Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Кафедра програмної інженерії

Лабораторна робота №2

з дисципліни: «Архітектура програмного забезпечення»

на тему: «РОЗРОБКА СЕРВЕРНОЇ ЧАСТИНИ ПРОГРАМНОЇ СИСТЕМИ»

Виконав

студент групи ПЗПІ-22-2

Верясов Владислав Олексійович

28 квітня 2025 р.

Перевірив

Старший викладач кафедри ПІ  
Сокорчук Ігор Петрович

Харків 2025

**МЕТА**

Метою цієї лабораторної роботи є розробити  розробити серверну/бек-енд частину програмної системи для підтримки та регуляції прокату електротранспорту.

**ЗАВДАННЯ**

Робота передбачає створення будови програмної системи з описом її компонентів, UML діаграми прецедентів для серверної частини, ER діаграми даних, бази даних (БД) системи та її структури, функцій роботи з БД (ORM або CoRM), API (REST, GraphQL, gRPC) для клієнт-серверної взаємодії, а також специфікації API. Необхідно реалізувати програмний код API та функцій роботи з БД, перевірити його функціональність, завантажити код у GitHub репозиторій, створити відеодемонстрацію перевірки роботи серверної частини, завантажити відео на YouTube, додати хронологічний опис до відео, створити звіт, вказати у звіті посилання на відео, експортувати звіт у PDF для завантаження на dl.nure.ua та у текстовий файл (UTF-8) для запису в GitHub.

**ХІД РОБОТИ**

Назва: «Система для підтримки та регуляції прокату електротранспорту»

Назва англійською мовою: «Electric Transport Rental Management System»

Власна назва: «E-Transport».

На першому етапі було створено ER-діаграму даних, яка визначила структуру бази даних та її взаємозв’язки. Це стало основою для подальшого створення бази даних і забезпечило логічну організацію даних. ER-діаграма представлена у додатку А.

На другому етапі було розроблено базу даних на основі створеної ER-діаграми. Для цього використовувалася PostgreSQL як основна СУБД, а адміністрування бази виконувалося через PgAdmin4. Усі необхідні таблиці, зв’язки та обмеження було реалізовано.

На третьому етапі було створено діаграму структури бази даних, яка наочно демонструє зв’язки між таблицями та їх ключові атрибути. Це допомогло краще зрозуміти загальну архітектуру даних. Діаграма структури бази даних наведена у додатку Б.

На четвертому етапі було реалізовано кодування бази даних за допомогою ORM Prisma. Це спростило взаємодію з базою даних і забезпечило типізацію даних для backend-частини. Схема бази даних наведена у додатку Б.

На п’ятому етапі було створено UML-діаграму прецедентів для серверної частини системи. Вона визначила основні сценарії використання API клієнтами та основні функції системи. UML-діаграму наведено у додатку В.

На шостому етапі, під час розробки API, було обрано реалізацію REST API з використанням Swagger для документування. Це забезпечило зручний інтерфейс для тестування та ознайомлення з ендпоїнтами API, дозволило автоматично генерувати документацію та значно полегшило взаємодію з клієнтами системи. Для цього було використано NestJS з мовою програмування TypeScript. Усі функції взаємодії з БД інтегрувалися через Prisma.

На сьомому етапі було створено специфікацію розробленого REST API з використанням Swagger, яка детально описує доступні ендпоїнти, методи, параметри запитів і відповіді. Логіку взаємодії серверної частини з БД наведено у додатку Д. Логіку взаємодії серверної частини з клієнтами наведено у додатку Д. Специфікацію розробленого REST API наведено у додатку Е.

На восьмому етапі було проведено тестування розробленого програмного коду серверної частини системи, включно з API та функціями роботи з БД, для перевірки їхньої працездатності та відповідності вимогам.

На дев’ятому етапі весь програмний код було завантажено в GitHub репозиторій у гілку main для забезпечення централізованого зберігання та доступу до нього.

На десятому етапі було створено відеозапис демонстрації перевірки функціональності серверної частини системи. Тривалість запису склала 7-9 хвилин, а відео завантажено на YouTube з українськими субтитрами.

