CS Professional Track 2 - 6531

WMR Trucking Services – Delivery Request System

Submitted by:

Joemire Dave Loremas

Andre John Llerin

Earl Lawrence Obguia

Submitted To:

Prof. Modesto C. Tarrazona

Date Submitted:

May 16, 2025

PROJECT OBJECTIVES

The objective of this project is to develop a dynamic and interactive web application for WMR Trucking Services that streamlines cargo delivery requests and logistics processes. The system will be built using PHP as the primary server-side scripting language, along with HTML, CSS, and JavaScript for a responsive and user-friendly interface. It aims to enable efficient client-server communication for handling delivery requests, estimating cargo weight or volume, and updating records in real-time. Additionally, the platform will be designed to be scalable and reliable to support the growing operational needs of the company.

PROJECT DESCRIPTION

The WMR Trucking Services website is made to help customers easily request a delivery. On the site, users can enter their name, business info, pick-up and delivery addresses, and cargo details like weight or number of boxes. They can also choose the dates for pick-up and delivery. This project was created using PHP, HTML, CSS, and JavaScript to make the process faster and more convenient.

TECHNOLOGY STACK

Front-End: **HTML, CSS**
Back-End: **PHP, Javascript**
Database: **MySQL**
Other Tools: **Github, Visual Studio Code, Figma**

SYSTEM FEATURES AND FUNCTIONAL REQUIREMENTS

1. Homepage

   Features:
   - Company branding and visuals
   - Promotional Tagline
   - Highlights of services:
     - Fast and Reliable Deliveries
     - Easy Online Booking

- Real - Time Updates

2. About Us Section

   Features:
   - Company journey and history
   - Vision and mission statements
   - Static content displaying the company background
   - Allow public access to company values and goals

3. Contact Us Section

   Features:
   - Contact Form (First Name, Last Name, Phone Number, Email)
   - Google Maps integration showing the company location
   - Form Submission (Send Inquiries)

4. Request Delivery Page

   Features:
   - Product Description: Input for items to be transported (e.g., sacks of rice, construction tools)
   - Estimated Quantity
     - Estimated Boxes
     - Estimated Weight

   - Pick-Up Address: Location of cargo origin
   - Delivery Address: Full destination address
   - Pick-Up Date: Preferred cargo collection date
   - Estimated Cargo Arrival Date (Client Wish): Optional preferred delivery date

5. Delivery Status Page

   Features:
   - Driver and Assistant Information
     - Display the driver's and assistant's names
     - Show contact number
     - Show the truck's plate number

- Cargo Status Updates
    - *Pending*
    - *For Pick-Up*
    - *In Transit*
    - *Delayed*
    - *Arrived*
    - Clearly visible status badge per transaction
    - Color-coded for better visibility (e.g., Red for Delayed and Green for Arrived)

- Current Location Tracking
    - Displays the latest known location of the cargo
    - Updated by Admin or Assistant in real time

- Transaction & Cargo Details
    - Show all request form details submitted by the client from request delivery page

- Departure and Arrival Info
    - Actual Departure Date & Time
    - Expected Arrival Date & Time (editable by Admin/Assistant)

- List of Transactions
    - Table or list view showing all delivery requests
    - Filters for status (e.g., show only "In Transit" or "Pending")
    - Clickable rows to view detailed tracking page

