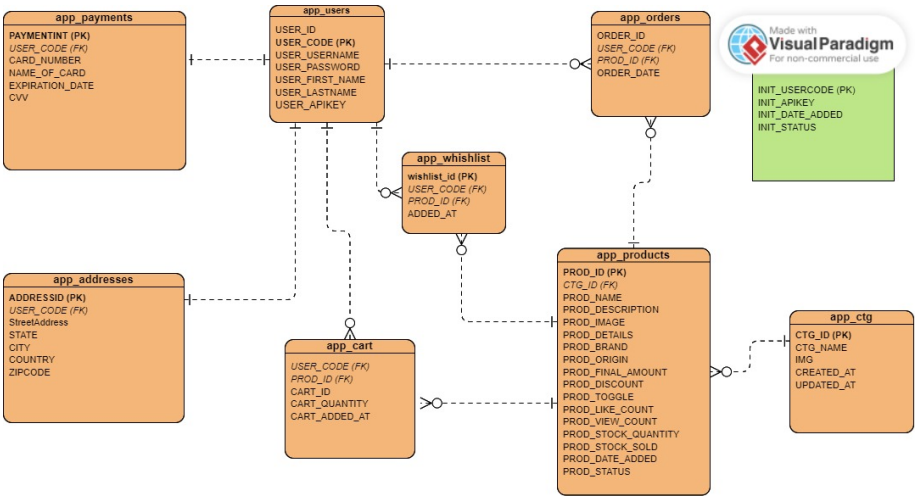


Figure 1: Database Schema

# 4 End-points of the application

The application endpoints are structured following RESTful API conventions. The primary endpoint for accessing the application backend is hosted at: **ec2-51-20-121-181.eu-north-1.compute.amazonaws.com/api.php**.

- addtocart.php (`/api.php?action=addtocart&usrCode={usercode}&prodId={productid}&apikey={apikey}`): Adds items to the user's shopping cart.
- changeaddress.php (`/api.php?action=changeaddress&apikey={apikey}&street={street}&city={city}&state={state}&country={country}&zip={zip}&userCode={userCode}`): Allows users to modify their address information.

- changepassword.php (/api.php?action=changepassword&apikey={apikey}&oldPassword={oldPassword}&newPassword={newPassword}&username={username}&userid={userid}): Enables users to change their password credentials.
- check_out.php (/api.php?action=check_out&apikey={apikey}&usrCode={usrCode}&prodId={prodId}): Facilitates the checkout process for the items in the user's cart.
- fetchcategory.php (/api.php?action=fetchcategory&apikey={yourapikey}): Retrieves categories of available groceries.
- fetchproducts.php (/api.php?action=fetchproducts&apikey={yourapikey}&categoryId={categoryId} &fetchby={fetchby}): Fetches the list of available products.
- fetchwishlist.php (/api.php?action=fetchwishlist&apikey={yourapikey}&usrCode={usrCode}): Retrieves the user's wishlist.
- filter.php (/api.php?action=filteroptions&apikey={yourapikey}&categoryId={categoryId}): Allows users to filter grocery items based on specific criteria.
- filteroptions.php (/api.php?action=filteroptions&apikey={yourapikey}&categoryId={categoryId}): Retrieves available options for filtering.
- getusercart.php (/api.php?action=getusercart&apikey={yourapikey}usrCode={userCode}): Fetches the contents of the user's shopping cart.
- getuserinfo.php (/api.php?action=getuserinfo&apikey={yourapikey}usrCode={userCode}): Retrieves user information.
- initapps.php (/api.php?action=initapps): Initializes the application.
- like.php (/api.php?action=like): Allows users to like or rate products.
- loginuser.php (/api.php?action=loginuser&apikey={yourapikey}&usrCode={userCode}&pwd={pwd}&usename={username}): Handles user login functionality.
- logout.php (/api.php?action=logout&apikey={yourapikey}): Logs the user out of the application.
- oauthuser.php (/api.php?action=oauthuser): Handles user authentication via OAuth.
- paymentmethod.php (/api.php?action=paymentmethod&apikey={yourapikey}&cardnumber={cardnumber}&nameoncard={nameoncard}&expirationdate={expirationdate}&cvv={cvv}&userCode={userCode}): Manages user's preferred payment methods.
- purchasedata.php (/api.php?action=purchasedata&apikey={yourapikey}&prodId={prodId}): Retrieves data related to user purchases.
- registeruser.php (/api.php?action=registeruser&apikey={yourapikey}&email={email}&pwd={pwd}&name={name}): Registers new users within the application.
- removefromcart.php (/api.php?action=removefromcart&apikey={yourapikey}&usrCode={usrCode}&prodId={prodId}): Removes items from the user's shopping cart.

- search.php ( `/api.php?action=search&apikey={yourapikey}&term={term}&min-price={min-price}&max-price={max-price}`): Allows users to perform searches for specific products.
- updatecart.php(`/api.php?action=updatecart&usrCode={usrCode}&prodId={prodId}&quantity={quantity}&apikey={yourapikey}`): Updates the contents of the user's shopping cart.
- upload.php (/api.php?action=upload): Handles file uploads within the application.