На одинадцятому етапі було створено хронологічний опис демонстрації відео (хвилина:секунда) і додано цей опис до опису відео на YouTube.

На дванадцятому етапі було створено звіт до лабораторної роботи, у якому вказано посилання на відеозапис на YouTube та описано основні етапи виконання завдання.

На тринадцятому етапі звіт було експортовано у формат PDF і завантажено на платформу dl.nure.ua, а також у текстовий файл з кодуванням UTF-8, який було записано до GitHub репозиторію.

У серверній частині було реалізовано наступне:

Аунтефікація в системі через JWT токени

Шифрування паролів на бекенд частині

Доступи до функцій системи відповідно до ролі

Методи Get, Post, Put, Delete для усіх таблиць

Логіку оренди та її завершення зі зміною стану самоката та його батареї, платежом при закінченні поїздки.

Отримання персоналізованої інформації для користувачів, такої як:

- усі поїздки

- усі платежі

- найближчі вільні самокати

- вільні самокати

Отримання персоналізованої інформації для адміністраторів, такої як:

- інформація по конкретному самокату

- середній час використання самокату

- усі оренди самокату

- усі користувачі та персонал системи

- усі оренди та платежі користувача

- усі самокати та їх стан

- найприбутковіші зони та їх дохід

Отримання персоналізованої інформації для технічного персоналу, такої як:

- інформація та стан самокату

- інформація про батареї

- інформація про середню швидкість

- інформація про енергозатрати

Резервне копіювання користувацьких даних

Середній час оренди

кількість поїздок на одиницю транспорту за одиницю часу

Розподіл активності за годинами/днями

Відстеження простоїв транспорту (час між орендами)

Найпопулярніші зони старту/фінішу

Розрахунок ефективності використання батарей

**ВИСНОВОК**

У ході виконання роботи було створено програмну систему, яка включає серверну частину з функціональним REST API, інтегрованим з базою даних через ORM Prisma. Було виконано повний цикл розробки: від проєктування архітектури системи та створення діаграм (ER, UML) до програмної реалізації та тестування. Завдяки цьому вдалося побудувати ефективну та зрозумілу систему для взаємодії клієнтів із сервером.

Для розробки серверної частини використовувалася мова програмування TypeScript у фреймворку NestJS. У ролі системи управління базами даних було використано PostgreSQL, адміністровану через PgAdmin4, а для ORM – Prisma, яка забезпечила зручну взаємодію з базою даних і строгий контроль типів. Документацію для REST API було реалізовано через Swagger, що зробило API доступним для тестування та інтеграції.

Виконання лабораторної роботи сприяло набуттю практичних навичок у створенні програмних систем, проєктуванні архітектури серверної частини та баз даних, а також реалізації REST API з використанням сучасних інструментів. Я навчився ефективно інтегрувати різні технології, такі як Docker для контейнеризації системи, Prisma для роботи з базами даних та Swagger для документування API.

Ця лабораторна робота навчила мене організовувати процес розробки від планування до завершення, включаючи тестування та підготовку звітності. Крім того, я набув досвіду використання GitHub для управління кодом і публікації результатів, а також створення відеодемонстрацій, що допомогло розвинути як технічні, так і презентаційні навички.

