

AMMAN ARAB UNIVERSITY

جامعة عمان العربـيــــــة

Faculty of IT

**كلية تكنولوجيا المعلومات**

PackTrack

Students:

|  |  |
| --- | --- |
| Sdra Awameh | 202210368 |
| Rasha Alsaleh  Dalaa Saqer | 202210632  202210366 |

*Supervisor:*

Dr. Alaa Abuthawabeh

*Amman – Jordan*

*2024/2025*

Table of Contents

[Chapter One](#CHAPTER_ONE)

[INTRODUCTION](#INTRODUCTION) 1

[1.1 Problem Statement](#_1.1_Problem_Statement) 1

[1.2 Business case](#_1.2_Business_Case) 1

[1.3 Aim and Objectives](#_1.3_Aim_and) 1  
[1.4 Project Scope](#_1.4_Project_Scope) 2

[Chapter Two](#CHAPTER_TWO)

[Project Planning and Requirements](#PROJECT_PLANNING_AND_REQUIREMENTS) 3

[2.1 Project Plan and Schedule](#_2.1_Project_Plan) 3

[2.1.1 Gantt Chart](#_2.1_Project_Plan) 3

[2.2 Functional Requirements](#_2.2_Functional_Requirements)  4

[2.3 Non-Functional Requirements](#_2.3_Non-Functional_Requirements)  4

[2.4 Software Development Methodology (SDLC Model)](#_2.4_Software_Development) 5

[Chapter Three](#CHAPTER_THREE)

[SYSTEM ANALYSIS AND DESIGN](#SYSTEM_ANALYSIS_AND_DESIGN)  6

[3.1 System Architecture](#_3.1_System_Architecture) 6

[3.2 Use Case Diagram](#_3.2_Use_Case) 7

[3.3 Relational Schema](#_3.3_Relational_Schema) 8

[3.4 ER Diagram](#_3.4_ER_Diagram) 8

[3.5 Class Diagram](#_3.5_Class_Diagram) 9

[3.6 UI Mockups](#_3.6_User_Interface) 10

Chapter One

Introduction

### 1.1 Problem Statement

In today’s fast-paced digital era, online shopping has become an integral part of everyday life. Individuals frequently receive multiple deliveries from different courier services such as Aramex, DHL, and FedEx. However, there is a significant gap when it comes to managing these deliveries in a centralized and user-friendly way.

Currently, users must depend on scattered and inconsistent sources like SMS messages, emails, and various courier apps to track their packages. This fragmented experience often leads to confusion, lost tracking numbers, delayed updates, and difficulty supporting delivery records. Additionally, there is no common platform that helps users manage, organize, or analyze their personal delivery history in one place.

Despite the increasing reliance on e-commerce, there is no lightweight tool available specifically tailored for personal delivery management — one that combines real-time package tracking, smart organization, and helpful tips to make the delivery experience more efficient and convenient. This lack of a centralized solution creates unnecessary hassle, especially for individuals who frequently receive packages and want better visibility and control over their deliveries.

### 1.2 Business Case

The **PackTrack** system is designed to fill this gap by providing a centralized, web-based solution that helps users manage, track, and analyze their personal deliveries. It acts as a personal delivery dashboard where users can log shipments, receive real-time status updates via courier APIs, and get value-added content like packaging or motivation-related tips.

Unlike traditional courier apps that focus on individual shipments, PackTrack allows users to manage multiple deliveries from **different couriers** all in one place. It eliminates the need to jump between apps or dig through emails for tracking links. Additionally, by offering features such as **delivery logs, and smart tips**, PackTrack enhances the user experience and offers practical tools to stay organized.

### 1.3 Aim and Objectives

The main aim of this project is to design and develop a personal delivery tracking web application that allows users to efficiently manage their deliveries, track their real-time status, and benefit from helpful packaging tips — all through an interactive dashboard.

**The key objectives of this project are:**

* To implement JWT-based user authentication for secure access and role-based dashboard features.
* To allow users to create, update, and manage their delivery records through an intuitive dashboard.
* To integrate a real-time courier tracking API (e.g., TrackingMore API) that fetches live delivery status updates.
* To display motivational quotes and packaging tips, creating an engaging user experience.
* To develop an Admin Panel for managing users and viewing or updating all delivery logs.
* To ensure the system is mobile responsive, visually clean, and user-friendly.
* To apply DevOps practices, including CI/CD pipelines using GitHub Actions and Docker for streamlined development and deployment workflows.

### 1.4 Project Scope

The scope of the PackTrack project is centered on building a full-stack web application that serves the needs of individuals who regularly receive deliveries. The system will include both **User** and **Admin** interfaces, each with specific features and access levels. The goal is to provide a simple yet professional platform that centralizes delivery data and improves delivery management for users.

* **User Authentication System:** Login and registration using JWT, with user roles (Admin/User).
* **User Dashboard:** A personalized interface where users can view and manage their delivery logs.
* **Add Delivery Form:** Users can add new delivery records including courier name, tracking number, item title, and expected delivery date.
* **Delivery Tracking:** Real-time status updates from courier services via third-party API integration.
* **Search:** Users can easily find specific delivery logs or view sorted data.
* **Quote/Tip Display:** A section displaying random motivational or packaging-related quotes.
* Admin Dashboard: Admins can manage all delivery records, users, and manually update statuses.
* PDF Export Feature: Users and Admins can export delivery records in printable PDF format.
* DevOps Tools Integration: GitHub Actions for CI/CD and Docker containerization for consistent environment deployment.

Chapter Two

Project Planning and Requirements

### 2.1 Project Plan and Schedule

The PackTrack project will be developed over several stages, following a structured timeline to ensure efficient progress and timely delivery. The project team has divided the tasks across planning, analysis, design, implementation, testing, and deployment phases.

Table 2.1 PackTrack Gantt Chart



### 2.2 Functional Requirements

FR1 User Registration and Login (JWT Auth): Users must be able to register and log in securely using JSON Web Token-based authentication.

FR2 Add Delivery Log: Users can add a new delivery record by entering courier name, tracking number, item title, and expected delivery date.

FR3 View Delivery Records: Users can view a list of all their deliveries with status, date, and item details.

FR4 Search, Filter & Sort Records: Users can search delivery logs and sort/filter by status or delivery date.

FR5 Display visual analytics charts on the user dashboard, including delivery status distribution, courier usage, and delivery trends over time.

