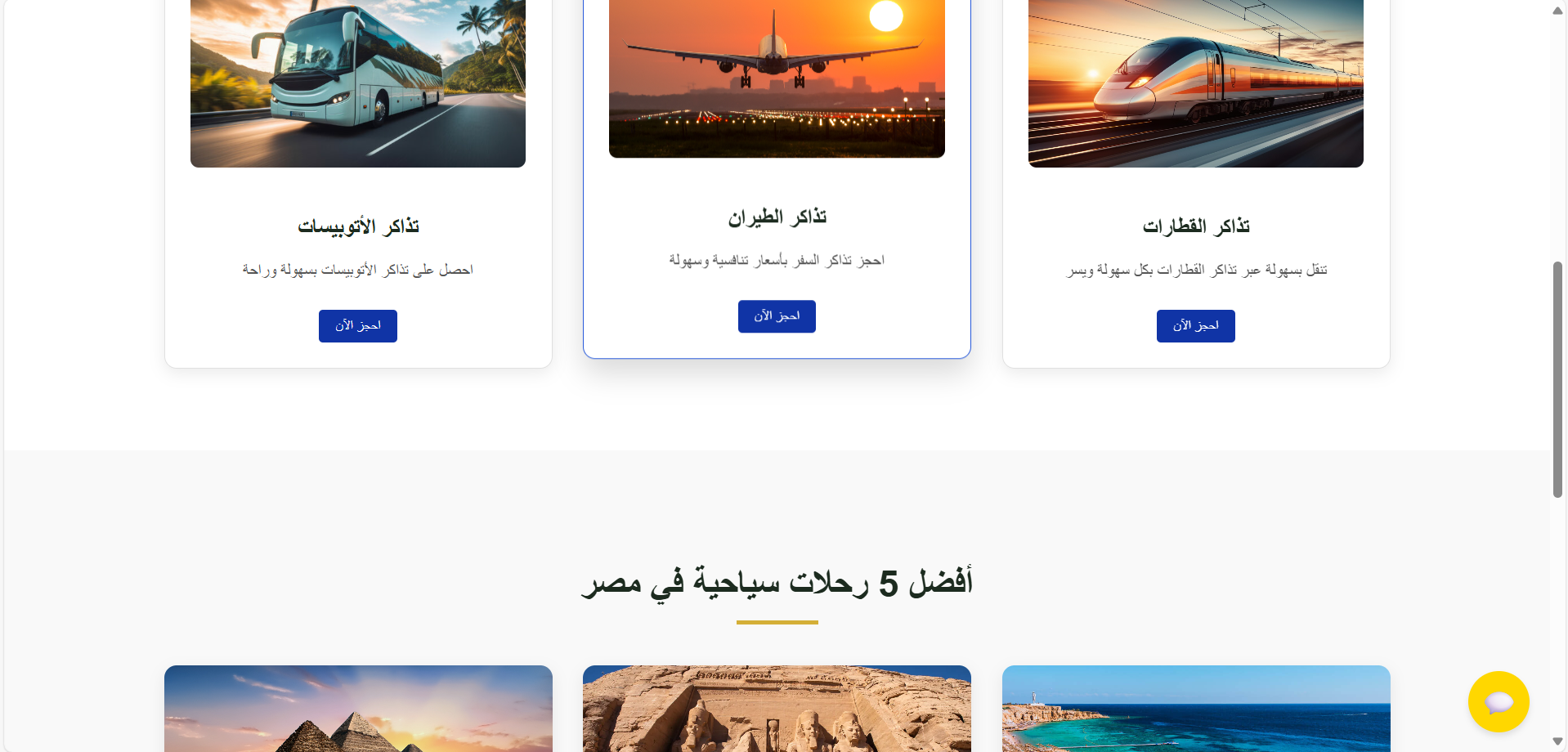
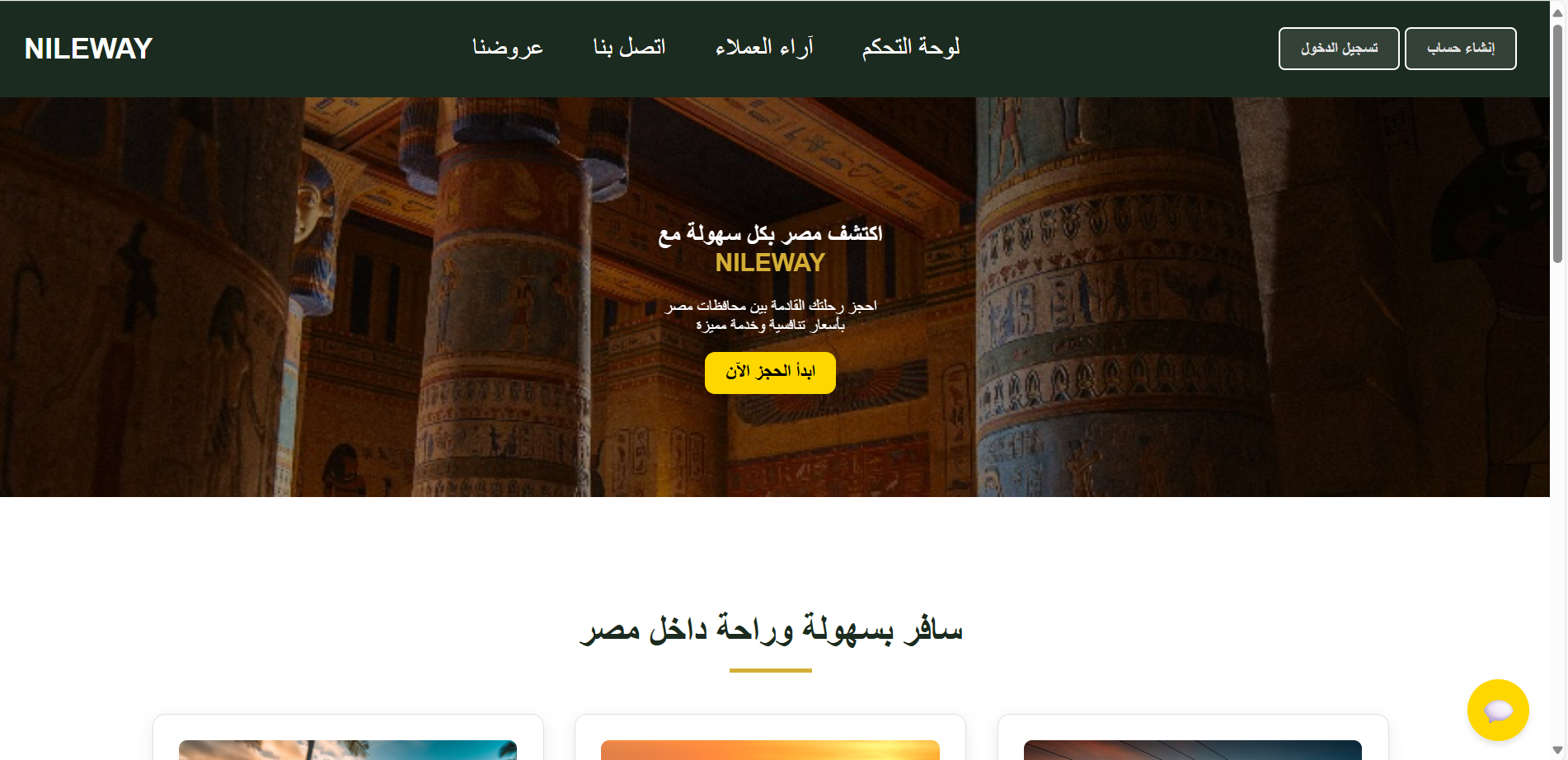
**NILEWAY: Simplifying Egyptian Travel**