Відео на YouTube: https://youtu.be/12y1tB1Qbo4

**ДОДАТОК А**

ER-діаграма

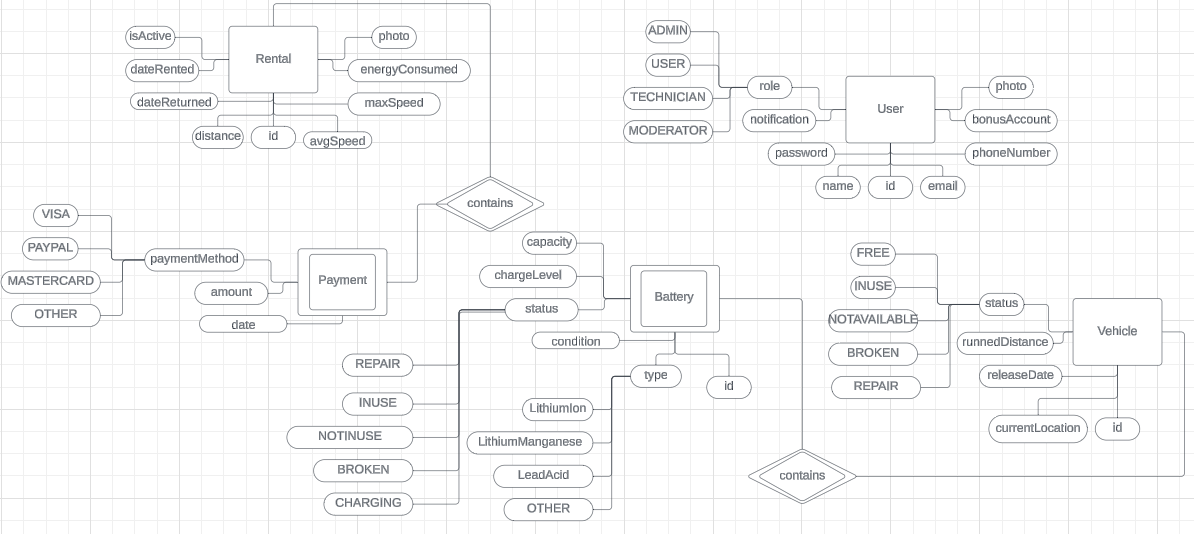


Рисунок 1- ER-діаграма (нотація Чена)

