

**Project name:**

Plant Nursery Management System.

**Project group name:**

Ghorsa project.

**Group name:**

Tamara Tumeh(Shiren Hijazi).

Tojan Naiem(Zaina AlSaed).

Ruba Alhaj(Shiren Hijazi).

* **Project Description:**

Ghorsa is an e-commerce platform specializing in selling plants and gardening products. It offers users an organized shopping experience, starting with a categorized catalog of plants and extending to a seamless checkout process. Users can register, manage multiple delivery addresses, and browse products with details like name, price, description, and stock availability. The platform allows users to add items to their shopping cart, mark products as favorites, and place orders that track their status from "Pending" to "Completed." Payments are securely processed and linked to each order. Users can add multiple addresses to their profiles for flexible delivery options. The website showcases trending products and the latest arrivals, providing a dynamic shopping experience. In addition to plants, Ghorsa also offers a variety of gardening tools and supplies, ensuring users have access to everything they need for their gardening endeavors.

**Category Table**

* Stores information about product categories.
* **Fields**:
  + category\_id: Primary key, unique identifier for each category.
  + name: Name of the category.
  + image: Image associated with the category.

**Product Table**

* Represents the products available in the system.
* **Fields**:
  + product\_id: Primary key, unique identifier for each product.
  + name: Name of the product.
  + price: Price of the product.
  + description: Description of the product.
  + pot\_color: Color of the pot for plant-related products.
  + plant\_care: Plant care instructions.
  + stock: Quantity available in inventory.
  + category\_id: Foreign key linking to the category table.

**User Table**

* Represents users of the system.
* **Fields**:
  + user\_id: Primary key, unique identifier for each user.
  + First\_Name: First name of the user.
  + Last\_Name: Last name of the user.
  + email: Email address of the user.
  + password: User's password for authentication.
  + Phone: Contact phone number.

**Address Table**

* Stores addresses associated with users.
* **Fields**:
  + address\_id: Primary key, unique identifier for each address.
  + city: City of the address.
  + country: Country of the address.
  + zipCode: Postal code of the address.
  + user\_id: Foreign key linking to the user table.

**Order Table**

* Tracks orders placed by users.
* **Fields**:
  + order\_id: Primary key, unique identifier for each order.
  + OrderDate: Date the order was placed.
  + status: Current status of the order (e.g., pending, shipped).
  + order\_amount: Total amount for the order.
  + user\_id: Foreign key linking to the user table.

**Payment Table**

* Manages payment details for orders.
* **Fields**:
  + payment\_id: Primary key, unique identifier for each payment.
  + amount: Payment amount.
  + payment\_method: Method of payment (e.g., credit card, PayPal).
  + order\_id: Foreign key linking to the order table.

**Cart Table**

* Manages shopping carts for users.
* **Fields**:
  + cart\_id: Primary key, unique identifier for each cart.
  + tracking\_id: Tracking information for the cart.
  + user\_id: Foreign key linking to the user table.

**User\_Favorites Table**

* Allows users to save their favorite items.
* **Fields**:
  + favorite\_id: Primary key, unique identifier for each favorite item.
  + created\_at: Timestamp when the favorite item was added.
  + user\_id: Foreign key linking to the user table.

**Role Table**

* Defines roles assigned to users.
* **Fields**:
  + id: Primary key, unique identifier for each role.
  + role\_name: Name of the role (e.g., admin, customer).
  + user\_id: Foreign key linking to the user table.

Each table is designed to efficiently manage specific data and is linked to others using foreign key relationships, ensuring data consistency and normalization.

* **Project scenario:**

A user registers on the system by providing their First\_Name, Last\_Name, email, password, and Phone number, which are stored in the user table under a unique user\_id. The user adds multiple addresses, such as their home and office addresses, with details like city, country, and pin\_Code, stored in the address table under a unique address\_id.