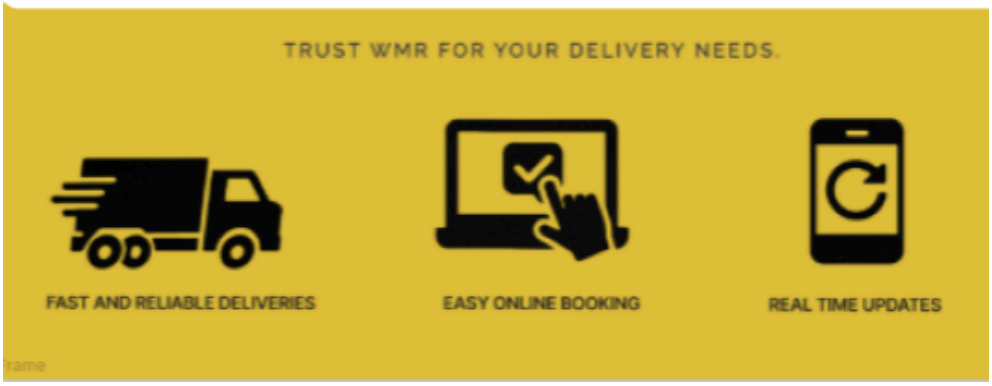
6. User Profile & Settings

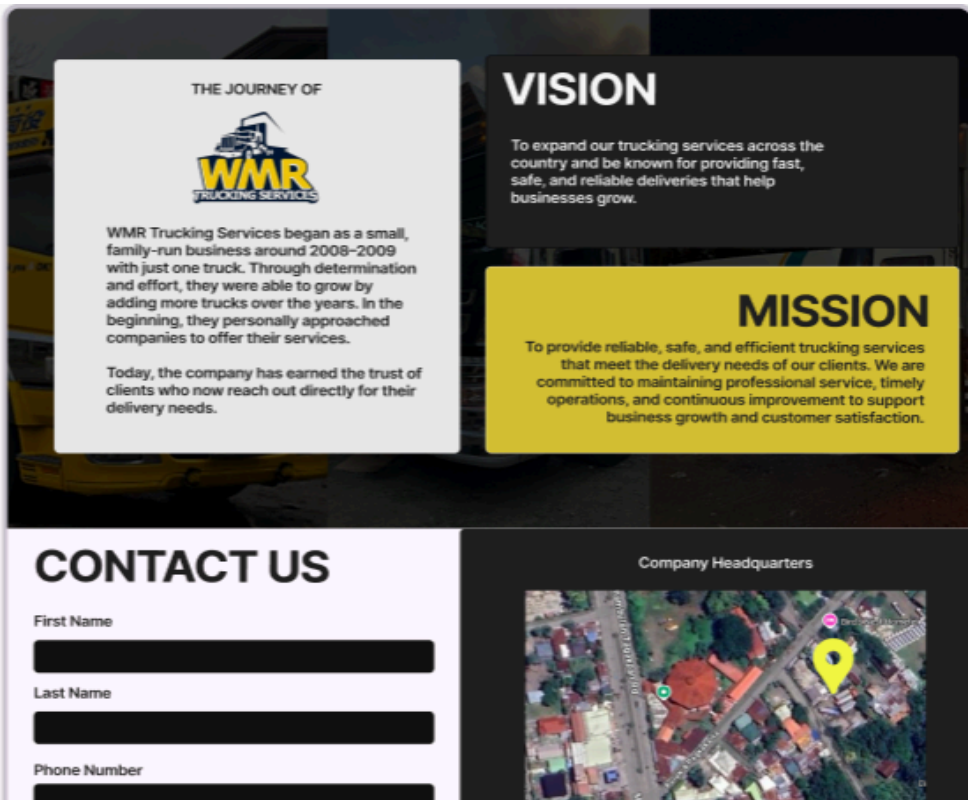
   Features:
   - View and edit user profile details
   - Upload/change profile picture
   - Update name, address, phone number, email address, nickname, etc.
   - Logout functionality

WIREFRAMES/UI MOCKUPS

We design prototypes in Figma.