**ДОДАТОК Б**

Схема бази даних

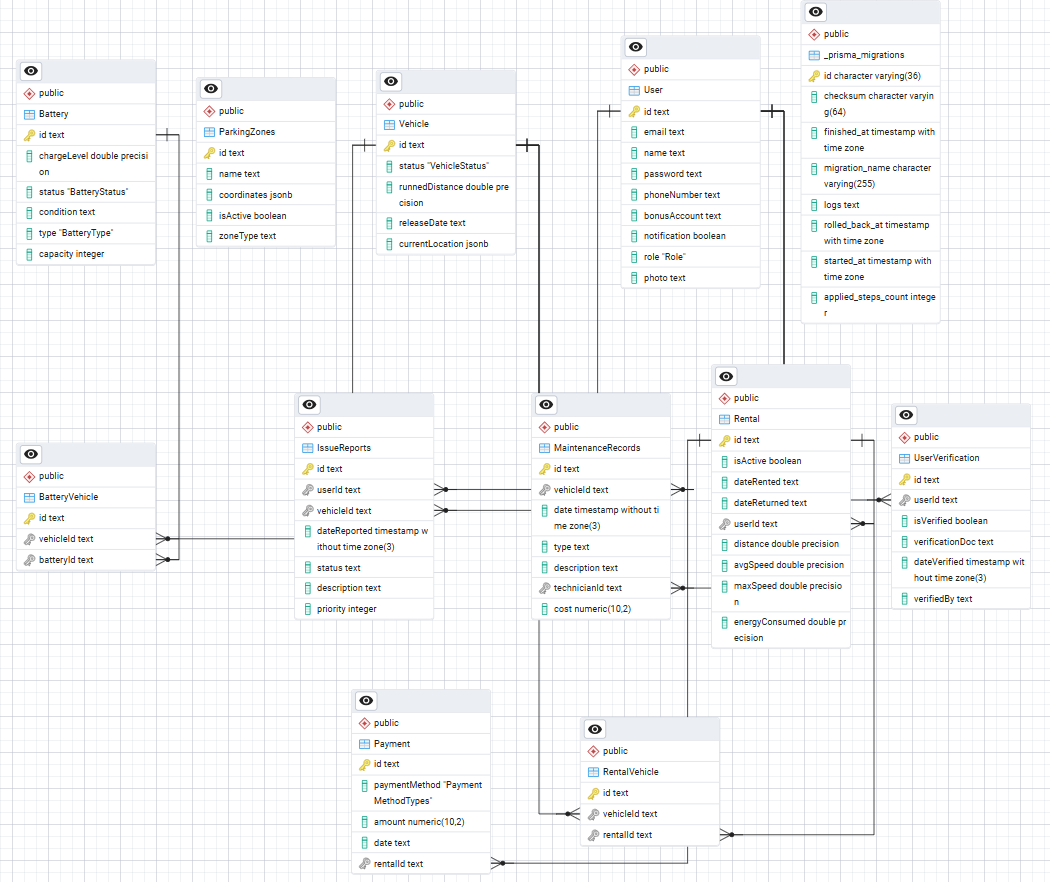


Рисунок 2- Схема бази даних

**ДОДАТОК В**

UML-діаграма прецедентів серверної частини

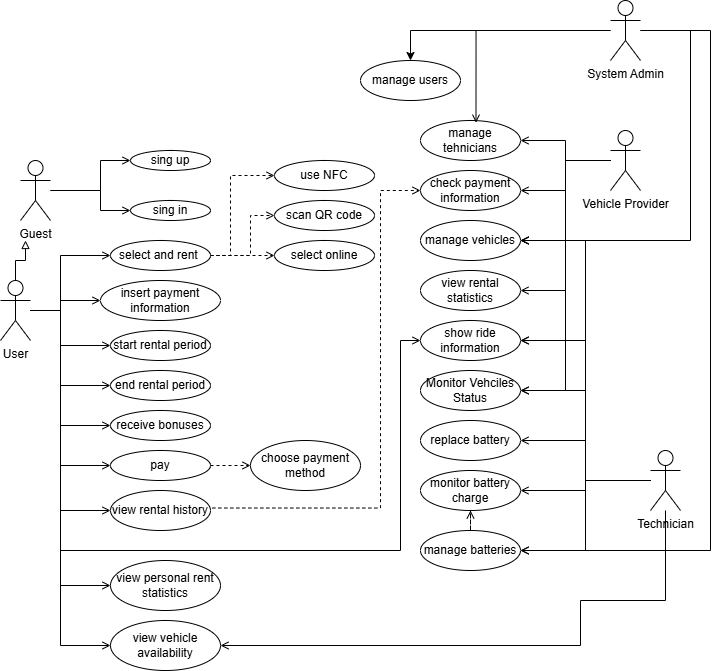


Рисунок 3 - UML-діаграма прецедентів серверної частини

**ДОДАТОК Г**

Логіка взаємодії серверної частини з БД (приклад розробленого сервісу для таблиці User)

import { Injectable } from '@nestjs/common';

import { PrismaService } from 'src/prisma/prisma.service';

import { UserDto } from './dtos/user.dto';

import { CreateUserDto } from './dtos/create-user.dto';

import { UpdateUserDto } from './dtos/update-user.dto';

@Injectable()

export class UserService {

    constructor(private prisma: PrismaService) { }

    async findAllUser(): Promise<UserDto[]> {

        const result = await this.prisma.user.findMany();

        return result;

    }

    async createUser(user: UserDto): Promise<CreateUserDto> {

        const result = await this.prisma.user.create({ data: { ...user } });

        return result;

    }

    async updateUser(user: UpdateUserDto, email: string): Promise<CreateUserDto> {

        const result = await this.prisma.user.update({ data: { ...user }, where: { email: email } });

        return result;

    }

    async deleteUser(id: string): Promise<CreateUserDto> {

        const result = await this.prisma.user.delete({ where: { id: id } });

        return result;

    }

    async getUserPlus(id: string): Promise<CreateUserDto> {

        const result = await this.prisma.user.findUnique({

            where: { id: id },

            include: {

              rental: {

                include: {

                  rentalVehicle: { include: { vehicle: true } },

                  payment: true,

                },

              },

            },

         });

        return result;

    }

}

**ДОДАТОК Д**

Логіка взаємодії серверної частини з клієнтами (приклад розробленого контролера для таблиці User)

import { Body, Controller, Delete, Get, HttpCode, Param, Post, Put } from '@nestjs/common';

import { ApiResponse, ApiTags } from '@nestjs/swagger';

import { UserService } from './user.service';

import { UserDto } from './dtos/user.dto';

import { UpdateUserDto } from './dtos/update-user.dto';