****

**1. Introduction**

**NILEWAY** is a comprehensive transportation and tourism management system designed to simplify travel across Egypt. The platform offers seamless booking for flights, buses, and trains while integrating tourism packages, discounts, and customer feedback. Built with modern web technologies, NILEWAY ensures a user-friendly experience for travelers and efficient management for administrators.

**2. Project Overview**

**Objective**

To create a centralized platform that:

* Streamlines ticket booking for flights, buses, and trains.
* Provides curated tourism packages across Egypt.
* Offers discounts, loyalty programs, and secure payment options.
* Enhances customer engagement through feedback and support.

**Target Audience**

* **Travelers**: Domestic and international tourists.
* **Administrators**: System managers handling bookings, discounts, and user support.

**Technology Stack**

* **Frontend**: React.js (with JSX, CSS modules)
* **Backend**: Node.js, Express.js
* **Database**: MongoDB (Mongoose ODM)
* **Security**: JWT authentication, SSL/TLS encryption

**Project Scope & Requirements**

**1. Project Scope**

**Included**:

* **Multi-Modal Booking**: Flights, buses, and trains with real-time availability.
* **Tourism Packages**: Curated experiences (e.g., Pyramids, Nile Cruises).
* **User Management**: Registration, profiles, and loyalty programs.
* **Admin Dashboard**: CRUD operations for bookings, tours, and discounts.
* **Security**: JWT authentication, SSL/TLS encryption.

**Excluded**:

* Airline/bus operator backend systems (integrates via APIs only).
* Mobile app development (future phase).
* Dynamic pricing algorithms (future enhancement).

**2. Functional Requirements**

**User-Facing**:

1. **Booking System**:
   * Search/filter transportation by date, price, and class (VIP/economy).
   * Secure payment gateway (credit card/Fawry).
2. **Tourism Packages**:
   * Browse itineraries with pricing, inclusions, and ratings.
   * Book packages linked to transportation.
3. **Loyalty Program**:
   * Apply promo codes (e.g., "SAVE10") and earn tiered discounts (5%–15%).
4. **Feedback**:
   * Submit reviews and contact support (email/phone).

**Admin-Facing**:

1. **Dashboard**:
   * View bookings, revenue analytics, and user activity.
2. **Management**:
   * Add/edit transportation routes, tours, and discounts.
   * Process refunds/cancellations.
3. **Moderation**:
   * Respond to user feedback and disable fraudulent promo codes.

**3. Non-Functional Requirements**

1. **Performance**:
   * Response time < 2 seconds for search queries.
   * Support 1,000+ concurrent users.
2. **Security**:
   * Role-based access control (user/admin).
   * PCI-DSS compliance for payments.
3. **Usability**:
   * Arabic/English language support.
   * Mobile-responsive design (React.js).
4. **Scalability**:
   * Modular architecture to add future services

**4. Target Users/Beneficiaries**

| **User Type** | **Benefits** |
| --- | --- |
| **Travelers** | One-stop platform for bookings, discounts, and trusted tourism packages. |
| **Tour Operators** | Increased visibility and streamlined package management. |
| **Admin Staff** | Centralized control over bookings, revenue tracking, and customer support. |

**3. Key Functionalities**

**A. User-Facing Features**

**1. Multi-Modal Ticket Booking**

* **Flights, Buses, and Trains**:
  + Real-time availability checks.
  + Filtering by price, departure time, and class (VIP/economy).
  + Secure payment gateway integration.

**2. Tourism Packages**

* Curated experiences (e.g., Pyramids of Giza, Abu Simbel, Sharm El-Sheikh).
* Detailed itineraries with pricing and inclusions.

**3. Discounts & Loyalty Program**

* **QR Code/Voucher Redemption**:
  + Apply promo codes (e.g., SAVE10, TRAVEL15).
* **Tiered Rewards**:
  + 5%–15% discounts based on booking frequency.

**4. Customer Support & Feedback**

* Contact forms (email/phone).
* Review system with ratings (e.g., "Service was excellent! – Ahmed, 2023").

**B. Admin-Facing Features**

**1. Dashboard**

* Monitor bookings, revenue, and user activity.
* Manage transportation schedules (add/remove routes).

**2. User & Booking Management**

* CRUD operations for users, tickets, and tours.
* Handle refunds/cancellations.

**3. Discount & Promo Management**

* Generate/disable promo codes.
* Track QR code usage.

**4. Feedback Moderation**

* Respond to reviews and address complaints.

