AutoRent Pro – Car Rental Management System
Business Requirements Description

1. Introduction

AutoRent Pro is an online car rental management system designed to support the end-to-end process of renting automobiles through a centralized digital platform. The system aims to connect customers who need short-term or long-term vehicle rentals with car owners and rental companies, while providing administrators with full control over system operations.

The platform focuses on automating core business processes such as vehicle management, booking management, rental pricing calculation, and contract tracking, thereby reducing manual work and improving operational efficiency.

2. Business Objectives

The main objectives of AutoRent Pro are:

- To provide customers with a convenient and reliable way to search, book, and rent vehicles online.
- To enable car owners to manage their vehicles, availability schedules, and rental prices.
- To ensure accurate rental cost calculation based on time, vehicle type, and additional fees.
- To maintain rental contracts, booking history, and vehicle status in a centralized system.
- To provide administrators with tools to monitor, manage, and audit all system activities.

3. Stakeholders

The primary stakeholders of the system include:

- **Customers**: Individuals who rent vehicles for personal or business purposes.
- **Car Owners**: Individuals or companies that provide vehicles for rental.
- **System Administrators**: Personnel responsible for managing users, vehicles, bookings, and system policies.

4. User Roles and Responsibilities

4.1 Customer

- Register and authenticate an account.
- View available vehicles based on date and location.
- Create, update, or cancel rental bookings.
- View rental history and booking status.
- Receive rental cost details before confirming a booking.

4.2 Car Owner

- Register and manage vehicle information.
- Define rental pricing and availability schedules.
- View booking requests related to their vehicles.
- Update vehicle status (available, rented, under maintenance).

4.3 Administrator

- Manage user accounts and roles.
- Approve or suspend vehicles and owners.
- Monitor bookings, contracts, and system activities.
- Resolve disputes and handle exceptional cases.

5. Business Processes

5.1 Vehicle Management

Car owners can register vehicles by providing essential information such as brand, model, license plate, rental price, and current status. Vehicles must be approved by an administrator before becoming available for rental.

5.2 Availability Management
Each vehicle has an availability schedule that determines when it can be rented. The system must prevent overlapping bookings for the same vehicle within the same rental period.

5.3 Booking Process
Customers select a vehicle, specify the rental start and end date, and submit a booking request. The system validates vehicle availability, calculates the rental cost, and creates a booking record.

5.4 Rental Pricing
Rental fees are calculated based on rental duration, vehicle price per day or hour, and any additional charges such as late return fees or service fees.

5.5 Contract and Rental Tracking
Once a booking is confirmed, a rental contract is generated. The system tracks rental status from booking to completion and records rental history for future reference.

6. Business Rules
- A vehicle cannot be booked by more than one customer during the same time period.
- Rental cost must be calculated and displayed before booking confirmation.
- Users must be authenticated to create or manage bookings.
- Only administrators can approve or deactivate vehicles.
- Booking status must follow a defined lifecycle (e.g., Pending, Confirmed, Active, Completed, Cancelled).

7. Functional Requirements
The system shall:
- Allow users to register, log in, and manage profiles.
- Allow car owners to add, update, and remove vehicle information.
- Allow customers to search and book vehicles.
- Automatically calculate rental fees based on business rules.
- Store booking, contract, and vehicle status data in a database.
- Provide administrative tools for system monitoring and management.

8. Non-Functional Requirements
- **Scalability**: The system should support a growing number of users and vehicles.
- **Performance**: Vehicle search and booking operations should respond within acceptable time limits.
- **Security**: User data and booking information must be protected through authentication and authorization mechanisms.
- **Availability**: The system should be accessible at all times except during scheduled maintenance.
- **Maintainability**: The system should be modular and easy to extend or modify.

9. Assumptions and Constraints
- Online payment processing may be simulated or integrated in later phases.
- The system operates under a centralized database architecture.
- Internet access is required to use the platform.
- Legal and insurance responsibilities are handled outside the system scope.

10. Conclusion
AutoRent Pro provides a comprehensive solution for managing car rental operations through a modern backend-driven architecture. By clearly defining business roles, processes, and rules, the system establishes a solid foundation for further technical design and implementation.

CHAPTER 1 – HIỂU BÀI TOÁN & NGHIỆP VỤ

Bài 1: Mô tả hệ thống AutoRent Pro

Yêu cầu

- Viết mô tả hệ thống cho thuê xe AutoRent Pro bằng tiếng Anh
- Trả lời các câu hỏi:
    - Hệ thống dùng để làm gì?
    - Có những ai sử dụng hệ thống?
    - Hệ thống hỗ trợ những chức năng chính nào?

Không cần code.

Nộp: File Chapter1_Business_Overview.pdf (1–2 trang)

Bài 2: Mô tả quy trình đặt xe

Yêu cầu

- Mô tả từng bước khi **khách hàng đặt xe**, ví dụ:
    1. Khách xem danh sách xe
    2. Chọn thời gian thuê
    3. Hệ thống kiểm tra xe còn trống
    4. Tính tiền
    5. Tạo booking
- Ghi rõ **trường hợp lỗi** (xe bị trùng lịch, chưa đăng nhập…)

Nộp: File Chapter1_Booking_Process.md

Bài 3: Xác định quyền của từng loại người dùng

Yêu cầu

- Lập bảng đơn giản:
    - Customer làm được gì?
    - Car Owner làm được gì?
    - Admin làm được gì?