FR6 Courier Tracking Integration: System fetches real-time delivery status using a third-party tracking API (e.g., TrackingMore API).

FR7 Smart Tips & Quotes: Random tips or quotes related to packaging, motivation, or delivery are shown to users.

FR8 Admin Panel: Admin can view all user delivery logs, manage user accounts, and update delivery statuses.

### 2.3 Non-Functional Requirements

NF1 Responsiveness: The UI must be responsive and improved for both desktop and mobile screens.

NF2 Usability: Interface must be intuitive and user-friendly for all types of users.

NF3 Performance: The app must fetch tracking data quickly and display it in real-time.

NF4 Security: JWT authentication will ensure secure access to user data.

NF5 Maintainability: Code will follow standard documentation and modular structure for easier maintenance.

NF6 Scalability: The application structure will support future scalability and new feature integration.

### 2.4 Software Development Methodology (SDLC Model)

For the development of PackTrack, we will follow the Incremental SDLC Model, which enables the system to be built and delivered in structured increments. This approach is well-suited for the project as it allows:

* Step-by-step feature development to ensure functionality is built gradually.
* Flexibility to update requirements based on feedback in each phase.
* Ongoing testing to maintain system reliability after every increment.

Each increment will follow a structured cycle including planning, implementation, testing, and integration of specific feature group (e.g., user authentication, dashboard, API integration, admin panel).

This approach minimizes complexity while ensuring a scalable and maintainable solution.



Figure 2.2 Incremental SDLC Model

Chapter Three

System Analysis and Design

### **3.1**System Architecture

**Figure 3.1** illustrates the system architecture of the PackTrack platform, presenting a high-level view of its layered structure and component interactions. The architecture is divided into five main layers: Client, API Controller, Middleware, Model, and Database. Each layer plays a distinct role in managing requests, processing data, and ensuring secure and efficient communication between the frontend and backend. The system is also containerized using Docker for modular deployment.

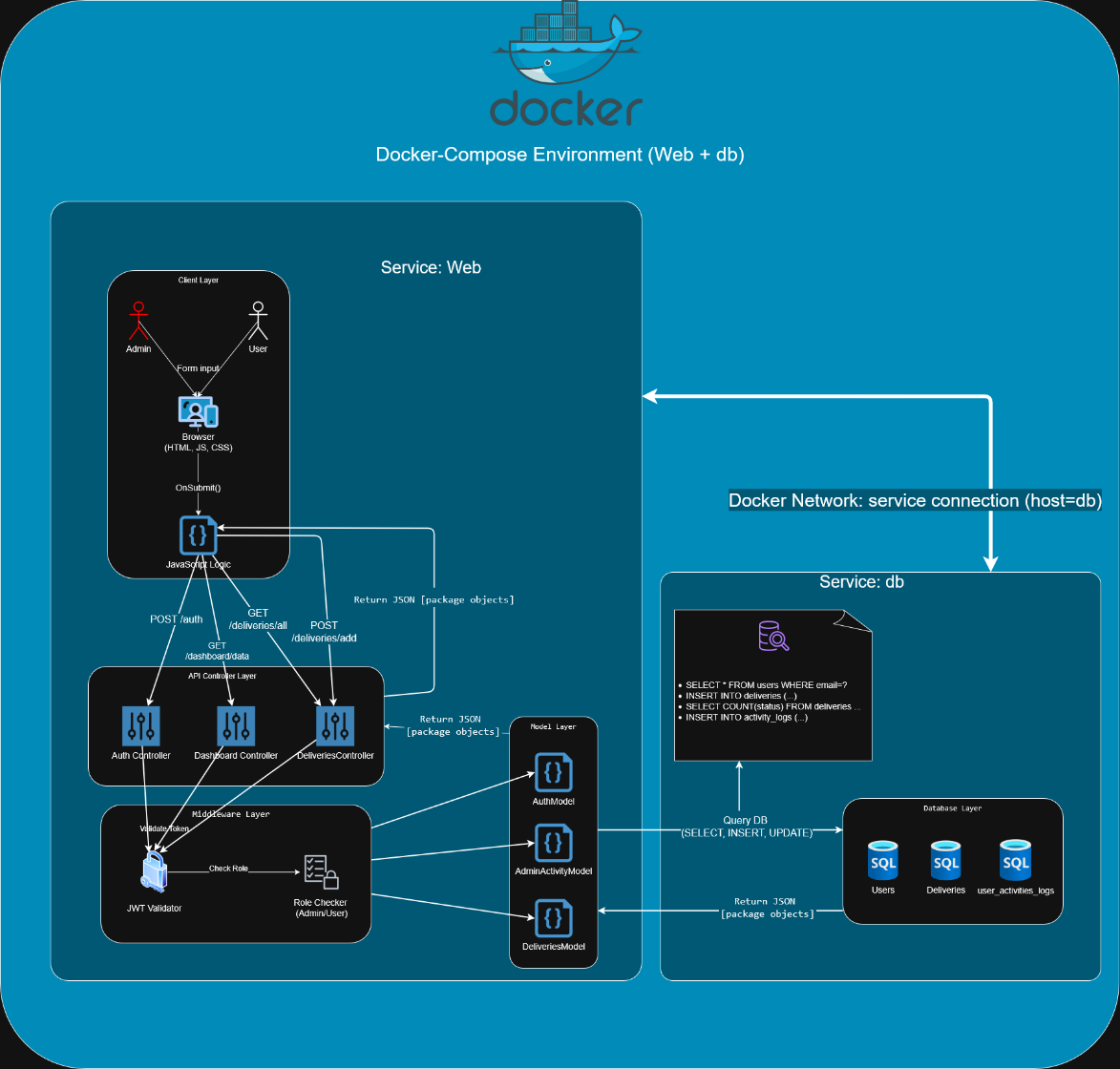


Figure 3.1 System Architecture

### **3.2**Use Case Diagram

In Figure 3.2, illustrates the use case diagram for the PackTrack System, highlighting the system’s core operations. It outlines two main actors: the User and the Admin, and the specific functions each actor can perform. The diagram also shows shared functions, such as Login, which are accessible by both actors.

Each actor has distinct privileges: the User primarily interacts with their own delivery records, tracking statuses, and performing actions like exporting data. Meanwhile, the Admin has broader system control, including managing delivery statuses, overseeing user records, and exporting system-level reports.