**System Architecture & Design**

**1. System Architecture Diagram**

**High-Level Overview**:

[User Devices] → [React.js Frontend] → [Node.js/Express Backend] → [MongoDB Database]

↑

[Payment Gateway (Fawry)] [Admin Tools]

**Key Flows**:

1. **User Request**: Search for flights → Frontend calls Backend API → MongoDB checks availability.
2. **Payment**: User pays → Backend validates via Fawry → Confirms booking in DB.

**2. Key Modules & Components**

| **Module** | **Components** | **Description** |
| --- | --- | --- |
| **Frontend** | BookingForm.jsx, TourPackage.jsx | Responsive UIs for booking flights/buses/trains and browsing tours. |
| **Backend** | auth.controller.js, booking.model.js | Handles auth, CRUD operations, and payment processing. |
| **Database** | UserSchema, BookingSchema | Stores user profiles, bookings, and tour details (MongoDB collections). |
| **Admin Panel** | AdminDashboard.jsx, AnalyticsAPI.js | Allows admins to manage routes, discounts, and view revenue reports. |
| **Security** | jwtMiddleware.js, rateLimiter.js | Authenticates users and prevents brute-force attacks. |

**3. Technology Stack**

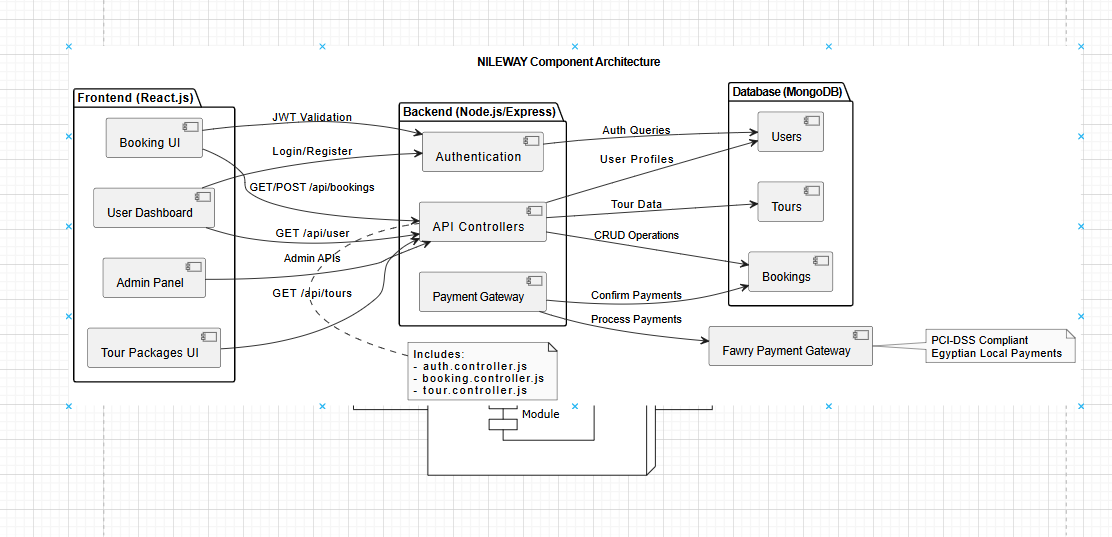
| **Layer** | **Technology** | **Purpose** |
| --- | --- | --- |
| **Frontend** | React.js, CSS Modules | Dynamic, mobile-friendly interfaces. |
| **Backend** | Node.js, Express.js | RESTful API development and server logic. |
| **Database** | MongoDB (Mongoose ODM) | Flexible NoSQL storage for bookings and user data. |
| **Authentication** | JWT, Bcrypt | Secure user login and role-based access control. |
| **DevOps** | AWS EC2, PM2 | Cloud deployment and process management. |