HOMEPAGE





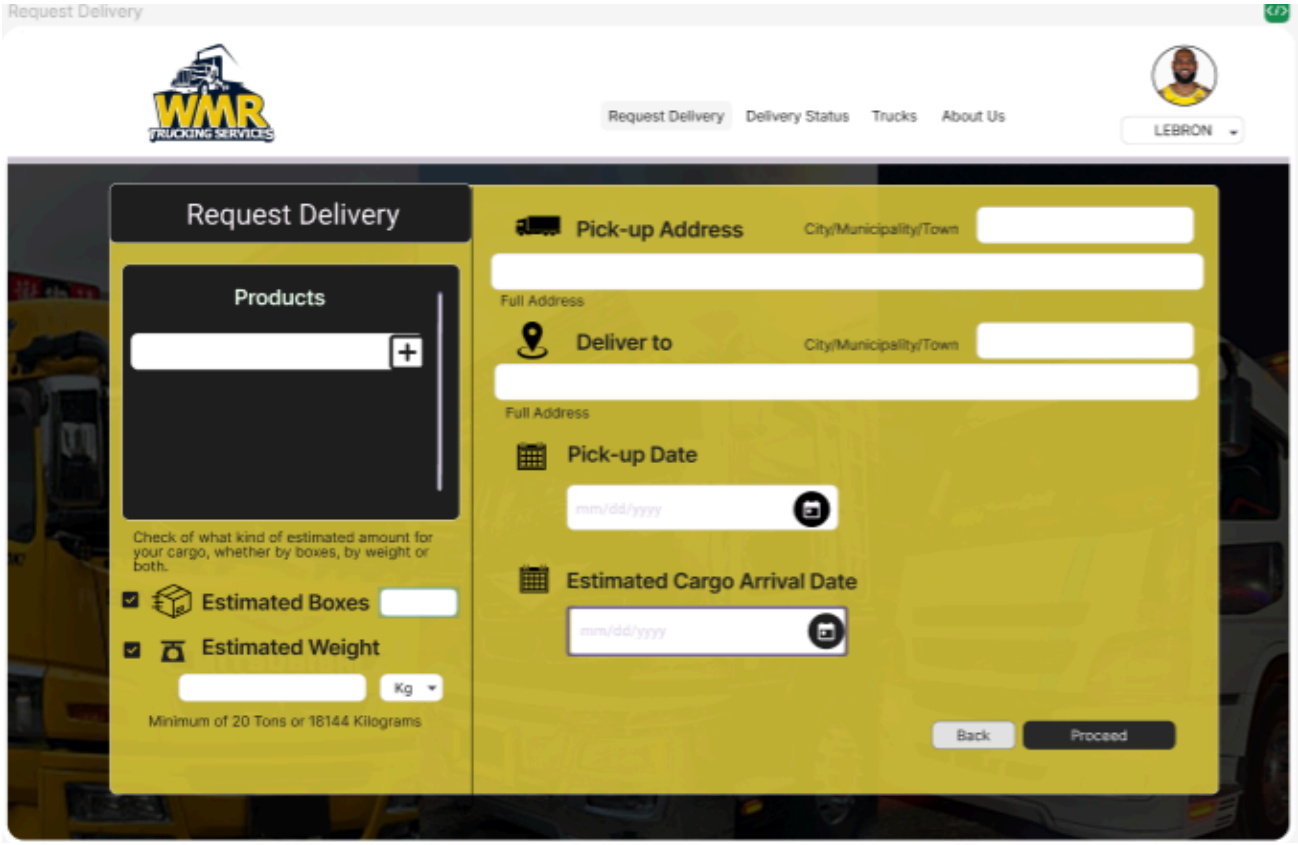
WMR Trucking Services

Purok 8 Pagkakaisa,
Barangay Lubogan, Toril,
Davao City, Philippines

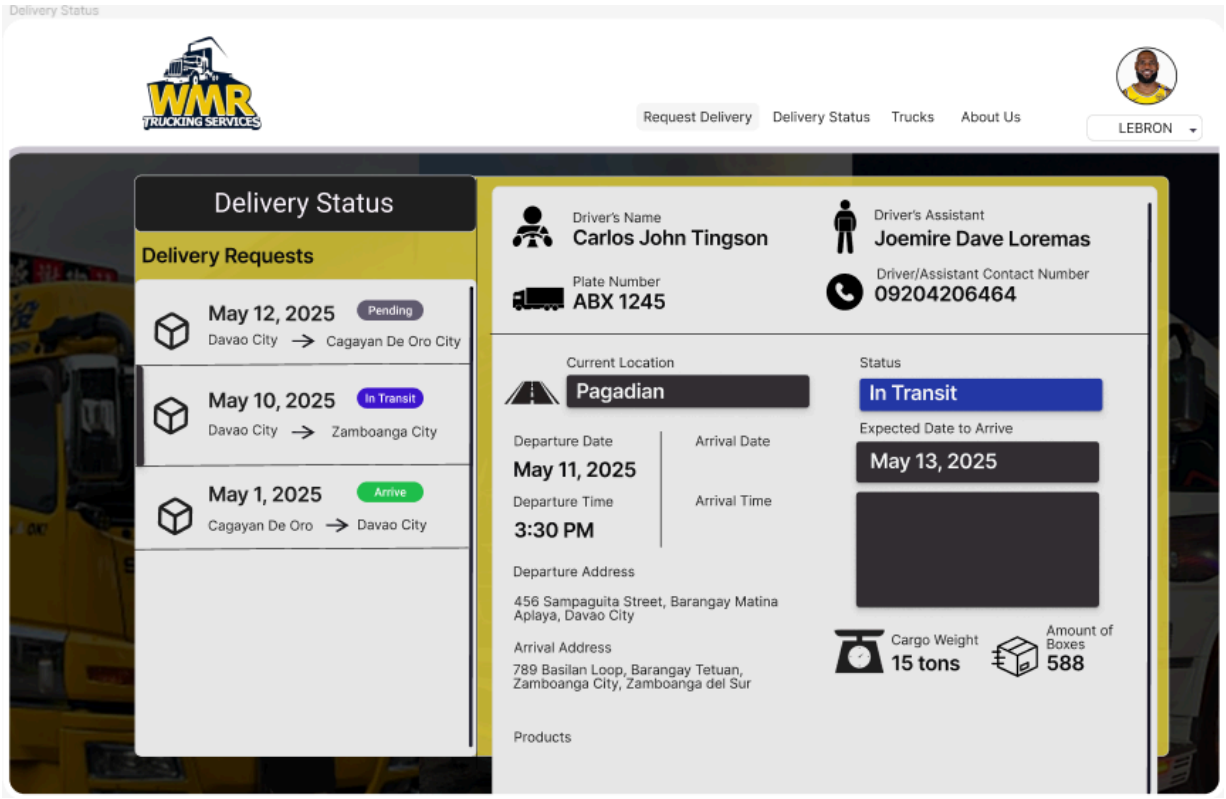
Randy Royce Ronquillo
CEO of WMR Trucking Services
+63 9088607455