Ví dụ:

Customer → book car, cancel booking
Admin → approve car, view all bookings

Nộp: File Chapter1_User_Roles.md

Bài 4: Trạng thái của xe và booking

Yêu cầu

- Liệt kê:
    - Các trạng thái của **Booking** (Pending, Confirmed, Active…)
    - Các trạng thái của **Car** (Available, Rented, Maintenance)
- Mô tả ngắn gọn khi nào chuyển trạng thái

Nộp: File Chapter1_Status_Definition.md

CHAPTER 2 – LOGIC CỐT LÕI BẰNG NODEJS

Bài 1: Tính tiền thuê xe

Yêu cầu

- Viết hàm JavaScript: calculateRentalCost(startDate, endDate, pricePerDay)
- Hàm phải:
  - Tính số ngày thuê
  - Tính tổng tiền

Chạy bằng NodeJS (console).

Nộp: File pricing.js

Bài 2: Kiểm tra trùng lịch thuê

Yêu cầu

- Viết hàm kiểm tra:
  - Một xe có bị đặt trùng thời gian hay không
- Hàm trả về:
  - true nếu trùng
  - false nếu không trùng

Nộp: File availability.js

Bài 3: Kiểm tra dữ liệu booking hợp lệ

Yêu cầu

- Viết hàm kiểm tra dữ liệu booking:
  - endDate phải sau startDate
  - carId không được rỗng
- Nếu sai → báo lỗi

Nộp: File bookingValidator.js

Bài 4: Demo bằng console

Yêu cầu

- Tạo file demo.js
- Chạy thử:
  - nhập thời gian thuê
  - in ra kết quả: có trùng lịch không + tổng tiền

Nộp: demo.js

CHAPTER 3 – THIẾT KẾ API

Bài 1: Thiết kế API cho hệ thống

Yêu cầu

- Thiết kế API cho:
  - User
  - Car
  - Booking
- Mỗi API ghi rõ:
  - Method (GET/POST/…)
  - URL
  - Chức năng

Nộp: File Chapter3_API_List.md

Bài 2: Thiết kế dữ liệu JSON

Yêu cầu

- Thiết kế JSON cho:
  - User
  - Car
  - Booking
- Ghi rõ các field cần có

Nộp: File Chapter3_JSON_Model.md

Bài 3: Quy định lỗi và mã trạng thái

Yêu cầu

- Xác định:
  - Khi nào trả 400?
  - Khi nào trả 401?
  - Khi nào trả 409 (trùng lịch)?

Nộp: File Chapter3_Status_Codes.md

Bài 4: Test case cho API

Yêu cầu

- Viết ít nhất 10 test case:
  - booking thành công
  - booking trùng lịch
  - booking khi chưa login

Nộp: File Chapter3_Test_Cases.md

CHAPTER 4 – NODEJS HTTP SERVER

Bài 1: Tạo HTTP server

Yêu cầu

- Dùng http module
- Tạo API:
  - GET /cars
  - POST /bookings

Nộp: server.js

Bài 2: Gắn logic đã viết

Yêu cầu

- Khi POST /bookings:
  - kiểm tra trùng lịch
  - tính tiền thuê

Nộp: server.js (update)

Bài 3: Lưu booking vào file

Yêu cầu

- Lưu booking vào bookings.json
- Khi gọi API → đọc lại file

Nộp: data/bookings.json

Bài 4: Lọc dữ liệu

Yêu cầu

- Cho phép:
  - /cars?status=AVAILABLE
  - /bookings?userId=...

CHAPTER 5 – EXPRESSJS

Bài 1: Chuyển HTTP server sang ExpressJS

Yêu cầu

- Tạo project ExpressJS
- Chuyển các API đã có ở Chapter 4 sang Express:
  - GET /api/v1/cars
  - POST /api/v1/bookings
- API phải trả JSON giống trước

Không yêu cầu database, vẫn dùng file JSON

Nộp:

- Project Express chạy được
- File app.js hoặc server.js

Bài 2: Tách route cho từng chức năng

Yêu cầu

- Tách route thành các file:
  - routes/cars.js
  - routes/bookings.js
- app.js chỉ dùng để cấu hình Express và gắn route

Mục tiêu: code dễ đọc, dễ mở rộng

Nộp: Thư mục routes/

Bài 3: Middleware kiểm tra người dùng (mock)

Yêu cầu

- Tạo middleware kiểm tra header x-user-id
- Nếu không có header → trả lỗi 401
- Áp dụng middleware cho:
  - POST /bookings