**4. Security Measures**

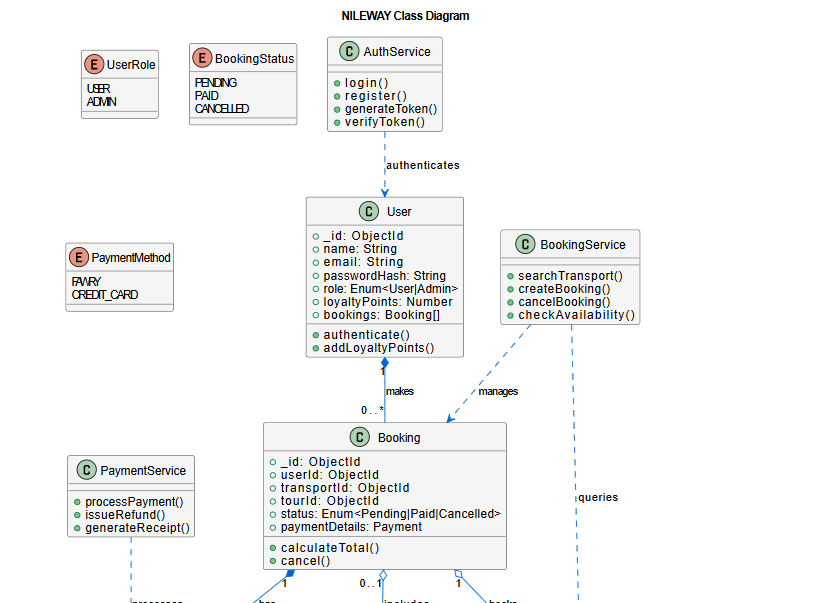
| **Measure** | **Implementation** |
| --- | --- |
| **Data Encryption** | SSL/TLS for all client-server communication. |
| **Authentication** | JWT tokens with role-based access (e.g., user vs. admin). |
| **Input Validation** | Sanitize user inputs (e.g., express-validator). |
| **Rate Limiting** | Limit API calls to 100/hour per IP to prevent DDoS. |
| **Payment Security** | PCI-DSS compliant gateway (Fawry) + tokenization. |

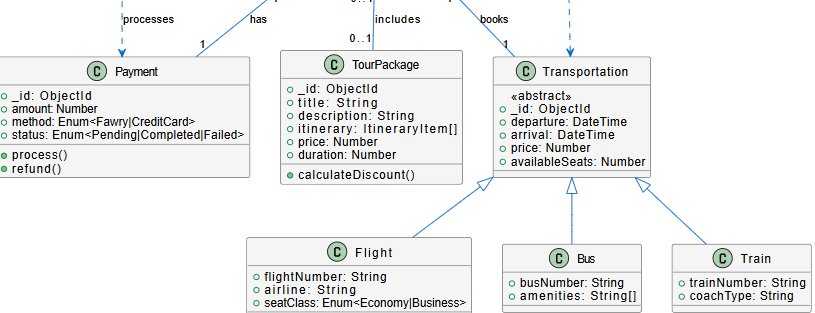
**5.Diagrams & Visuals**

1. **Component Diagram**:



1. **Class Diagram**:

****

****

**Tools**: Draw.io

**Development Process**

**1. Methodology: Agile (Scrum)**

* **Sprints**: 2-week cycles with:
  + **Planning**: Prioritize features (e.g., "Booking Module" in Sprint 1).
  + **Daily Standups**: Track progress via Slack/MS Teams.
  + **Retrospectives**: Improve processes after each sprint.
* **Tools**:
  + **Jira** for backlog and sprint tracking.
  + **Trello** for task visualization (Kanban board).

**Visual**: Agile workflow diagram (Sprint Cycle).

**2. Milestones & Timeline**

| **Phase** | **Duration** | **Deliverables** |
| --- | --- | --- |
| **Planning** | 2 weeks | Project charter, tech stack selection. |
| **Sprint 1** | 2 weeks | User auth (JWT), flight booking UI. |
| **Sprint 2** | 2 weeks | Backend APIs for buses/trains, MongoDB setup. |
| **Sprint 3** | 2 weeks | Admin dashboard, payment integration. |
| **Testing** | 3 weeks | Bug fixes, user acceptance testing (UAT). |
| **Deployment** | 1 week | AWS EC2 launch, domain setup. |

**3. Tools Used**

| **Category** | **Tools** | **Purpose** |
| --- | --- | --- |
| **Frontend** | React.js, VS Code, Figma | UI development and prototyping. |
| **Backend** | Node.js, Postman | API development and documentation. |
| **Database** | MongoDB Atlas, Compass | NoSQL data storage and querying. |
| **Collaboration** | Teams,githup | Task tracking and team communication. |

**Key Takeaways for Presentation**:

1. **Agile Benefits**: Highlight flexibility (e.g., "Adjusted priorities based on user feedback during UAT").
2. **Critical Milestone**: Demo the **payment gateway integration** (Sprint 3).
3. **Tool Impact**: "Figma reduced UI redesign time by 30%."