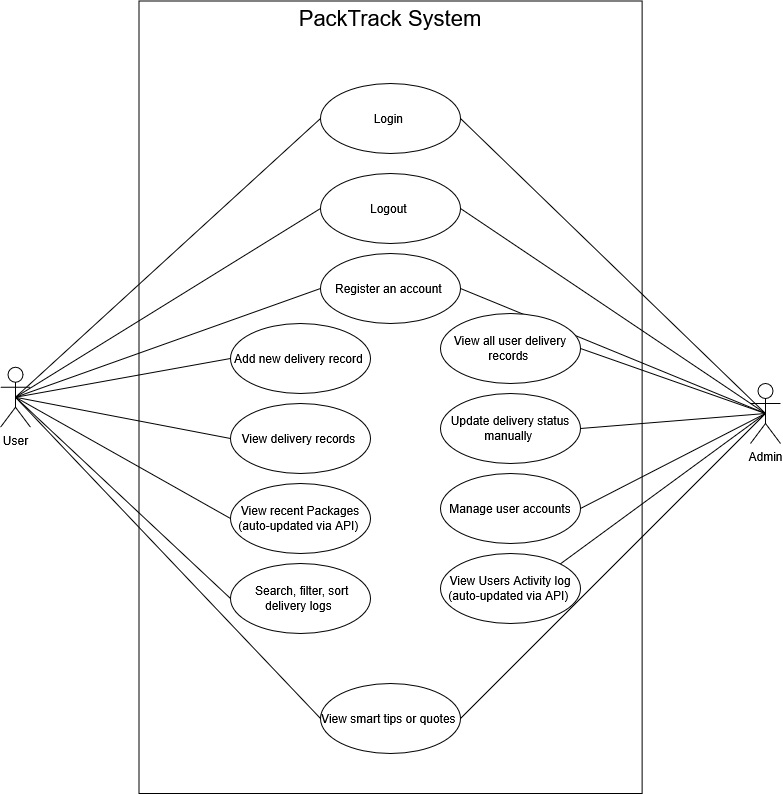


Figure 3.2 Use Case Diagram

### **3.3**Relational Schema

Figure 3.3 illustrates the relational schema of the PackTrack system. It defines the core database structure, including three main tables: users, deliveries, and user\_activity\_logs. The schema outlines how these entities are related using primary and foreign keys, ensuring data consistency. Each user can have multiple deliveries and activity logs, establishing one-to-many relationships that reflect user interactions and package tracking activities.

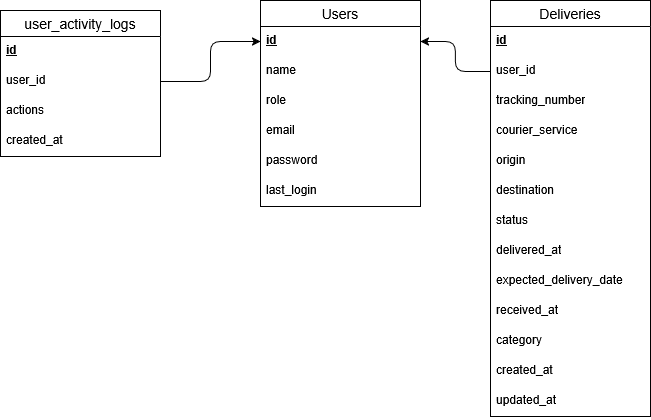


Figure 3.3 Relational Schema

### **3.4**ER Diagram

In Figure 3.4, ERD provides a conceptual overview of the database structure using Chen notation. It illustrates key entities, their attributes (with primary keys underlined), and the relationships between them using diamonds. This diagram helps visualize how entities such as users, deliveries, and activity logs are logically connected within the system.

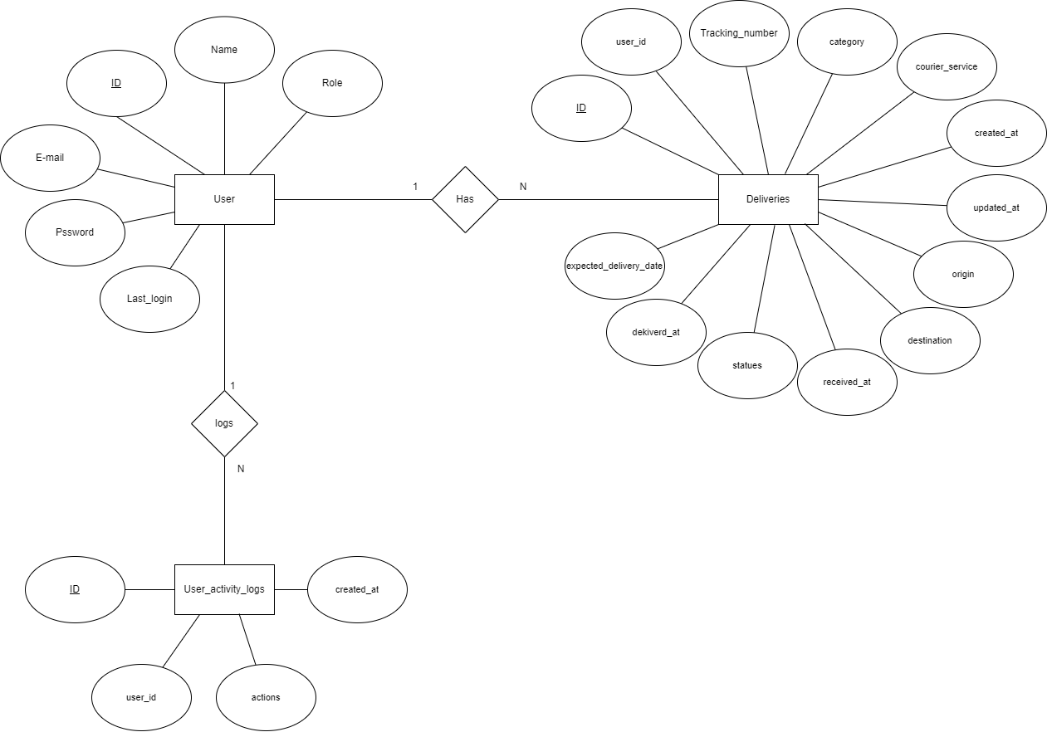


Figure 3.4 ER Diagram

### **3.5**Class Diagram

Figure 3.5 illustrates the Class Diagram of the PackTrack system, modeling its core components using object-oriented principles. It defines the system’s structure through classes representing users, deliveries, activity logs, and dashboards, along with their attributes, methods, and inter-class relationships such as associations and generalizations.