Chưa cần login thật

Nộp: File middlewares/auth.js

Bài 4: Chuẩn hóa lỗi API

Yêu cầu

- Khi có lỗi (thiếu dữ liệu, trùng lịch…)
- API trả về JSON dạng:

```
{
  "error": {
    "message": "Booking time overlaps",
    "code": "BOOKING_CONFLICT"
  }
}
```

Nộp: API trả lỗi đúng format

CHAPTER 6 – MONGODB

Bài 1: Thiết kế collections MongoDB

Yêu cầu

- Thiết kế các collection:
  - users
  - cars
  - bookings
- Mỗi collection có:
  - _id
  - các field chính theo business

Không cần code NodeJS

Nộp: File Chapter6_Mongo_Design.md

Bài 2: Tạo dữ liệu mẫu trong MongoDB

Yêu cầu

- Insert:
  - 5 xe
  - 3 user
  - 6 booking
- Dùng MongoDB Compass hoặc shell

Nộp: Screenshot dữ liệu hoặc file lệnh insert

Bài 3: Truy vấn dữ liệu cơ bản

Yêu cầu

- Viết truy vấn:
  - lấy xe đang AVAILABLE
  - lấy booking theo userId
  - lấy booking theo carId

Nộp: File Chapter6_Mongo_Queries.md

Bài 4: Chuẩn bị thay thế file JSON

Yêu cầu

- Liệt kê:
  - API nào đang dùng file JSON
  - API nào sẽ chuyển sang MongoDB ở Chapter 7

Nộp: File Chapter6_Migration_Plan.md

CHAPTER 7 – NODEJS + MONGODB

Bài 1: Kết nối Express với MongoDB

Yêu cầu

- Cài MongoDB NodeJS Driver
- Tạo file db.js để kết nối DB
- Khi server chạy → kết nối thành công

Nộp:

- File db.js
- Server chạy không lỗi

Bài 2: API lấy danh sách xe từ MongoDB

Yêu cầu

- Thay GET /cars:
  - không đọc file JSON
  - đọc dữ liệu từ collection cars

Nộp: API /cars trả dữ liệu từ DB

Bài 3: API tạo booking với MongoDB

Yêu cầu

- Khi POST /bookings:
  - kiểm tra trùng lịch trong DB
  - nếu không trùng → lưu booking vào MongoDB

Nộp: API hoạt động đúng với DB

Bài 4: Tạo contract khi booking được xác nhận

Yêu cầu

- Khi booking có status CONFIRMED
- Tạo 1 record trong collection contracts

Nộp:

- Collection contracts
- Code xử lý tạo contract

CHAPTER 8 – MONGOOSE ODM

Bài 1: Tạo Mongoose models

Yêu cầu

- Tạo schema:
  - User
  - Car
  - Booking
  - Contract
- Có required field và enum status

Nộp: Thư mục models/

Bài 2: Validation dữ liệu bằng schema

Yêu cầu

- endDate phải sau startDate
- status chỉ nhận giá trị hợp lệ

Nộp: Validation hoạt động (test bằng Postman)

Bài 3: Tự động tính tiền khi tạo booking

Yêu cầu

- Khi tạo booking:
  - gọi lại hàm tính tiền (Chapter 2)
  - lưu totalPrice vào booking

Nộp: Booking có field totalPrice

Bài 4: Refactor code theo MVC

Yêu cầu

- Tách:
  - models
  - controllers
  - services
  - routes

Nộp: Cấu trúc project rõ ràng

CHAPTER 9 – MONGOOSE POPULATION

Bài 1: API xem chi tiết booking

Yêu cầu

- GET /bookings/:id
- Trả:
    - thông tin booking
    - thông tin user
    - thông tin car

Nộp: JSON đầy đủ (populate)

Bài 2: API lịch sử thuê xe của user

Yêu cầu

- GET /users/:id/bookings
- Trả danh sách booking + car info

Nộp: API hoạt động đúng

Bài 3: API cho car owner

Yêu cầu

- Owner xem booking liên quan xe của mình
- Chỉ trả booking của xe thuộc owner đó

Nộp: API /owner/bookings

Bài 4: API cho admin

Yêu cầu

- Admin xem:
    - tổng số booking
    - booking theo trạng thái

Nộp: API /admin/bookings/summary

CHAPTER 10 – AUTHENTICATION & SECURITY

Bài 1: Đăng ký và đăng nhập

Yêu cầu

- API:
  - POST /auth/register
  - POST /auth/login
- Mật khẩu phải được mã hóa

Nộp: Auth API hoạt động

Bài 2: Bảo vệ API booking

Yêu cầu

- Chỉ user đã login mới:
  - tạo booking
  - hủy booking
  - xem lịch sử

Nộp: Middleware auth

Bài 3: Phân quyền theo vai trò

Yêu cầu

- Customer: chỉ xem booking của mình
- Owner: quản lý xe của mình
- Admin: duyệt xe, xem toàn hệ thống

Nộp: Middleware authorize(role)

Bài 4: Hoàn thiện project

Yêu cầu

- Kiểm tra lại toàn bộ:
  - API
  - business rules
  - security
- Viết README hướng dẫn chạy project

Nộp:

- Source code
- README.md
- Postman collection