

Phase 1 Acknowledged of what it need to be done

Designing a database for a food oasis application with a simple GPS-based interface to find fresh food sources is an interesting project. Here's a high-level overview of the database structure and resources you might need to implement this:

Database Schema:

Users Table:

- UserID (Primary Key)
- Username
- AcceptedPolicies (Boolean, indicating whether they have accepted the policies)
- Email
- Registration Date
- GPSSAccess (Boolean, indicating whether they have granted GPS access)

Relationships:

- One-to-Many relationship with Session (a user can have multiple sessions)
- Many-to-Many relationship with Food Source through User_Favorite (users can have multiple favorite food sources)

Food Sources Table:

- SourceID (Primary Key)
- SupplierID (Foreign Key)
- Food Source Name
- Food Description
- Availability Schedule (days and hours)
- GPS Location (coordinates of the food source)

Relationships:

- Many-to-One relationship with Supplier (a food source is supplied by one supplier)
- Many-to-Many relationship with User through User_Favorite (multiple users can have the same favorite food source)

Suppliers Table:

- SupplierID (Primary Key)
- Supplier Name
- Location (GPS coordinates)
- Contact Information (phone, email, etc.)
- Registration Date

Relationships:

- One-to-Many relationship with Food Source (a supplier can have multiple food sources)
-

User Favorites Table (Optional for registered users):

- UserID (Foreign Key)
- SourceID (Foreign Key)

Relationships:

- Many-to-One relationship with User (maps users to their favorite food sources)
- Many-to-One relationship with Food Source (maps food sources to users who favorited them)

Sessions Table (For managing active sessions and GPS access):

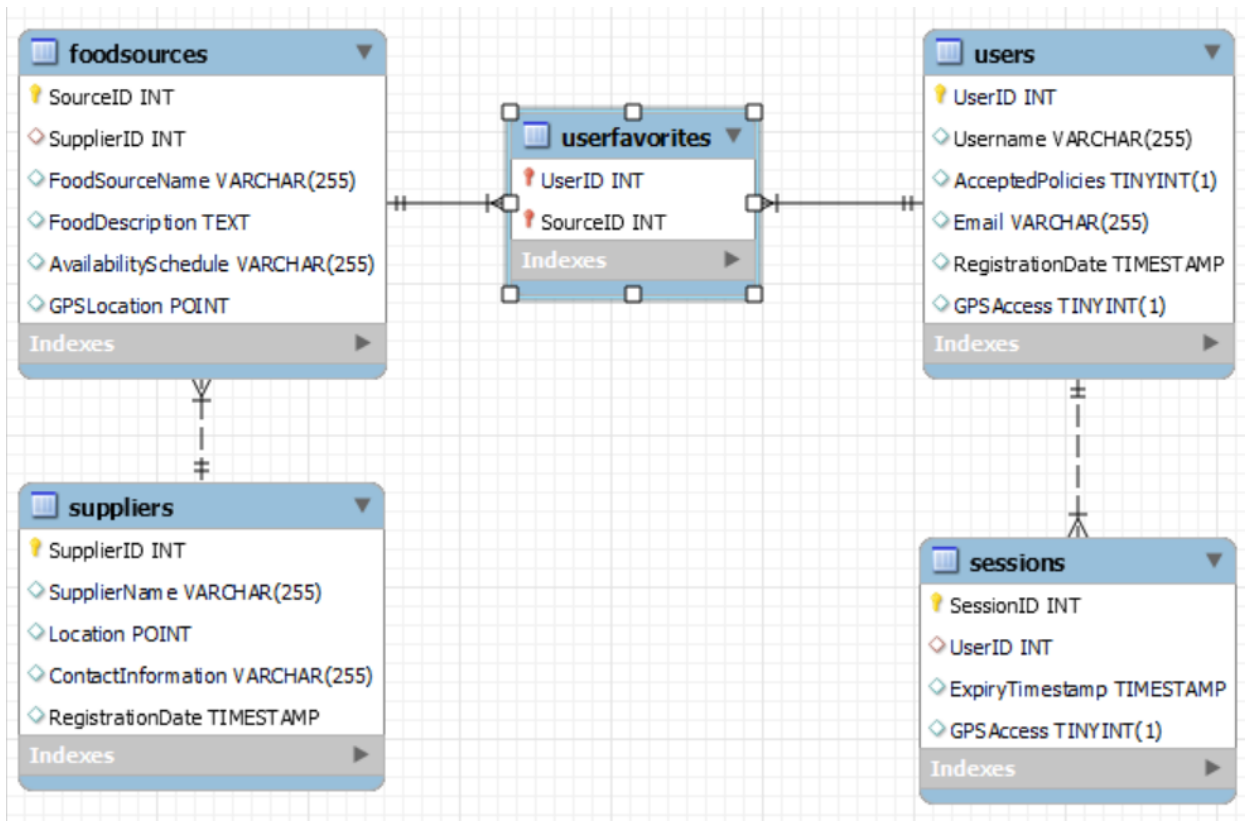
- SessionID (Primary Key)
- UserID (Foreign Key)
- Expiry Timestamp
- GPS Access (boolean flag)