@ApiTags('user')

@Controller('user')

export class UserController {

    constructor(private readonly userService: UserService) { }

    @Get()

    @HttpCode(200)

    @ApiResponse({ status: 200, description: 'List of all users returned successfully.' })

    @ApiResponse({ status: 500, description: 'Internal server error.' })

    public async findAllUser() {

        const result = await this.userService.findAllUser();

        return result;

    }

    @Post()

    @HttpCode(201)

    @ApiResponse({ status: 201, description: 'User created successfully.' })

    @ApiResponse({ status: 400, description: 'Invalid input data.' })

    @ApiResponse({ status: 500, description: 'Internal server error.' })

    public async createUser(@Body() user: UserDto) {

        const result = await this.userService.createUser(user);

        return result;

    }

    @Put(':id')

    @HttpCode(200)

    @ApiResponse({ status: 200, description: 'User updated successfully.' })

    @ApiResponse({ status: 404, description: 'User not found.' })

    @ApiResponse({ status: 500, description: 'Internal server error.' })

    public async updateUser(@Body() user: UpdateUserDto, @Param('id') id: string) {

        const result = await this.userService.updateUser(user, id);

        return result;

    }

    @Delete(':id')

    @HttpCode(200)

    @ApiResponse({ status: 200, description: 'User deleted successfully.' })

    @ApiResponse({ status: 404, description: 'User not found.' })

    @ApiResponse({ status: 500, description: 'Internal server error.' })

    public async deleteUser(@Param('id') id: string) {

        const result = await this.userService.deleteUser(id);

        return result;

    }

    @Get(':id')

    @HttpCode(200)

    @ApiResponse({ status: 200, description: 'User details returned successfully.' })

    @HttpCode(404)

    @ApiResponse({ status: 404, description: 'User not found.' })

    @HttpCode(500)

    @ApiResponse({ status: 500, description: 'Internal server error.' })

    public async getUserDetails(@Param('id') userId: string): Promise<UserDto> {

        return this.userService.getUserPlus(userId);

    }

}

**ДОДАТОК Е**

Специфікація API

{

    "openapi": "3.0.0",

    "paths": {

        "/": {

            "get": {

                "operationId": "AppController\_getHello",

                "parameters": [],

                "responses": {

                    "200": {

                        "description": ""

                    }

                },

                "tags": [

                    "App"

                ]

            }

        },

        "/user": {

            "get": {

                "operationId": "UserController\_findAllUser",

                "parameters": [],

                "responses": {

                    "200": {

                        "description": "List of all users returned successfully."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "user"

                ]

            },

            "post": {

                "operationId": "UserController\_createUser",

                "parameters": [],

                "requestBody": {

                    "required": true,

                    "content": {

                        "application/json": {

                            "schema": {

                                "$ref": "#/components/schemas/UserDto"

                            }

                        }

                    }

                },

                "responses": {

                    "201": {

                        "description": "User created successfully."

                    },

                    "400": {

                        "description": "Invalid input data."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "user"

                ]

            }

        },

        "/user/{id}": {

            "put": {

                "operationId": "UserController\_updateUser",

                "parameters": [

                    {

                        "name": "id",

                        "required": true,

                        "in": "path",

                        "schema": {

                            "type": "string"

                        }

                    }

                ],

                "requestBody": {

                    "required": true,

                    "content": {

                        "application/json": {

                            "schema": {

                                "$ref": "#/components/schemas/UpdateUserDto"

                            }

                        }

                    }

                },

                "responses": {

                    "200": {

                        "description": "User updated successfully."

                    },

                    "404": {

                        "description": "User not found."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "user"

                ]

            },

            "delete": {

                "operationId": "UserController\_deleteUser",

                "parameters": [

                    {

                        "name": "id",

                        "required": true,

                        "in": "path",

                        "schema": {

                            "type": "string"

                        }

                    }

                ],

                "responses": {

                    "200": {

                        "description": "User deleted successfully."