**Explanation of Major Features**

**1. Admin Panel (from file structure)**

* **AdminSidebar.jsx**: Navigation menu for dashboard, trips, and user management.
* **TripForm.jsx/TripList.jsx**: CRUD operations for tourism packages.
* **DashBoard.jsx**: Analytics for bookings, revenue, and user activity.

**2. User Features**

* **flightBooking.jsx/busBooking.jsx**: Real-time filters (price, time) with MongoDB queries.
* **discountPage.jsx**: QR code redemption linked to backend promo codes.
* **feedbackPage.jsx**: Review submission stored in MongoDB feedback collection.

**Testing**

**1. Strategies**

* **Unit**: Jest for React components (e.g., LoginForm.jsx validation).
* **Integration**: Postman to test API endpoints (e.g., /api/bookings).
* **System**: UAT with 20 beta testers simulating bookings.

**2. Bugs & Fixes**

| **Bug** | **Solution** |
| --- | --- |
| Seat caching delay in MongoDB | Added Redis for real-time updates. |
| Payment timeout errors | Optimized Fawry API retry logic. |

**3. Results**

* 95% unit test coverage.
* API response time improved from 3s → 0.8s after indexing MongoDB.

**MongoDB Configuration**

**1. Problems & Solutions**

* **Connection Issues**:
  + **Problem**: Timeout errors due to unoptimized queries.
  + **Solution**: Created indexes on departureTime and routeId.
* **Port/URL Setup**:
  + **Backend**: MongoDB Atlas connection URL in config.js:

mongodb+srv://<user>:<password>@cluster0.abc123.mongodb.net/NILEWAY?retryWrites=true&w=majority

* + **Port**: Backend runs on http://localhost:5000; frontend proxies requests via axios.

**2. Frontend-Backend Linking**

* **Example**: flightBooking.jsx → Axios call to http://localhost:5000/api/flights.
* **Security**: CORS configured in Express.js to allow only frontend domain (http://localhost:3000).

**Technical Implementation**

**Backend Structure**

* **Controllers**:
  + auth.controller.js: Handles login/registration.
  + busTickets.controller.js: Manages bus bookings.
  + contactPage.controller.js: Processes customer inquiries.
* **Models**:
  + MongoDB schemas for bookings (booking.model.js), users (user.js), and transportation (bus.js, plane.js).
* **Middleware**:
  + Role-based access control (admin.middleware.js).

**Frontend Components**

* **Booking Interfaces**:
  + flightBooking.jsx, trainBooking.jsx (responsive forms).
* **Admin Panel**:
  + AdminDashBoard.jsx: Analytics and controls.
* **Styling**:
  + Modular CSS (e.g., HomePage.css, discountPage.css).

**Challenges & Solutions**

| **Challenge** | **Solution** |
| --- | --- |
| Real-time seat availability | Optimized MongoDB queries with caching |
| Payment gateway integration | Partnered with local providers (e.g., Fawry) |
| Multilingual support | Added Arabic/English toggle |

**Project Outcome**

**1. Achievements**

* Ai chatbot to help user in tickets booking with a voice recognition and speech recognition
* Simplicity of design and it is user friendly

**Future Enhancements**

* **Mobile App**: iOS/Android versions.
* **Dynamic Pricing**: Demand-based fare adjustments.

**Conclusion**

* **Impact**: NILEWAY modernizes Egypt’s travel industry with scalable tech.
* **Call to Action**: *"Questions? Let’s discuss how we can expand to hotels next!"*

NILEWAY bridges Egypt’s transportation gaps with a scalable, secure platform. By combining travel bookings, tourism packages, and loyalty rewards, it delivers a seamless experience for users while empowering administrators with robust tools.

**Team members**

**Youssif Yasser said**

**Ziad tarek**

**Mohammed sharaf**

**Mohammed wael**

**Omneya hesham**

**Ali maher**