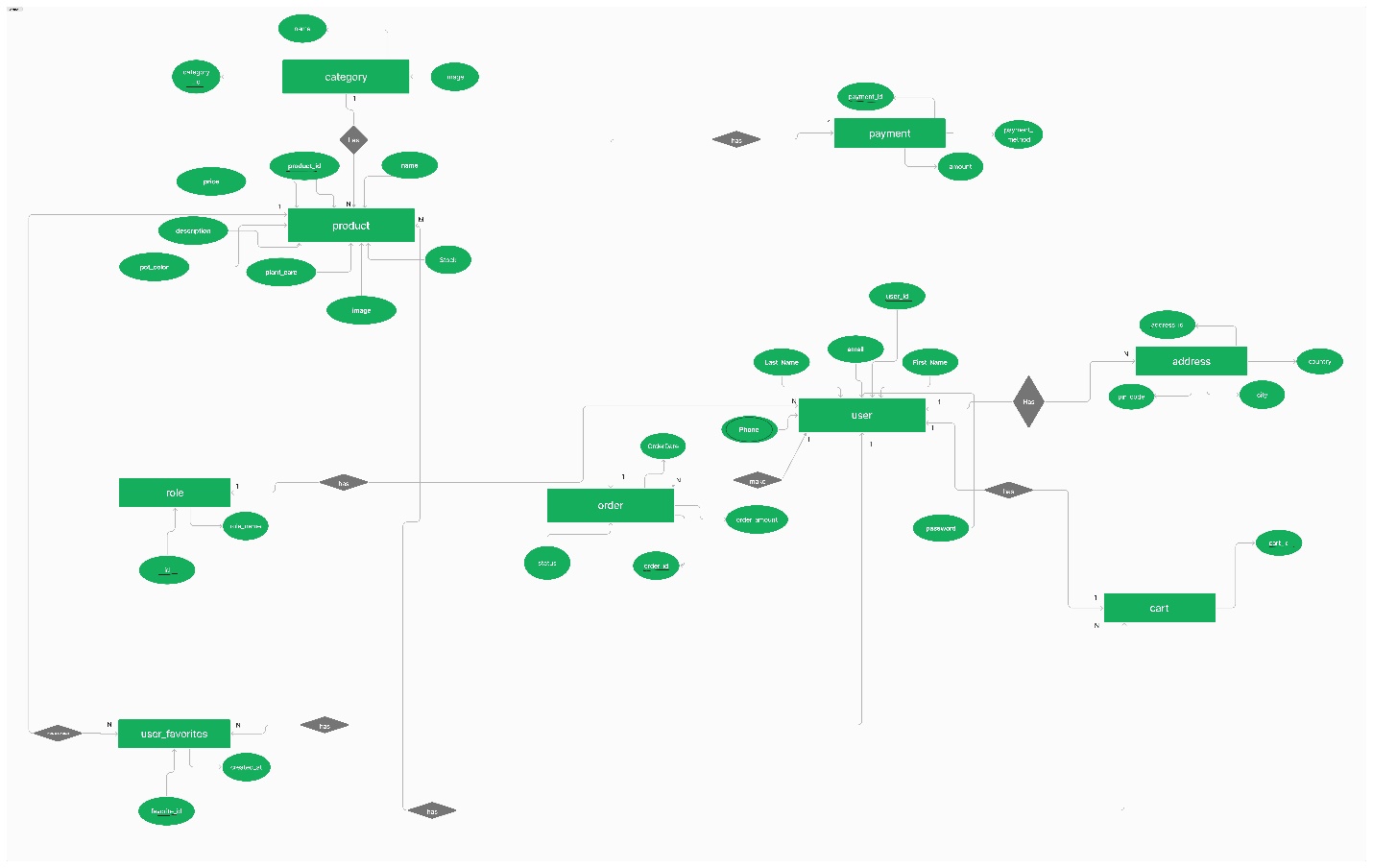
The user browses products in various categories such as "Indoor Plants" or "Outdoor Plants", managed in the category table.category are stored in the category table with attributes like name, image. Each category is identified by a category\_id, and its products are stored in the product table with attributes like product\_id, name, price, description, pot\_color, plant\_care,image, and stock.

The user adds products to their cart, represented in the cart table with a unique cart\_id. The cart may contain multiple products, linked to the user table via user\_id for tracking. Once the user places an order, the details are stored in the order table with a unique order\_id, including the OrderDate, status, and order\_amount. The order links to the user and their cart.