SANKYU

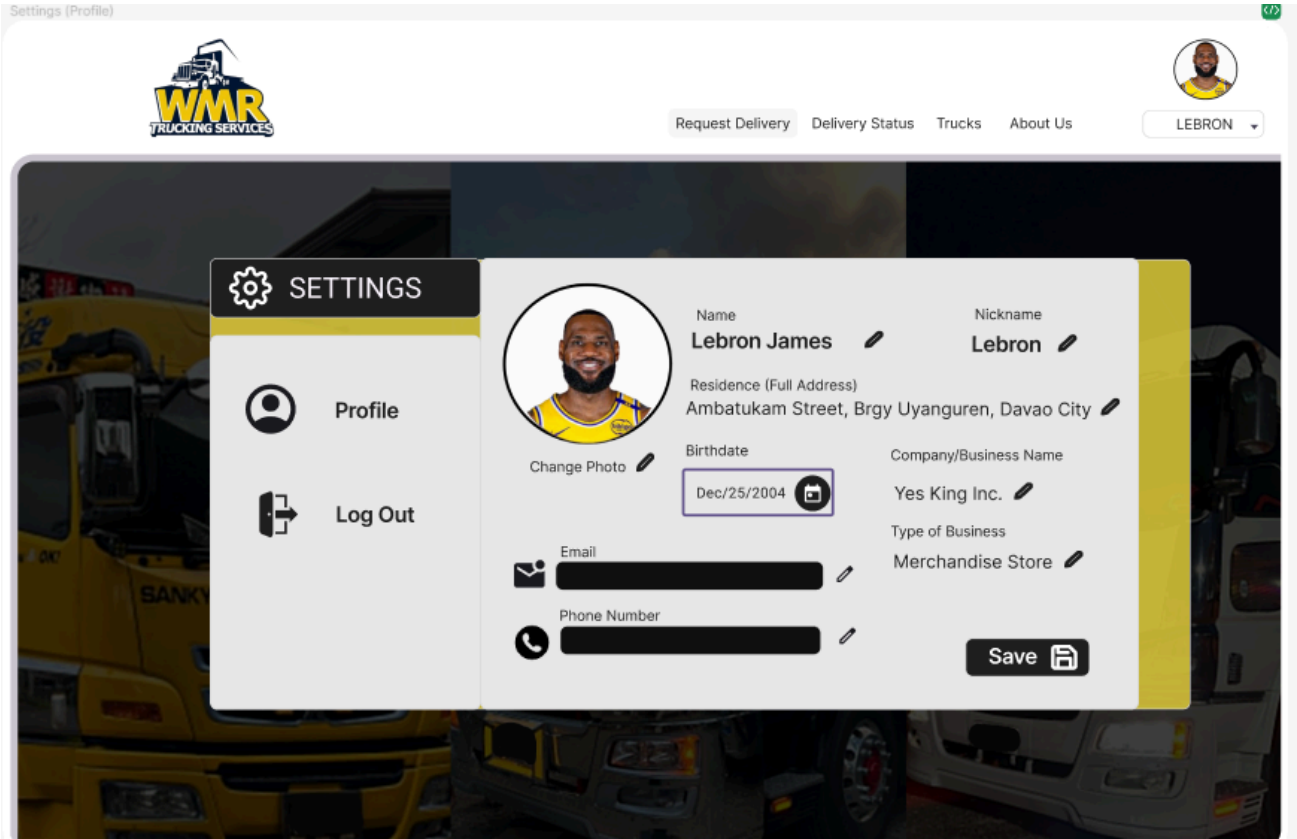
REQUEST DELIVERY

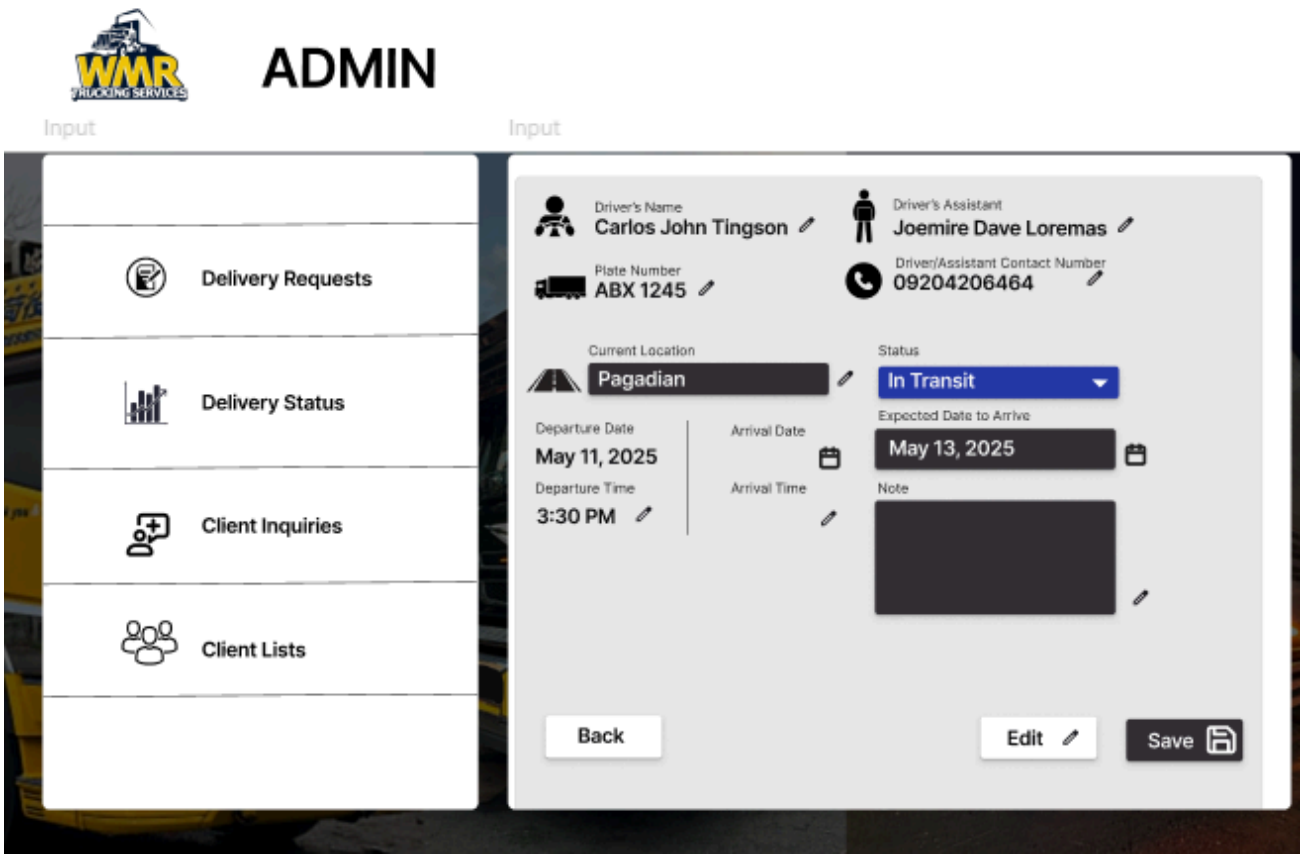


DELIVERY STATUS

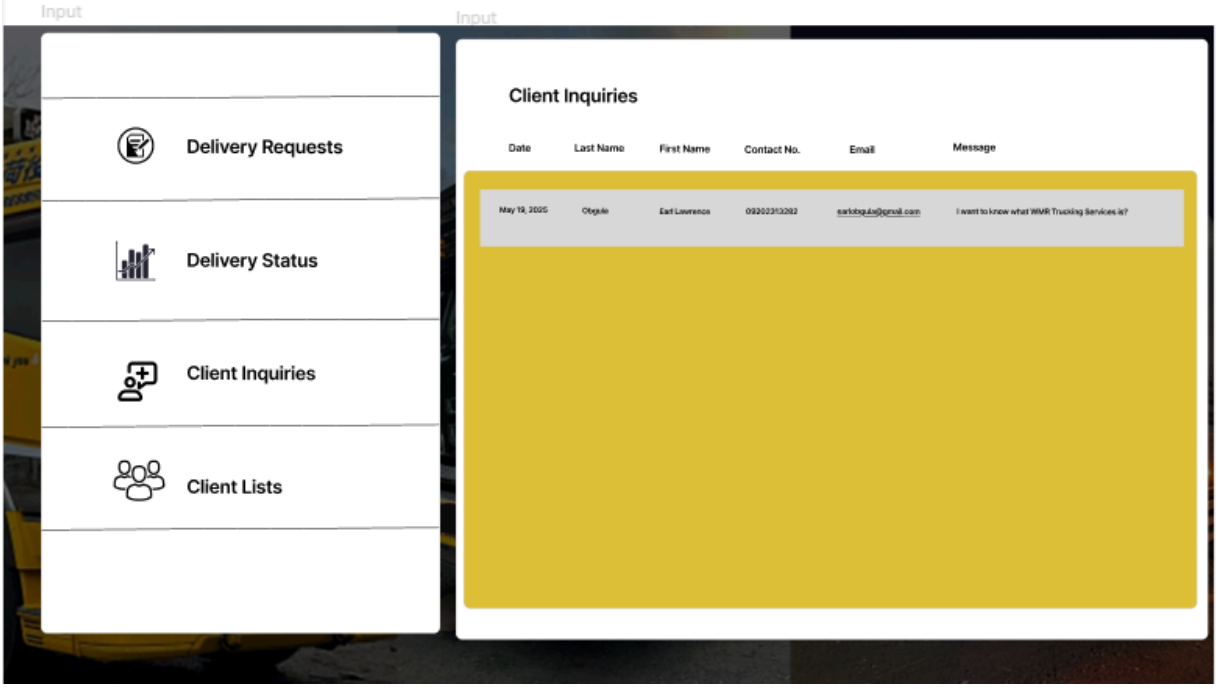


SETTINGS

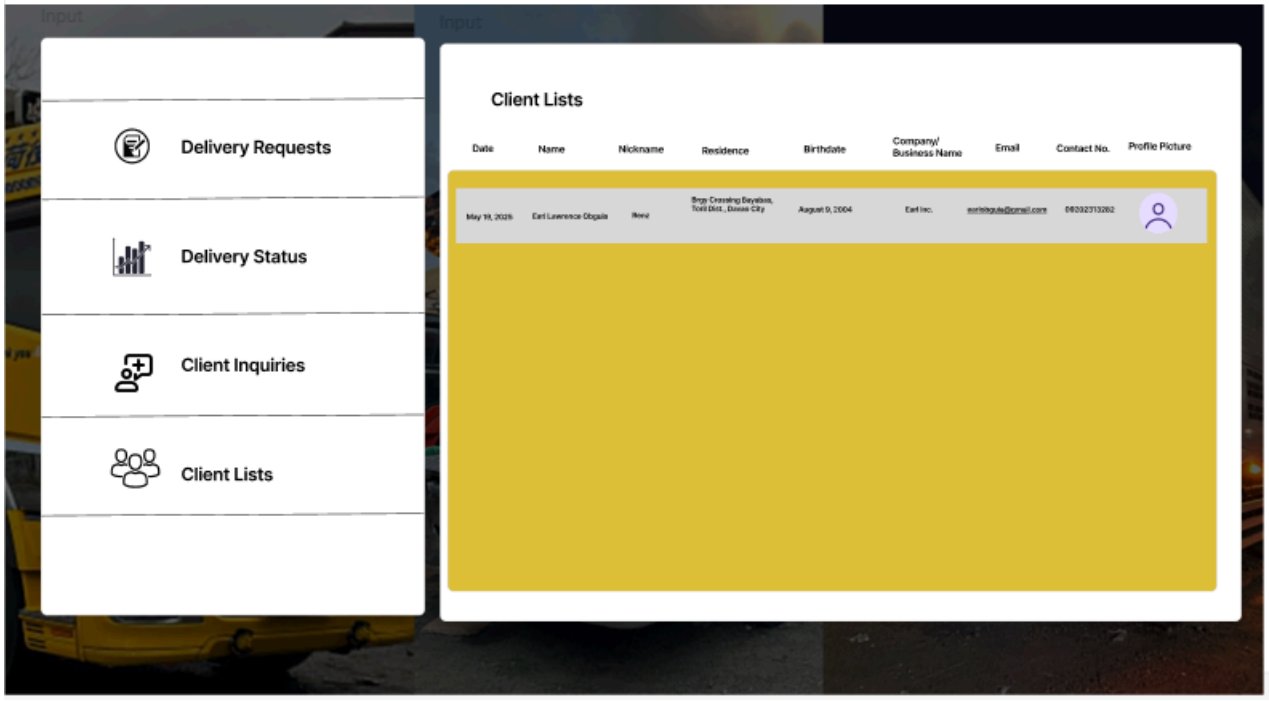


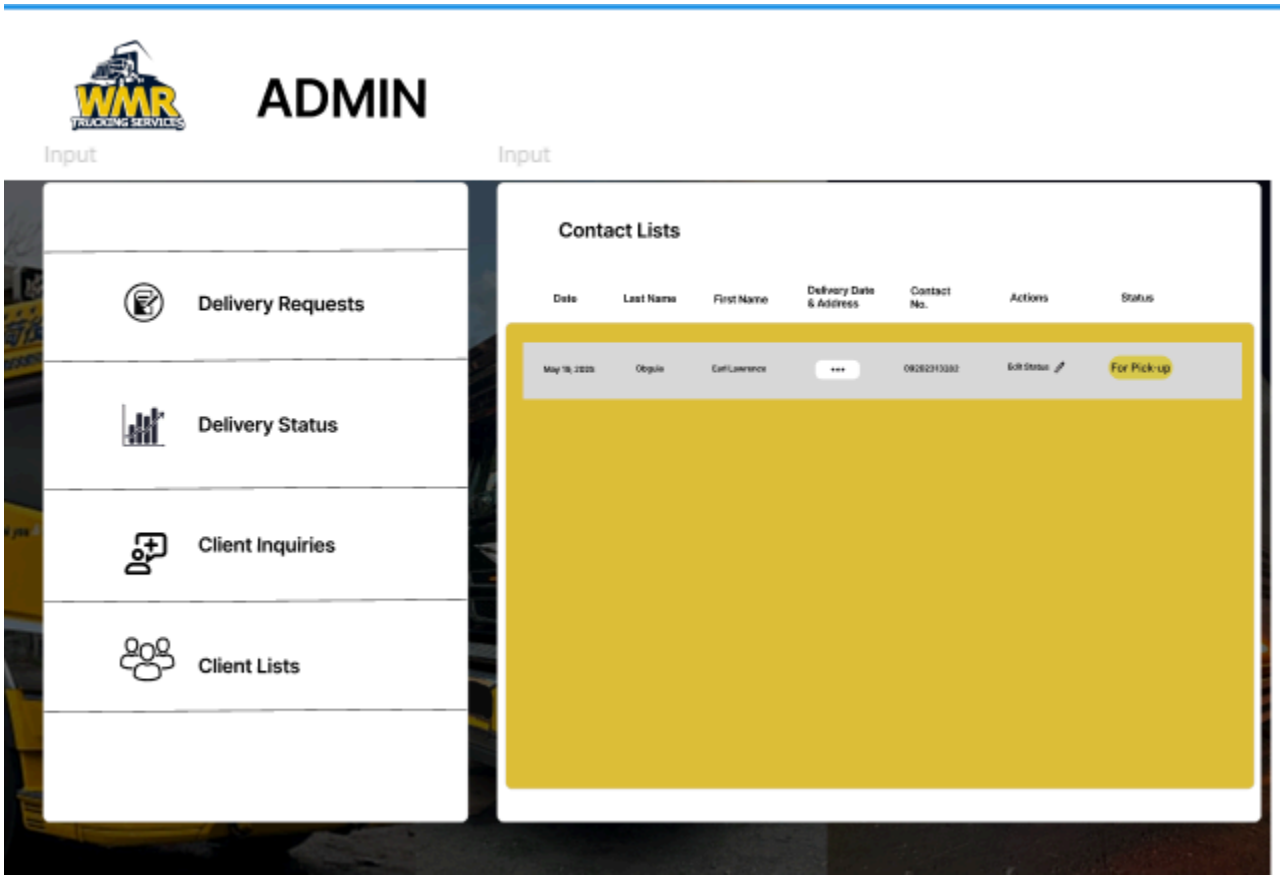

ADMIN
Input
Input
Delivery Requests
Delivery Status
Client Inquiries
Client Lists
Driver's Name
Carlos John Tingson
Driver's Assistant
Joemire Dave Loremas
Plate Number
ABX 1245
Driver/Assistant Contact Number
09204206464
Current Location
Pagadian
Status
In Transit
Departure Date
May 11, 2025
Arrival Date
Expected Date to Arrive
May 13, 2025
Departure Time
3:30 PM
Arrival Time
Note
Back
Edit
Save

ADMIN


Delivery Requests
Delivery Status
Client Inquiries
Client Lists
Client Inquiries
Date
Last Name
First Name
Contact No.
Email
Message

ADMIN


Delivery Requests
Delivery Status
Client Inquiries
Client Lists
Client Lists
Date
Name
Nickname
Residence
Birthdate
Company/ Business Name
Email
Contact No.
Profile Picture



IMPLEMENTATION PLAN

Week 1-2: Our group will focus on requirement analysis and creating wireframes. We will communicate with the company to gather the necessary system requirements, identify the core features of the website, and sketch initial wireframes to visualize the structure and user interface of the system.

Week 3-4: We will begin front-end development. This includes designing the layout of the website using HTML, CSS, and JavaScript. We will build pages such as the homepage, delivery request form, delivery status page, and contact page, ensuring the website is mobile-friendly and responsive.