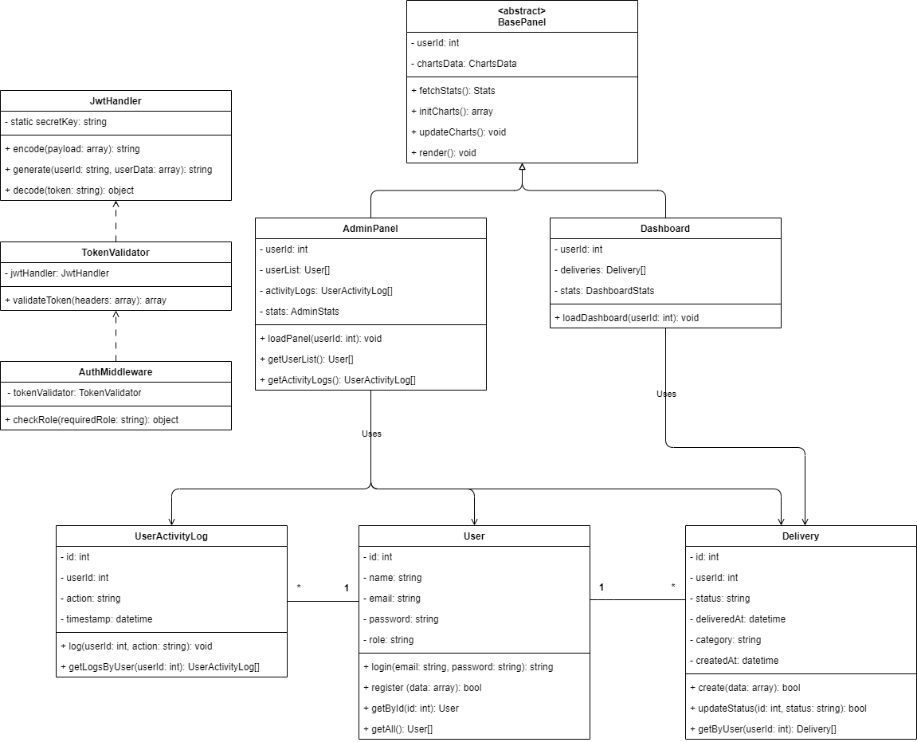


Figure 3.5 Class Diagram

### **3.6**User Interface Mockups

We designed basic wireframes for the most essential screens of PackTrack. The UI aims to be simple, clean, and user-friendly with a dashboard-style layout.

Figure 3.6.1 shows the login form used by users and admins to access the system. The login process is secured using a JWT-based authentication mechanism, ensuring that access to dashboards is restricted to logged-in users only.

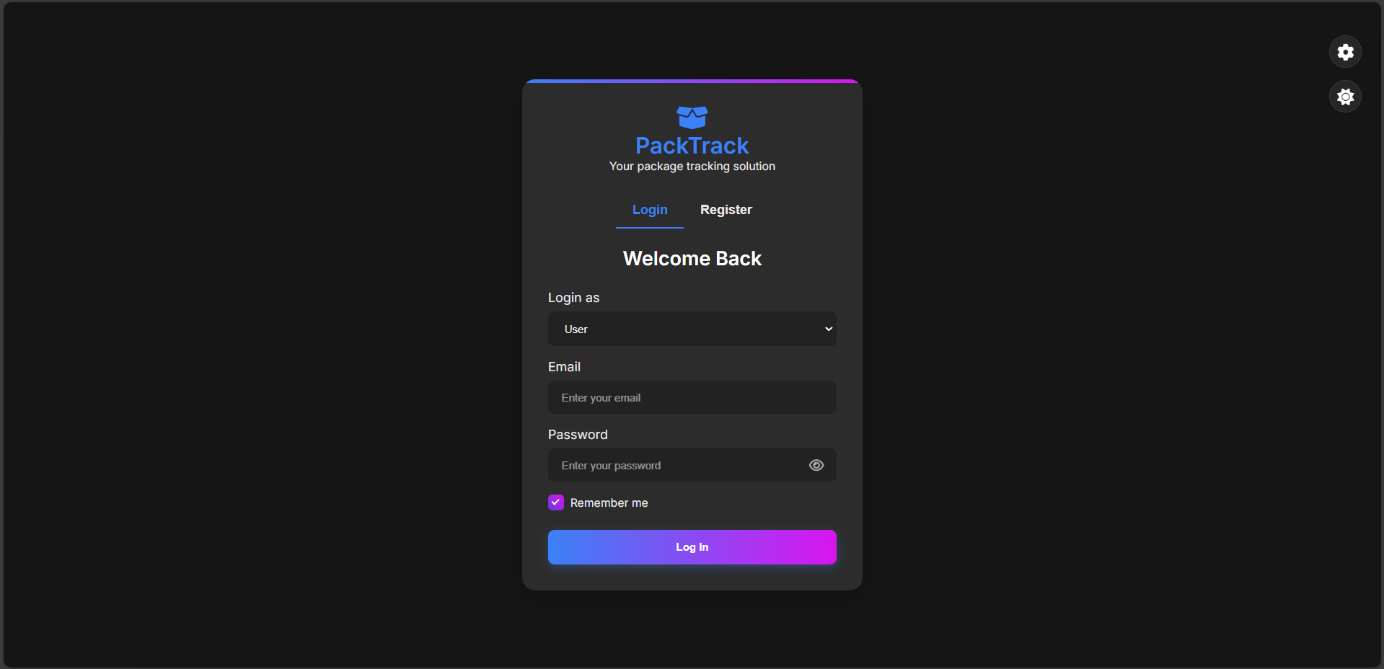


Figure 3.6.1 Login Form (JWT-Based Authentication)

Figure 3.6.2 displays the registration form. Users can sign up as either a User or an Admin. When selecting Admin, an additional field for Admin Secret Key appears. Only users with this key (provided by system owners or managers) can register as admins.