# 5 Class Work Division

Our team collaborated on various aspects of the grocery app project, delineating specific responsibilities to ensure a comprehensive and well-executed outcome. The division of work among team members is as follows:

## 5.1 Database Design

### 5.1.1 Team Members

The entire team, including Bassant, Rewan, Menna, Nancy,and Abdelrahman collaborated extensively on designing the comprehensive database schema for the grocery app. Their collective expertise and concerted efforts resulted in the creation of a structured and efficient database that serves as the backbone of the application.

The database schema, meticulously crafted by the team, encompasses various tables serving distinct purposes within the grocery app. These tables, along with their creators, are detailed below:

- 'appcart': Created by (Nancy-Abdelrahman), Stores information related to user shopping carts.
- 'appctg': Created by (Bassant), Contains categories for organizing products within the app.
- 'apporders':Created by (Bassant - Rewan), Manages user orders and their details.
- 'appproducts' :Created by (Nancy-Abdelrahman), Stores product details including names, descriptions, prices, etc.
- 'appusers' : Created by (Bassant - Rewan), Contains user-related information such as usernames, passwords, etc.
- 'appinit': Created by (Rewan), holds initial data or configurations crucial for app functionality.
- 'appaddresses': Created by (Menna), Stores information related to user addresses.
- 'apppayments':Created by (Menna), Stores information related to user payment-methods.

## 5.2 Data Entry into Database

### 5.2.1 Bassant and Rewan

Both Bassant and Rewan collaborated closely on populating the 'appproducts' table within the database.

They shared responsibilities in performing web scraping from Carrefour's website, extracting essential product information like names, prices, and basic details. This collected data was meticulously integrated into the 'appproducts' table. Then, Bassant involved in the manual entry of additional product details and descriptions. Their combined efforts resulted in a meticulously designed schema populated with comprehensive product data obtained from web scraping and manual entry, providing users with a rich and informative grocery app experience.

## 5.3 Rewan's Contributions

### 5.3.1 API Services Initialization

- **API Services Setup**:
  - Initialize all API services required for the app functionalities.

### 5.3.2 Register Page

- **Frontend**:
  - Created register page UI using HTML,and TypeScript.
- **Backend**:
  - Implemented registeruser.php for user registration handling.
  - **API Endpoint**: Created '/api/register' for user registration.

### 5.3.3 Details Page

- **Frontend**:
  - Developed UI layout using HTML, and CSS for displaying product details.

### 5.3.4 Product Page

- **Frontend**:
  - Created UI using HTML, CSS, and TypeScript for displaying various product categories.

### 5.3.5 Routing

- **App Routing**:
  - Configured routing within the app for seamless navigation between pages.

#### 5.3.6 Cart Functionality Backend APIs

- **Cart Management APIs**:
  - Developed backend API services for cart functionalities:
    - '/api/addtocart': Adds products to the cart.
    - '/api/getusercart': Retrieves the user's cart information.
    - '/api/updatecart': Updates items in the user's cart.
    - '/api/removefromcart': Removes items from the cart.

#### 5.3.7 Purchase Data Display

- **Display Purchase Data**:
  - Implemented functionality to show the number of people who purchased a product in the last 24 hours on the details page.

#### 5.3.8 Integration of Egyptian Products in Homepage

- **Frontend Integration**:
  - Utilized and integrated modified fetchproduct.php to display Egyptian-origin products (100%) in the homepage based on the origin information fetched for each product.

### 5.4 Bassant's Contributions

#### 5.4.1 Login Page

- **Frontend**:
  - Developed login page UI using HTML, CSS, and TypeScript.
- **Backend**:
  - Implemented loginuser.php for backend authentication handling.
  - API Endpoint: Created '/api/login' for user authentication.

#### 5.4.2 Register Page

- **Frontend (CSS)**:
  - Developed CSS styling for the register page UI.

#### 5.4.3 Image Upload to Database

- **Backend**:
  - Implemented upload.php to associate images with respective category-ids in the database.
  - API Endpoint: Created '/api/upload' for image uploads.

#### 5.4.4 Category Page

- **Frontend**:
  - Designed category page UI using HTML, CSS, and TypeScript.
- **Backend**:
  - Developed fetchcategory.php for retrieving category data.
  - API Endpoint: Created '/api/fetchcategory' to fetch category data.

#### 5.4.5 Checkout Functionality

- **Backend**:
  - Implemented checkout.php for processing user purchases.
  - API Endpoint: Created '/api/checkout' for handling user purchases.

#### 5.4.6 Purchase Data Display

- **Frontend**:
  - Developed UI layout using CSS for displaying purchase data.
- **Backend**:
  - Developed purchasedata.php backend functionality to retrieve and display the number of people who purchased each product.
  - API Endpoint: Created '/api/purchasedata' to retrieve purchase data.

#### 5.4.7 Details Page - Total Price Calculation Function

- **Frontend (TypeScript)**:
  - Modified and implemented the function responsible for dynamically calculating the total price when increasing the quantity of a product.
  - Developed TypeScript logic to update the total price based on the selected quantity.