                    },

                    "404": {

                        "description": "User not found."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "user"

                ]

            },

            "get": {

                "operationId": "UserController\_getUserDetails",

                "parameters": [

                    {

                        "name": "id",

                        "required": true,

                        "in": "path",

                        "schema": {

                            "type": "string"

                        }

                    }

                ],

                "responses": {

                    "200": {

                        "description": "User details returned successfully."

                    },

                    "404": {

                        "description": "User not found."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "user"

                ]

            }

        },

        "/rental": {

            "get": {

                "operationId": "RentalController\_findAllRental",

                "parameters": [],

                "responses": {

                    "200": {

                        "description": "List of all rentals returned successfully."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "rental"

                ]

            },

            "post": {

                "operationId": "RentalController\_createRental",

                "parameters": [],

                "requestBody": {

                    "required": true,

                    "content": {

                        "application/json": {

                            "schema": {

                                "$ref": "#/components/schemas/RentalDto"

                            }

                        }

                    }

                },

                "responses": {

                    "201": {

                        "description": "Rental created successfully."

                    },

                    "400": {

                        "description": "Invalid input data."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "rental"

                ]

            }

        },

        "/rental/{id}": {

            "put": {

                "operationId": "RentalController\_updateRental",

                "parameters": [

                    {

                        "name": "id",

                        "required": true,

                        "in": "path",

                        "schema": {

                            "type": "string"

                        }

                    }

                ],

                "requestBody": {

                    "required": true,

                    "content": {

                        "application/json": {

                            "schema": {

                                "$ref": "#/components/schemas/UpdateRentalDto"

                            }

                        }

                    }

                },

                "responses": {

                    "200": {

                        "description": "Rental updated successfully."

                    },

                    "404": {

                        "description": "Rental not found."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "rental"

                ]

            },

            "delete": {

                "operationId": "RentalController\_deleteRental",

                "parameters": [

                    {

                        "name": "id",

                        "required": true,

                        "in": "path",

                        "schema": {

                            "type": "string"

                        }

                    }

                ],

                "responses": {

                    "200": {

                        "description": "Rental deleted successfully."

                    },

                    "404": {

                        "description": "Rental not found."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "rental"

                ]

            }

        },

        "/payment": {

            "get": {

                "operationId": "PaymentController\_findAllPayment",

                "parameters": [],

                "responses": {

                    "200": {

                        "description": "List of all payments returned successfully."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "payment"

                ]

            },

            "post": {

                "operationId": "PaymentController\_createPayment",

                "parameters": [],

                "requestBody": {

                    "required": true,

                    "content": {

                        "application/json": {

                            "schema": {

                                "$ref": "#/components/schemas/PaymentDto"

                            }

                        }

                    }

                },

                "responses": {

                    "201": {

                        "description": "Payment created successfully."

                    },

                    "400": {

                        "description": "Invalid input data."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "payment"

                ]

            }

        },

        "/payment/{id}": {

            "put": {

                "operationId": "PaymentController\_updatePayment",

                "parameters": [

                    {

                        "name": "id",

                        "required": true,

                        "in": "path",

                        "schema": {

                            "type": "string"

                        }

                    }

                ],

                "requestBody": {

                    "required": true,

                    "content": {

                        "application/json": {

                            "schema": {

                                "$ref": "#/components/schemas/UpdatePaymentDto"

                            }

                        }

                    }

                },

                "responses": {

                    "200": {

                        "description": "Payment updated successfully."

                    },

                    "404": {

                        "description": "Payment not found."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "payment"

                ]

            },

            "delete": {

                "operationId": "PaymentController\_deletePayment",

                "parameters": [

                    {

                        "name": "id",

                        "required": true,

                        "in": "path",

                        "schema": {

                            "type": "string"

                        }

                    }

                ],

                "responses": {

                    "200": {

                        "description": "Payment deleted successfully."