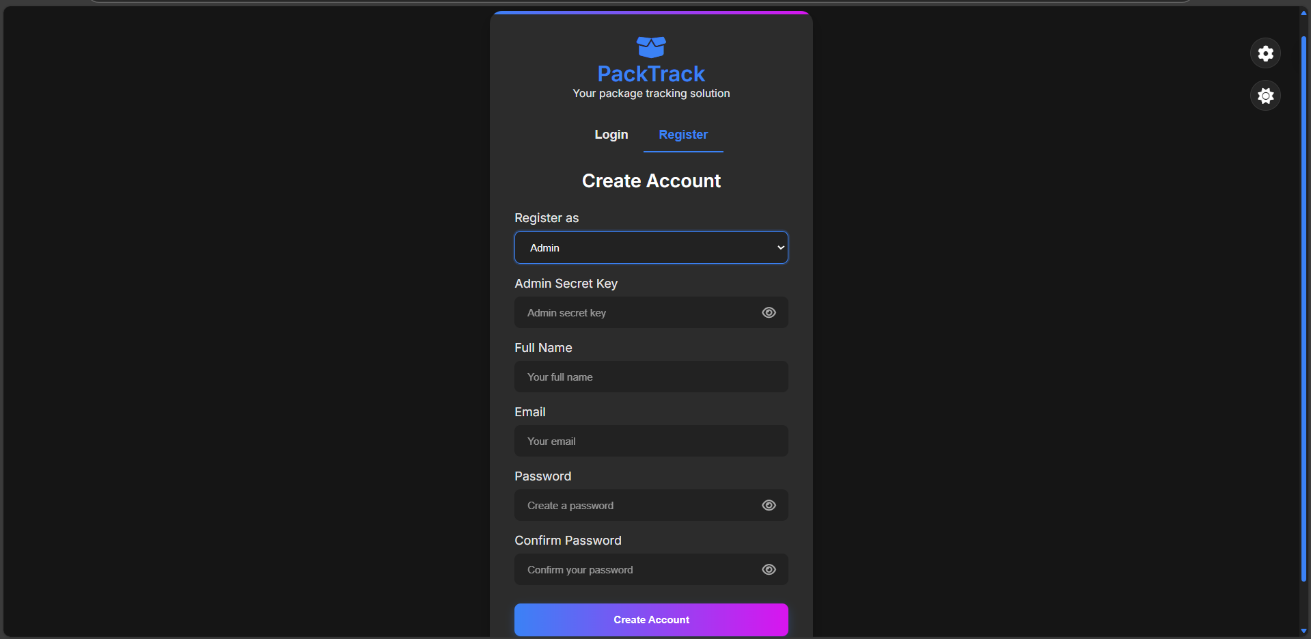


Figure 3.6.2 Registration Form with Role-Based Access

Figure 3.6.3 illustrates the User Dashboard, accessible exclusively to authenticated users, this dashboard provides an overview of package statuses for the current month, along with interactive data visualizations:

* Package Activity Chart (line chart): Tracks the number of received and delivered packages over a selected period (week, month, or year).
* Package Status Chart (donut chart): Displays the distribution of package statuses, including delivered, in transit, out for delivery, and delayed.

The dashboard also features an Add Delivery Form for sending new package details and a Delivery Logs Table for tracking past deliveries. A navigation menu ensures seamless interaction with other sections of the application.

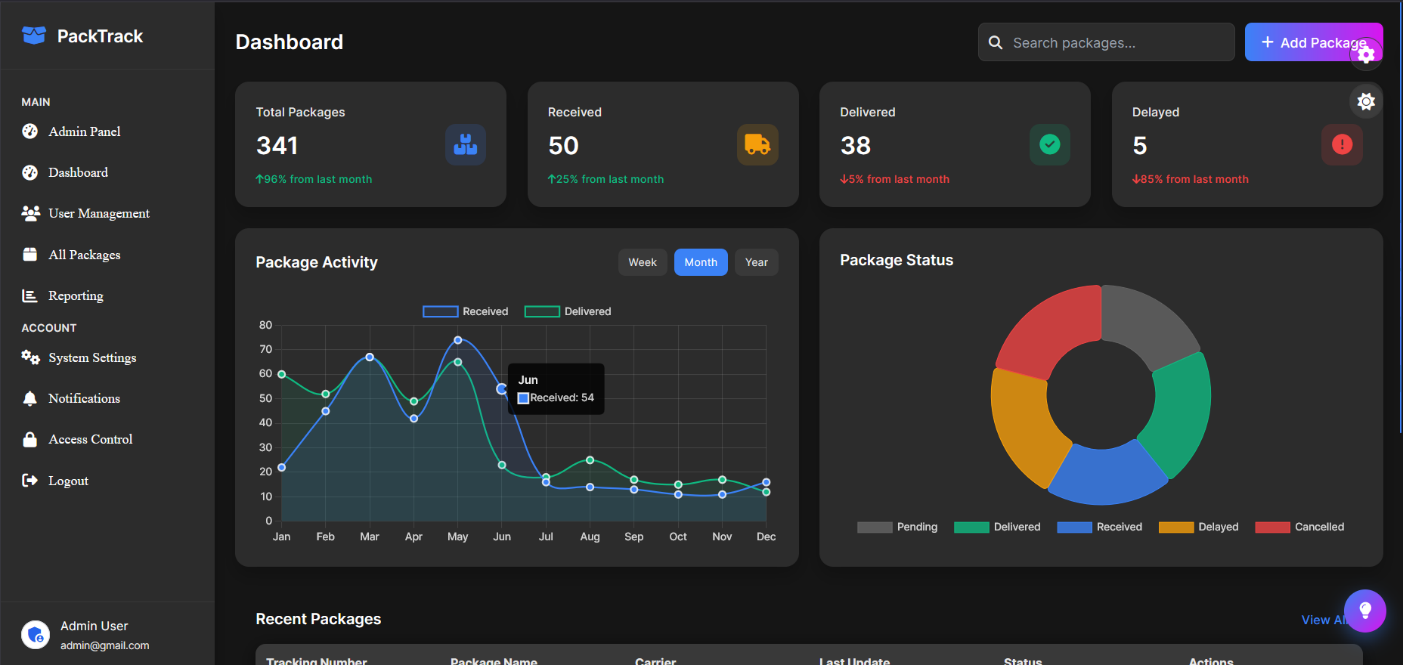


Figure 3.6.3 User Dashboard

Figures 3.6.4 and 3.6.5 present the Admin Panel, which is accessible exclusively to authenticated administrators. The panel provides a comprehensive dashboard for managing users, tracking package statuses, and monitoring system performance. Key functionalities include:

* User Management: View registered users, track their last login activity, and manage their accounts.
* System Statistics: Displays key metrics, like total users, total packages, active users, and delayed packages, with percentage-based trend indicators.
* Analytics & Reports: Interactive charts supporting data-driven decision-making.
* Activity Logs: Logs recent user actions, including login timestamps and packages status changes.