Week 6-7: We will work on the back-end logic and integrate the database. This involves setting up the database (e.g., MySQL), connecting the front-end forms to the back-end using PHP, implementing user authentication, and making sure the delivery request and tracking system store and retrieve data properly.

Week 8: We will conduct testing and debugging. The goal is to identify and fix any errors or issues in the system. We will test the website's functionality, user roles, input validation, and responsiveness across different devices and browsers.

Week 9: We will prepare our final presentation and complete the project documentation. We will demonstrate the fully working system, explain its features and purpose, and submit the final files such as source codes, database, and documentation.

CODE HIGHLIGHTS

1. Advance OOP / Modular Programming
   a.) PDO Database Wrapper (db.php)
   ```
   $pdo = new PDO($dsn, $user, $pass, $options);  // OOP approach
   ```
   - Uses object-oriented PDO (PHP Data Objects) for database connections.
   - Configures error handling (ERRMODE_EXCEPTION) and fetch modes for consistency.

   b.) Modular Components (Reusable Templates)
   - header.php
     Included in multiple pages (index.php, profile.php, etc.) for consistent UI.
   - sidebar.php (Admin Dashboard)
     Separated for reuse in admin panels.

   c.) Class-like Structured Functions
   - update_profile.php
     $stmt = $conn->prepare("UPDATE users SET full_name=?, nickname=?, ...");
   - $stmt->bind_param("ssssssssi", ...);
     Uses prepared statements in a modular fashion

2. Asynchronous Communication (AJAX/Fetch API)
   a.) Google Sign-In (login.php)
   ```
   fetch('../php/verify_google_login.php', {
   method: 'POST',
   headers: { 'Content-Type': 'application/json' },
   body: JSON.stringify(data)
   })
   ```

- Fetch API for asynchronous Google OAuth handling.

b.) Dynamic Profile Picture Upload (profile.php)

```
fetch('upload_profile_picture.php', {
  method: 'POST',
    body: formData  // FormData for file upload
})
```

- AJAX file upload without page reload.

c.) Real-time Profile Updates (profile.php)

```
fetch('update_profile.php', {
    method: 'POST',
    body: JSON.stringify({ ...userData })
})
```

- Sends JSON payload to update profile fields without refreshing.

3. Session & Cookie Handling

a) Session-Based Authentication

```
// login.php
if (password_verify($password, $userPassword)) {
   $_SESSION['user_id'] = $userId;  // Stores user ID in session
   $_SESSION['username'] = $username;
}
```

- Persists login state using $_SESSION.

b.) Login Redirection Logic

```
// deliveryStatus.php
if (!isset($_SESSION['user_id'])) {
   $_SESSION['redirect_after_login'] = 'deliveryStatus.php';  // Saves requested page
   header("Location: login.php");  // Redirects to login
}
```

- Remember intended page post-login.

c.) Secure Logout (logout.php)

```
session_unset();  // Clears session vars
session_destroy(); // Destroys session
header("Location: index.php"); // Redirects
```

- Proper session termination to prevent hijacking.

4. Secure Database Access

a) Prepared Statements (SQL Injection Protection)

MySQLI(Procedural)

```
// login.php
$stmt = $conn->prepare("SELECT id, username, password FROM users WHERE username = ?");
$stmt->bind_param("s", $username);  // Input sanitization
```

PDO (OOP)

```
// db.php
$stmt = $pdo->prepare("INSERT INTO contact_messages (...) VALUES (?, ?, ?, ?, ?)");
$stmt->execute([$first_name, $last_name, ...]);
```

b.) Password Hashing

```
// login.php
if (password_verify($password, $userPassword)) { ... }  // Secure password check
```

- Uses password_hash() (implied during registration).

c.) File Upload Security (upload_profile_picture.php)

```
$filename = uniqid() . "_" . basename($file["name"]);  // Prevents overwrites
move_uploaded_file($file["tmp_name"], $targetFile);  // Validates upload
```

- Prevents directory traversal and malicious uploads.

TESTING AND VALIDATION

1. Unit Testing

   Check: Individual CSS classes & HTML elements
   Tool: Chrome DevTools & Visual Studio Code
   Example:
   Disable .hero-background in DevTools → Verify fallback

2. Integration Testing

   Check: Component interactions & responsiveness
   Tool: Chrome Device Toolbar
   Test:
   - Mobile Menu (header.css at 950px)
   - Form Submissions (login.php, request.php)

3. UI/UX Testing

   Check: Visual consistency & accessibility
   Tool: Chrome
   Fix:
   - Color Contrast (e.g., yellow buttons)
   - Image loading speed

4. User Acceptance (UAT)

   Check: Real-user flows
   Method: Peer testing
   Verify:
   - Profile Picture Upload
   - Delivery Request Submission

   Tools Used

   VS Code (Live Server for instant preview)

Chrome (DevTools, Lighthouse, Device Toolbar)