                    },

                    "404": {

                        "description": "Payment not found."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "payment"

                ]

            }

        },

        "/vehicle": {

            "get": {

                "operationId": "VehicleController\_findAllVehicle",

                "parameters": [],

                "responses": {

                    "200": {

                        "description": "List of all vehicles returned successfully."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "vehicle"

                ]

            },

            "post": {

                "operationId": "VehicleController\_createVehicle",

                "parameters": [],

                "requestBody": {

                    "required": true,

                    "content": {

                        "application/json": {

                            "schema": {

                                "$ref": "#/components/schemas/VehicleDto"

                            }

                        }

                    }

                },

                "responses": {

                    "201": {

                        "description": "Vehicle created successfully."

                    },

                    "400": {

                        "description": "Invalid input data."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "vehicle"

                ]

            }

        },

        "/vehicle/{id}": {

            "put": {

                "operationId": "VehicleController\_updateVehicle",

                "parameters": [

                    {

                        "name": "id",

                        "required": true,

                        "in": "path",

                        "schema": {

                            "type": "string"

                        }

                    }

                ],

                "requestBody": {

                    "required": true,

                    "content": {

                        "application/json": {

                            "schema": {

                                "$ref": "#/components/schemas/UpdateVehicleDto"

                            }

                        }

                    }

                },

                "responses": {

                    "200": {

                        "description": "Vehicle updated successfully."

                    },

                    "404": {

                        "description": "Vehicle not found."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "vehicle"

                ]

            },

            "delete": {

                "operationId": "VehicleController\_deleteVehicle",

                "parameters": [

                    {

                        "name": "id",

                        "required": true,

                        "in": "path",

                        "schema": {

                            "type": "string"

                        }

                    }

                ],

                "responses": {

                    "200": {

                        "description": "Vehicle deleted successfully."

                    },

                    "404": {

                        "description": "Vehicle not found."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "vehicle"

                ]

            }

        },

        "/battery": {

            "get": {

                "operationId": "BatteryController\_findAllBattery",

                "parameters": [],

                "responses": {

                    "200": {

                        "description": "List of all batteries returned successfully."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "battery"

                ]

            },

            "post": {

                "operationId": "BatteryController\_createBattery",

                "parameters": [],

                "requestBody": {

                    "required": true,

                    "content": {

                        "application/json": {

                            "schema": {

                                "$ref": "#/components/schemas/BatteryDto"

                            }

                        }

                    }

                },

                "responses": {

                    "201": {

                        "description": "Battery created successfully."

                    },

                    "400": {

                        "description": "Invalid input data."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "battery"

                ]

            }

        },

        "/battery/{id}": {

            "put": {

                "operationId": "BatteryController\_updateBattery",

                "parameters": [

                    {

                        "name": "id",

                        "required": true,

                        "in": "path",

                        "schema": {

                            "type": "string"

                        }

                    }

                ],

                "requestBody": {

                    "required": true,

                    "content": {

                        "application/json": {

                            "schema": {

                                "$ref": "#/components/schemas/UpdateBatteryDto"

                            }

                        }

                    }

                },

                "responses": {

                    "200": {

                        "description": "Battery updated successfully."

                    },

                    "404": {

                        "description": "Battery not found."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "battery"

                ]

            },

            "delete": {

                "operationId": "BatteryController\_deleteBattery",

                "parameters": [

                    {

                        "name": "id",

                        "required": true,

                        "in": "path",

                        "schema": {

                            "type": "string"

                        }

                    }

                ],

                "responses": {

                    "200": {

                        "description": "Battery deleted successfully."

                    },

                    "404": {

                        "description": "Battery not found."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "battery"

                ]

            }

        },

        "/rental-vehicle": {

            "get": {

                "operationId": "RentalVehicleController\_findAllRentalVehicle",

                "parameters": [],

                "responses": {

                    "200": {

                        "description": "List of all rental vehicles returned successfully."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "rental-vehicle"

                ]

            },

            "post": {

                "operationId": "RentalVehicleController\_createRentalVehicle",

                "parameters": [],

                "requestBody": {

                    "required": true,

                    "content": {

                        "application/json": {

                            "schema": {

                                "$ref": "#/components/schemas/RentalVehicleDto"

                            }

                        }

                    }

                },

                "responses": {

                    "201": {

                        "description": "Rental vehicle created successfully."