Figure 3.6.4 Admin panel Overview

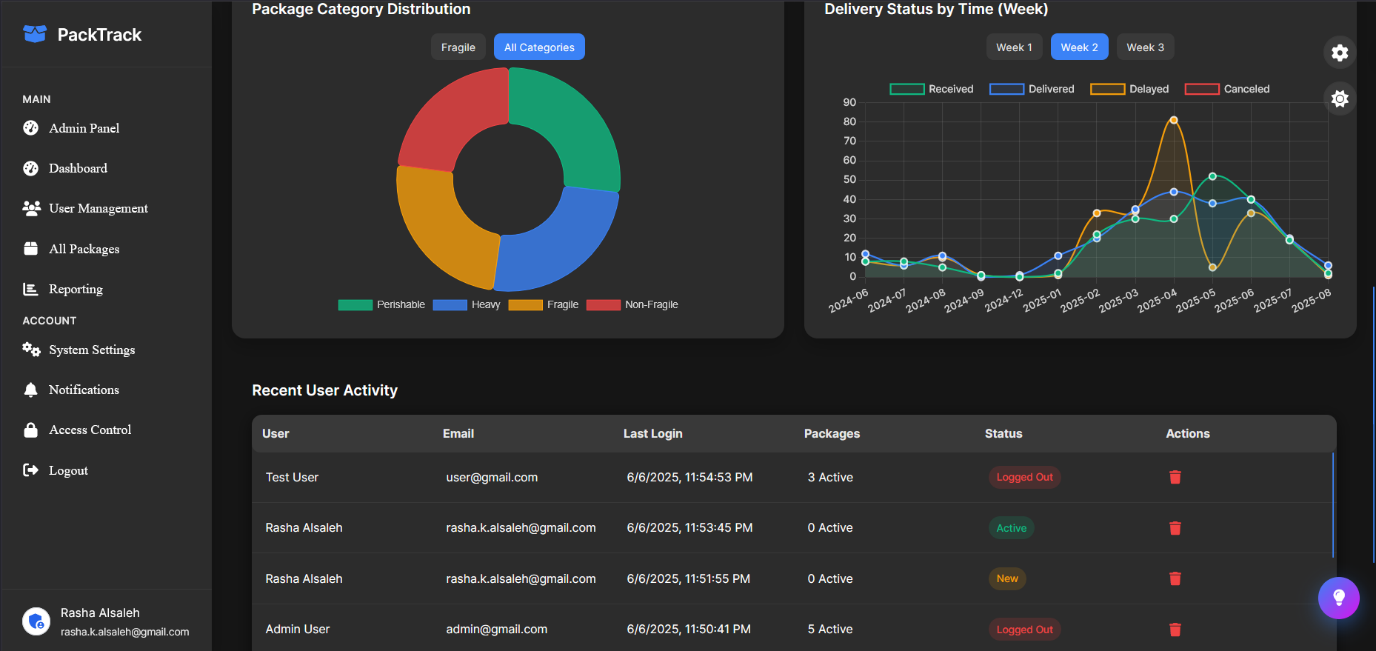


Figure 3.6.5 Admin panel User Management & Recent Users Activity Log

Figure 3.6.6 shows a dashboard widget that adds a motivational touch to the dashboard by displaying daily quotes. It dynamically pulls content from two sources: the Quotable API (a public quote service) and a custom-built API at /PackTrack/backend/api/random/quotes.json, which includes personalized quotes crafted to match the theme of the PackTrack platform. These custom quotes often reference productivity, smart delivery practices, or user encouragement. The widget allows users to refresh the quote manually, enable automatic updates, or hide it based on their preferences.