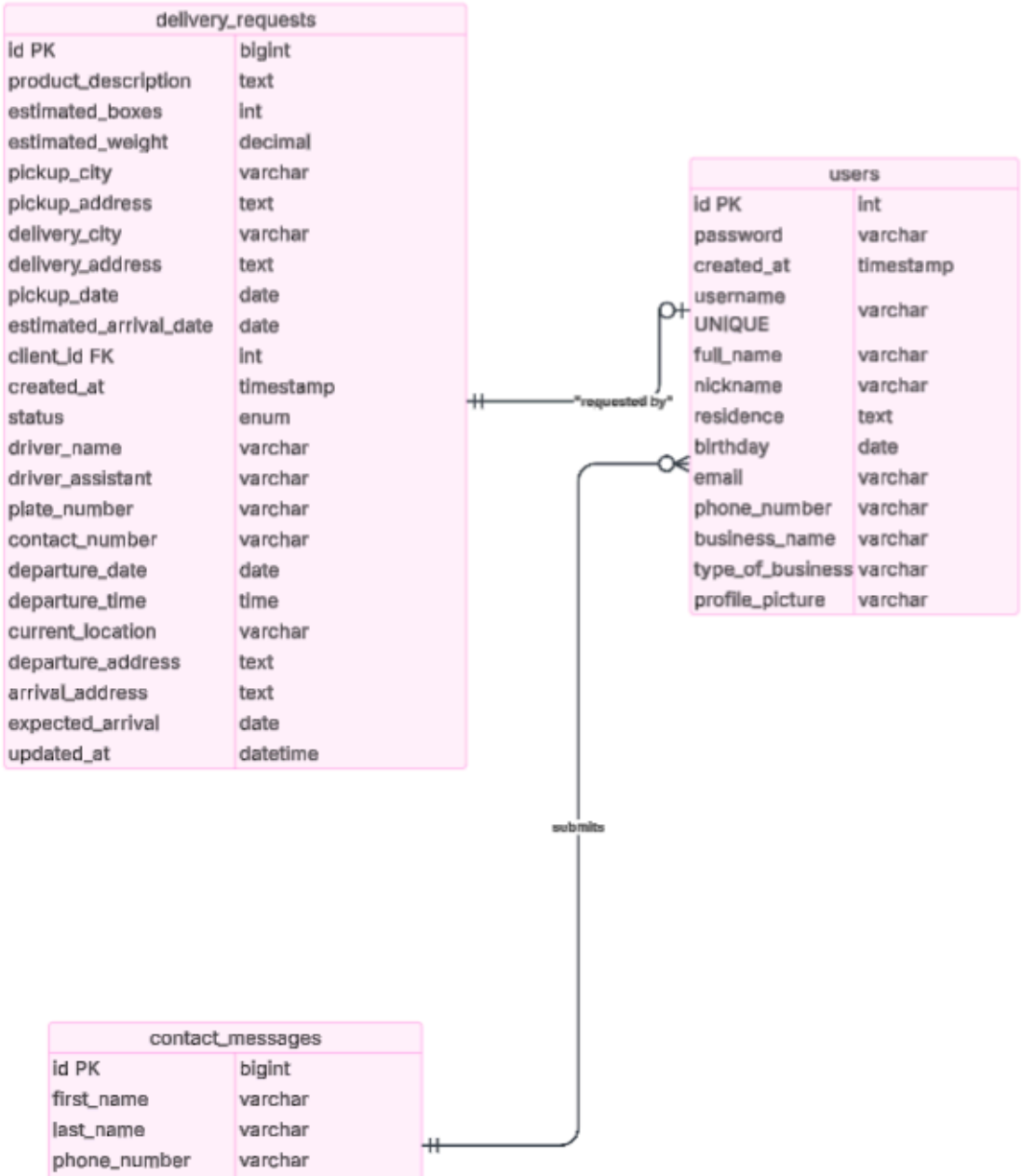
USER MANUAL

- Users can register for an account by clicking the “Sign Up” button on the homepage and filling in the required information.
- Admins and assistants use assigned credentials to log in securely through the login page.
- After logging in, clients can navigate to the "Request Delivery" page to submit a cargo booking request.
- Clients must fill in the cargo details such as the product, estimated weight or number of boxes, pick-up and delivery addresses, and desired arrival date.
- Once the form is complete, users can click the “Submit” button to send the request for processing.
- Clients can track the progress of their delivery by opening the "Delivery Status" page from the navigation bar.
- The delivery status page shows information such as the current cargo status, location, assigned driver and assistant, plate number, and their contact numbers.
- Users can update their personal details by accessing the "Settings" section, where they can edit their name, password, and address.
- If a user forgets their password, they can use the password recovery feature available on the login page.
- The contact page allows users to send inquiries to the company by filling in their name, phone number, and email.

DEVELOPER GUIDE

To set up the WMR Trucking project, first clone the repository to your local machine. Install VS Code with the Live Server extension for real-time previews, and ensure you have Chrome or Firefox for testing. Set up XAMPP or WAMP to run Apache and MySQL servers locally for PHP processing. Import the provided wmr_db.sql file into PHPMyAdmin to initialize the database. Finally, launch the project through your local server environment to begin development and testing.

DATABASE SCHEMA/ERD



Summary of key learnings and features

Through this project, we learned how to design and implement a database-driven cargo delivery system that efficiently manages bookings, driver assignments, cargo details, and transaction records. Key features include automated updates via database triggers, comprehensive tracking of cargo status, and clear role-based data management. This system enhances the company's ability to monitor deliveries and improve client satisfaction.

Suggestions for future upgrades or new features

Future improvements could include adding GPS-based real-time tracking, integrating online payment processing, and enabling automated notifications for clients and staff. Implementing stronger user authentication and role-based access controls would improve security. Additionally, developing a mobile-friendly interface and route optimization features could further streamline operations and improve usability.