The user makes a payment, and the details, such as payment\_id, amount, and payment\_method, are stored in the payment table, linked to the order table. Additionally, the user can save favorite products in the user\_favorites table , with attributes like created\_at, and a unique favorite\_id, enabling them to quickly revisit their preferred items.

Finally, the role table manages user roles, assigning roles such as "Admin" or "Customer" to users using role\_name and a foreign key user\_id. This structure ensures proper role-based access control within the system.

* **"Entity-Relationship Diagram" (ERD)**



**Project code**

CREATE DATABASE ghorsa;

USE ghorsa;

CREATE TABLE category(

category\_id INT PRIMARY KEY AUTO\_INCREMENT,

image VARCHAR(100) NOT NULL,

name VARCHAR(30) NOT NULL

);

CREATE TABLE product(

product\_id INT PRIMARY KEY AUTO\_INCREMENT,

name VARCHAR(50) NOT NULL,

description VARCHAR(500),

plant\_care VARCHAR(300),

image VARCHAR(100) NOT NULL,

pot\_color VARCHAR(50),

price DECIMAL(10,2) NOT NULL,

stock INT DEFAULT 0 CHECK(stock >= 0),

category\_id INT,

FOREIGN KEY (category\_id) REFERENCES category(category\_id)

);

CREATE TABLE roles (

id INT PRIMARY KEY AUTO\_INCREMENT,

role\_name VARCHAR(50) NOT NULL UNIQUE

);

CREATE TABLE user (

user\_id INT PRIMARY KEY AUTO\_INCREMENT,

name VARCHAR(100) NOT NULL,

email VARCHAR(100) NOT NULL UNIQUE,

password VARCHAR(255) NOT NULL,

phone varchar(15),

role\_id INT,

FOREIGN KEY (role\_id) REFERENCES roles(id)

);

CREATE TABLE address(

address\_id INT PRIMARY KEY AUTO\_INCREMENT,

country VARCHAR(20) NOT NULL,

city VARCHAR(20) NOT NULL,

street VARCHAR(100),

pin\_code INT NOT NULL,

user\_id INT,

FOREIGN KEY (user\_id) REFERENCES user(user\_id)

);

CREATE TABLE order\_table(

order\_id INT PRIMARY KEY AUTO\_INCREMENT,

order\_amount DECIMAL(10,2) NOT NULL,

order\_date DATETIME DEFAULT CURRENT\_TIMESTAMP,

status VARCHAR(20) DEFAULT 'Pending' CHECK(status IN ('Pending', 'Completed', 'Shipped')),

user\_id INT,

FOREIGN KEY (user\_id) REFERENCES user(user\_id)

);

CREATE TABLE payment(

payment\_id INT PRIMARY KEY AUTO\_INCREMENT,

payment\_method VARCHAR(50) DEFAULT 'Cash on Delivery',

amount DECIMAL(10,2) NOT NULL,

order\_id INT,

FOREIGN KEY (order\_id) REFERENCES order\_table(order\_id)

);

CREATE TABLE cart(

cart\_id INT PRIMARY KEY AUTO\_INCREMENT,

user\_id INT,

FOREIGN KEY (user\_id) REFERENCES user(user\_id)

);

CREATE TABLE cart\_item(

cart\_item\_id INT PRIMARY KEY AUTO\_INCREMENT,

cart\_id INT,

price DECIMAL(10,2),

product\_id INT,

quantity INT NOT NULL DEFAULT 0 CHECK(quantity >= 0),

FOREIGN KEY (cart\_id) REFERENCES cart(cart\_id),

FOREIGN KEY (product\_id) REFERENCES product(product\_id)

);

CREATE TABLE user\_favorites (

favorite\_id INT PRIMARY KEY AUTO\_INCREMENT,

user\_id INT NOT NULL,

product\_id INT NOT NULL,

created\_at DATETIME DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (user\_id) REFERENCES user(user\_id),

FOREIGN KEY (product\_id) REFERENCES product(product\_id),

UNIQUE (user\_id, product\_id)

);