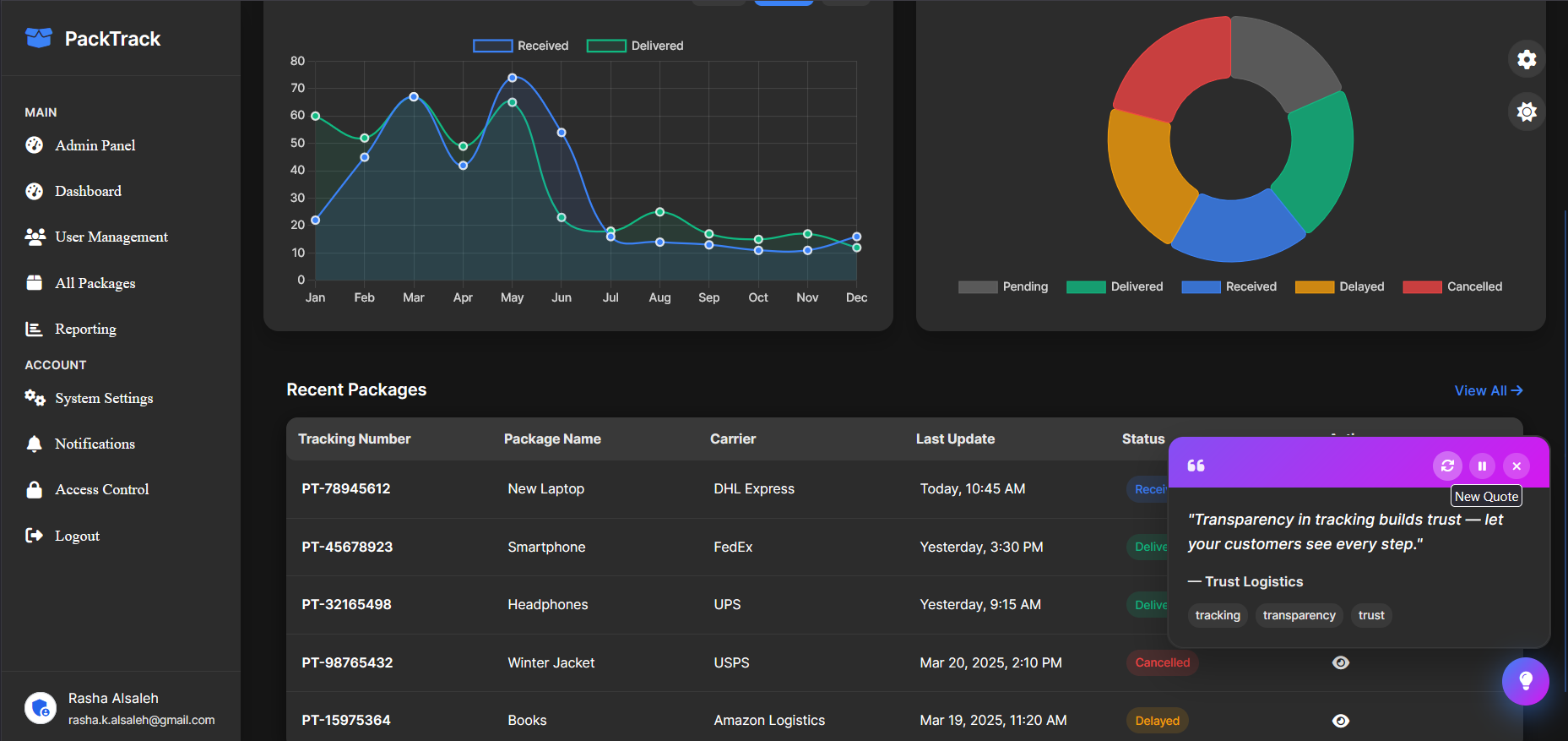


Figure 3.6.6 Daily Quote Widget

Figure 3.6.7 appears when a user tries to access a protected route without the required permissions (e.g., accessing the admin panel as a regular user). It enforces role-based access control and ensures only authorized users can view sensitive content.

Key Points:

* Security Enforcement: Blocks unauthorized access based on user roles validated through JWT tokens.
* Clear User Feedback: Friendly error message and provides quick navigation to login or dashboard.

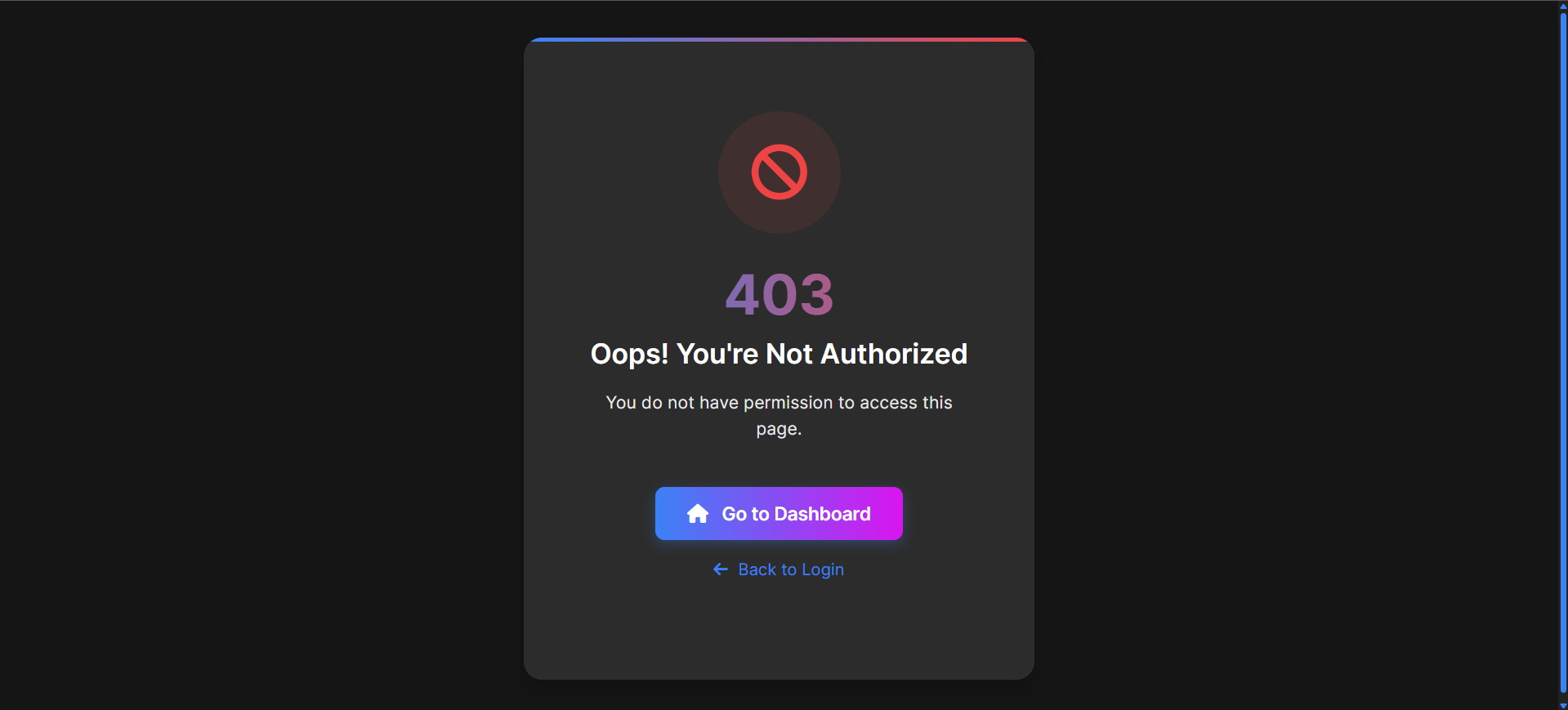


Figure 3.6.7 Forbidden Access Page (403)

Figure 3.6.8 is displayed when an unexpected issue occurs on the server side, preventing the requested operation from being completed.

Key Points:

* Error Handling: Captures server-side issues and displays an informative error message to users.
* User Experience: Offers options to navigate back to the dashboard or reload the page, ensuring users can recover from errors quickly.
* Security: Does not reveal sensitive server details, protecting the system from potential exploits.