Relationships:

- Many-to-One relationship with User (a session belongs to one user)

Phase 2 Creation of the Entities Relational Diagrams (ERD) & Entities Relational Model (ERM)

Entities Relational Model (ERM)



Database Schema:

Users Table:

- UserID (Primary Key)
- Username
- AcceptedPolicies (Boolean, indicating whether they have accepted the policies)
- Email

- Registration Date
- GPSTime (Boolean, indicating whether they have granted GPS access)

Relationships:

- One-to-Many relationship with Session (a user can have multiple sessions)
- Many-to-Many relationship with Food Source through User_Favorite (users can have multiple favorite food sources)

Food Sources Table:

- SourceID (Primary Key)
- SupplierID (Foreign Key)
- Food Source Name
- Food Description
- Availability Schedule (days and hours)
- GPS Location (coordinates of the food source)

Relationships:

- Many-to-One relationship with Supplier (a food source is supplied by one supplier)
- Many-to-Many relationship with User through User_Favorite (multiple users can have the same favorite food source)

Suppliers Table:

- SupplierID (Primary Key)
- Supplier Name
- Location (GPS coordinates)
- Contact Information (phone, email, etc.)
- Registration Date

Relationships:

- One-to-Many relationship with Food Source (a supplier can have multiple food sources)
-

User Favorites Table (Optional for registered users):

- UserID (Foreign Key)
- SourceID (Foreign Key)

Relationships:

- Many-to-One relationship with User (maps users to their favorite food sources)
- Many-to-One relationship with Food Source (maps food sources to users who favorited them)

Sessions Table (For managing active sessions and GPS access):

- SessionID (Primary Key)
- UserID (Foreign Key)
- Expiry Timestamp
- GPS Access (boolean flag)

Relationships:

- Many-to-One relationship with User (a session belongs to one user)

Code to create the schema with the tables and relationship

-- Create Users Table

CREATE TABLE Users (

UserID INT AUTO_INCREMENT PRIMARY KEY,

```

    Username VARCHAR(255),
    AcceptedPolicies BOOLEAN,
    Email VARCHAR(255),
    RegistrationDate TIMESTAMP,
    GPSSAccess BOOLEAN
);

-- Create Food Sources Table
CREATE TABLE FoodSources (
    SourceID INT AUTO_INCREMENT PRIMARY KEY,
    SupplierID INT,
    FoodSourceName VARCHAR(255),
    FoodDescription TEXT,
    AvailabilitySchedule VARCHAR(255),
    GPSLocation POINT,
    FOREIGN KEY (SupplierID) REFERENCES Suppliers(SupplierID)
);

-- Create Suppliers Table
CREATE TABLE Suppliers (
    SupplierID INT AUTO_INCREMENT PRIMARY KEY,
    SupplierName VARCHAR(255),
    Location POINT,
    ContactInformation VARCHAR(255),
    RegistrationDate TIMESTAMP
);

-- Create User Favorites Table
CREATE TABLE UserFavorites (
    UserID INT,
    SourceID INT,
    FOREIGN KEY (UserID) REFERENCES Users(UserID),
    FOREIGN KEY (SourceID) REFERENCES FoodSources(SourceID),
    PRIMARY KEY (UserID, SourceID)
);

-- Create Sessions Table
CREATE TABLE Sessions (
    SessionID INT AUTO_INCREMENT PRIMARY KEY,
    UserID INT,
    ExpiryTimestamp TIMESTAMP,
    GPSSAccess BOOLEAN,
    FOREIGN KEY (UserID) REFERENCES Users(UserID)
);

```

Phase 3 Backend Framework: Go to file Project Structure.doc

- Use a backend framework like Flask (Python), Express (Node.js), or Ruby on Rails (Ruby) to create API endpoints for your application.
 - ● Javascript (NodeJS)
 - NodeJS is an open source server
 - environment that will allow the use of
 - Javascript code outside the browser
 - ● Package Manager
 - npm (node package manager)
 - ■ Used to install and manage
 - packages
 - ■ This is bundled with Node.js (comes with Node.js when installing
 - Node.js)
 - ● MySQL (relational database)
-