### 5.5 Nancy's Contributions

#### 5.5.1 Product Card Component

- **Frontend (HTML, CSS, TypeScript)**:
  - Enhanced the product-card component's appearance and styling using HTML, CSS, and TypeScript.
  - Implemented the discount feature for products within the product-card interface.

#### 5.5.2 Liked Component for Products

- **Frontend**:
  - Created HTML, CSS, and TypeScript files for the liked component embedded within each product.
  - Designed the visual representation and behavior of the liked feature.
- **Backend**:
  - Developed backend functionality for the liked component in the `like.php` file.
  - API Endpoint: Created '/api/like' to handle the liked feature.

**Cart-Button Development**

- **Frontend (CSS)**:
  - Implemented CSS styling for the cart button.

#### 5.5.3 Cart Services

- **Services**:
  - Developed functionalities related to the cart within the app in the dedicated cart services file.

#### 5.5.4 Details Page Update with Discount Feature

- **Frontend (CSS, TypeScript)**:
  - Updated CSS for the details page UI.
  - Implemented the discount feature for each product on the details page.
  - Established the connection between frontend and backend in the TypeScript file for the details page.

### 5.6 Menna's Contributions

#### 5.6.1 Search Component

- **Frontend (HTML, CSS, TypeScript)**:
  - Developed the search component's UI using HTML, CSS, and TypeScript and devleop the UI layout of the search component on the home page
- **Backend**:
  - Implemented backend functionality for handling search requests and responses in `search.php` file.
  - Created the necessary API Endpoint: '/api/search' to handle search requests and responses.
- **Services**:
  - Developed essential services related to the search functionality within the application to connect between the frontend and the backend.

#### 5.6.2 Wishlist Page

- **Frontend (HTML, CSS, TypeScript)**:
  - Develop the UI of the Wishlist page using HTML , CSS and Type Script and Ensured an engaging and user-friendly interface for the wishlist functionality.
- **Backend**:
  - Implemented the backend functionality in the `fetchwishlist.php` file to fetch wishlist data.
  - Created the API Endpoint: '/api/fetchwishlist' to retrieve wishlist data from the backend.

#### 5.6.3 Account Page

- **Frontend (HTML, CSS, TypeScript)**:
  - Designed and developed the account page frontend depending on the login user status.
  - Designed and developed the frontend for the following modals: Address-change , Payment-method-change and Settings and then integrate them into the account page.
- **Backend**:
  - Implemented backend functionality in `paymentmethod.php` and `changeaddress.php` and `changepassword.php` to handle payment method changes, address updates and password change respectively.
  - Created API Endpoints: '/api/paymentmethod' , '/api/changeaddress' and '/api/changepassword'to manage payment method and address changes from the backend.
  - Developed `getuserinfo.php` to retrieve user information for display on the account page.
  - Created API Endpoints:'/api/getuserinfo' to manage relevant actions from the backend.

#### 5.6.4 Log-Out

- **Backend**:
  - Implemented the logout functionality in `logout.php` and to handle user logout
  - Created API Endpoints: '/api/logout'.

### 5.7 Abdelrahman's Contributions

#### 5.7.1 Homepage Development

- **Frontend (HTML, CSS, TypeScript)**:
  - Developed the frontend of the entire homepage, ensuring a cohesive and engaging user interface.
  - Designed and implemented various sections of the homepage, including Trending, Sale, and other relevant sections.

#### 5.7.2 Cart-Button Development

- **Frontend (HTML,TypeScript)**:
  - Created the cart-btn functionality.
  - Developed HTML,and TypeScript for the cart button.

#### 5.7.3 Backend Development

- **fetchproduct.php**:
  - Implemented backend functionality in the `fetchproduct.php` file to fetch product data.
  - Created the necessary API Endpoint: '/api/fetchproduct' to retrieve product data from the backend.

#### 5.7.4 Product Page - Filter Component

- **Frontend (HTML, CSS, TypeScript)**:
  - Developed the frontend of the filter component on the product page using HTML, CSS, and TypeScript.
- **Backend**:
  - Implemented backend functionality in the `filter.php` file to manage product filtering.
  - Created the API Endpoint: '/api/filter' to handle filter requests and responses.

#### 5.7.5 Toggle Component

- **Frontend (HTML, CSS, TypeScript)**:
  - Designed and implemented the toggle component, providing a user-friendly and interactive interface element.

# 6 Running the Application

1. Download APK from GitHub: The APK file for the mobile application is provided in the GitHub repository's README file.
   - Locate the README file in the root directory of the repository.
   - Find the section mentioning the APK file.
   - Click on the provided download link to obtain the APK file.
2. Install APK on Android Device:
   - Transfer the downloaded APK file to an Android device.
   - Ensure installation from unknown sources is allowed in the device settings if necessary.
   - Tap on the downloaded APK file to begin the installation process.
   - Follow the on-screen instructions to complete the installation.

3. Launch the Application:

    - Once installed, locate the app icon on your device's home screen.
    - Tap on the app icon to launch the Grocery Shop Application on your Android device.