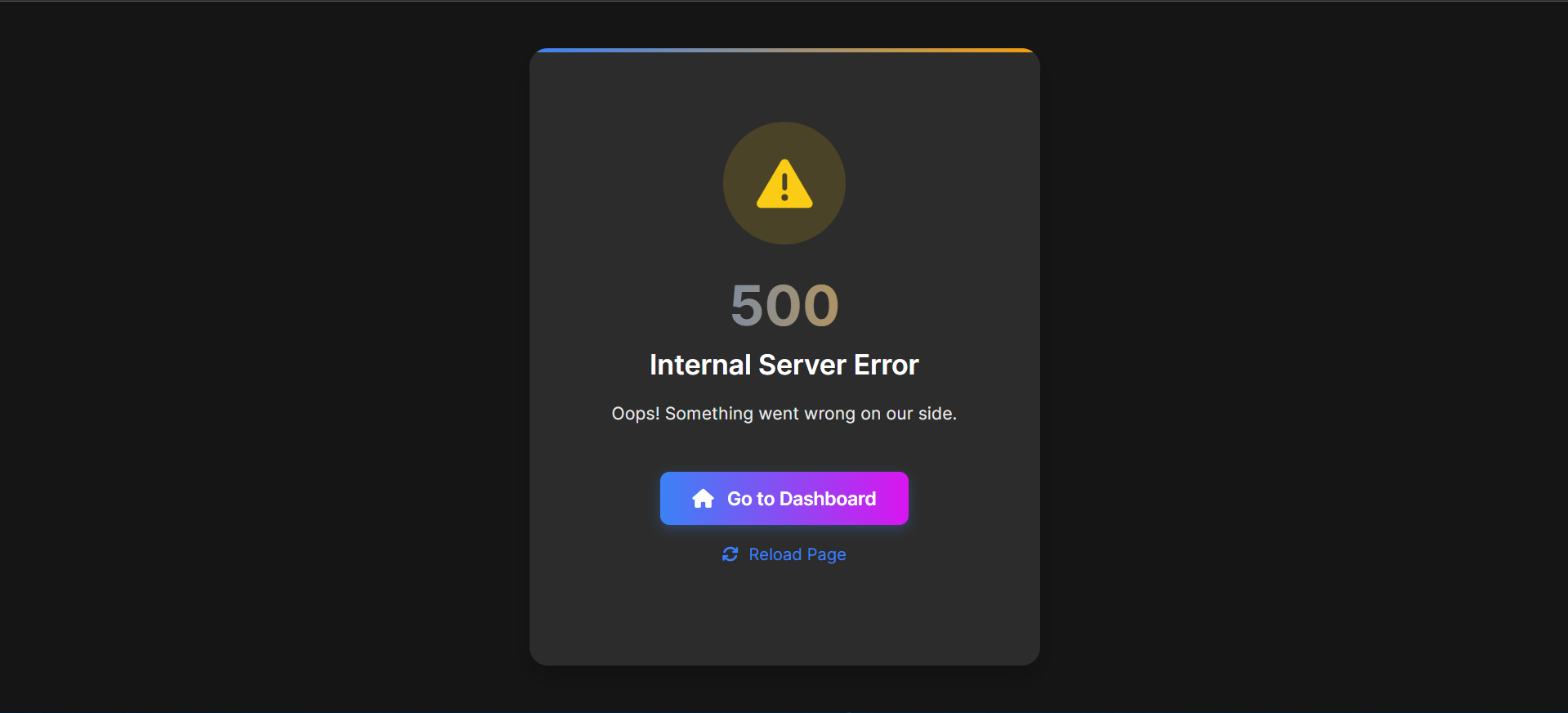


Figure 3.6.8 Internal Server Error (500)

Figure 3.6.9 enhances the user experience by displaying an animated loading screen during page transitions or while retrieving user data. With dynamic messages and helpful tips, it keeps users informed and engaged, effectively reducing perceived wait time.

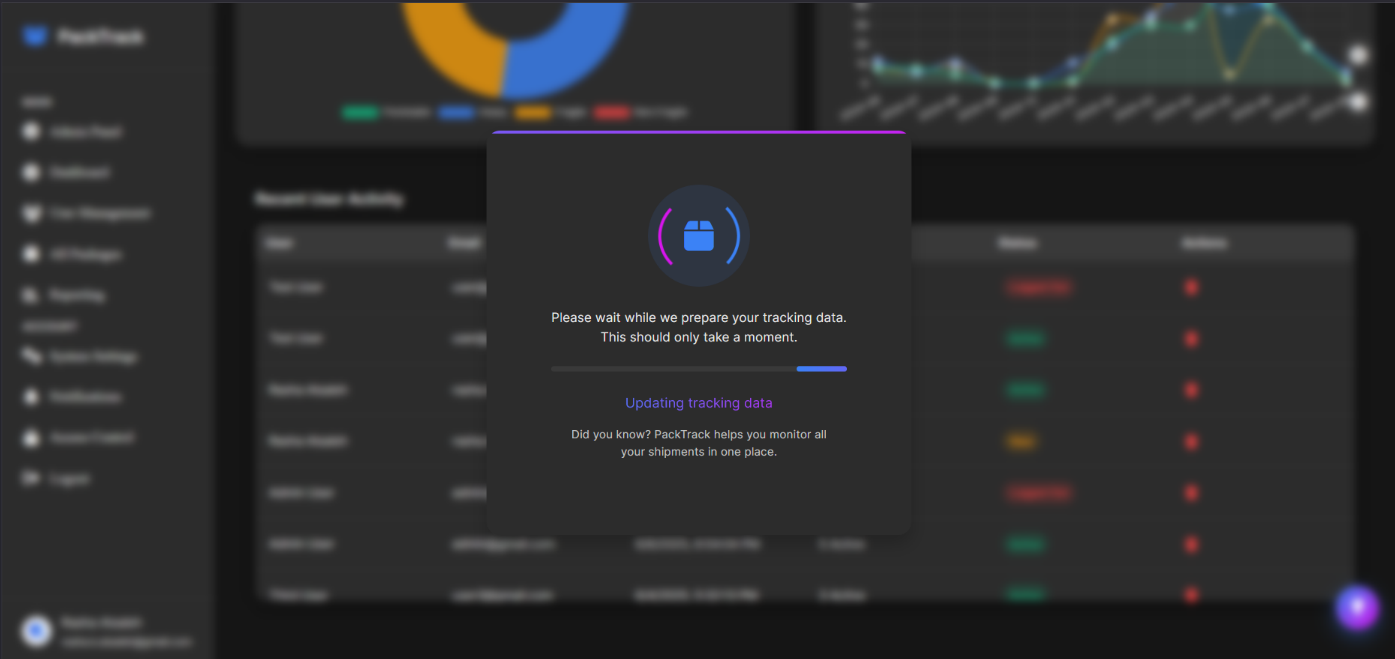


Figure 3.6.9 Loading Animation