                    },

                    "400": {

                        "description": "Invalid input data."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "rental-vehicle"

                ]

            }

        },

        "/rental-vehicle/{id}": {

            "put": {

                "operationId": "RentalVehicleController\_updateRentalVehicle",

                "parameters": [

                    {

                        "name": "id",

                        "required": true,

                        "in": "path",

                        "schema": {

                            "type": "string"

                        }

                    }

                ],

                "requestBody": {

                    "required": true,

                    "content": {

                        "application/json": {

                            "schema": {

                                "$ref": "#/components/schemas/UpdateRentalVehicleDto"

                            }

                        }

                    }

                },

                "responses": {

                    "200": {

                        "description": "Rental vehicle updated successfully."

                    },

                    "404": {

                        "description": "Rental vehicle not found."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "rental-vehicle"

                ]

            },

            "delete": {

                "operationId": "RentalVehicleController\_deleteRentalVehicle",

                "parameters": [

                    {

                        "name": "id",

                        "required": true,

                        "in": "path",

                        "schema": {

                            "type": "string"

                        }

                    }

                ],

                "responses": {

                    "200": {

                        "description": "Rental vehicle deleted successfully."

                    },

                    "404": {

                        "description": "Rental vehicle not found."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "rental-vehicle"

                ]

            }

        },

        "/battery-vehicle": {

            "get": {

                "operationId": "BatteryVehicleController\_findAllBatteryVehicle",

                "parameters": [],

                "responses": {

                    "200": {

                        "description": "List of all battery vehicles returned successfully."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "battery-vehicle"

                ]

            },

            "post": {

                "operationId": "BatteryVehicleController\_createBatteryVehicle",

                "parameters": [],

                "requestBody": {

                    "required": true,

                    "content": {

                        "application/json": {

                            "schema": {

                                "$ref": "#/components/schemas/BatteryVehicleDto"

                            }

                        }

                    }

                },

                "responses": {

                    "201": {

                        "description": "Battery vehicle created successfully."

                    },

                    "400": {

                        "description": "Invalid input data."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "battery-vehicle"

                ]

            }

        },

        "/battery-vehicle/{id}": {

            "put": {

                "operationId": "BatteryVehicleController\_updateBatteryVehicle",

                "parameters": [

                    {

                        "name": "id",

                        "required": true,

                        "in": "path",

                        "schema": {

                            "type": "string"

                        }

                    }

                ],

                "requestBody": {

                    "required": true,

                    "content": {

                        "application/json": {

                            "schema": {

                                "$ref": "#/components/schemas/UpdateBatteryVehicleDto"

                            }

                        }

                    }

                },

                "responses": {

                    "200": {

                        "description": "Battery vehicle updated successfully."

                    },

                    "404": {

                        "description": "Battery vehicle not found."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "battery-vehicle"

                ]

            },

            "delete": {

                "operationId": "BatteryVehicleController\_deleteBatteryVehicle",

                "parameters": [

                    {

                        "name": "id",

                        "required": true,

                        "in": "path",

                        "schema": {

                            "type": "string"

                        }

                    }

                ],

                "responses": {

                    "200": {

                        "description": "Battery vehicle deleted successfully."

                    },

                    "404": {

                        "description": "Battery vehicle not found."

                    },

                    "500": {

                        "description": "Internal server error."

                    }

                },

                "tags": [

                    "battery-vehicle"

                ]

            }

        }

    },

    "info": {

        "title": "e-transport",

        "description": "The e-transport API description",

        "version": "1.0",

        "contact": {}

    },

    "tags": [],

    "servers": [],

    "components": {

        "schemas": {

            "UserDto": {

                "type": "object",

                "properties": {

                    "email": {

                        "type": "string"

                    },

                    "name": {

                        "type": "string"

                    },

                    "password": {

                        "type": "string"

                    },

                    "phoneNumber": {

                        "type": "string"

                    },

                    "bonusAccount": {

                        "type": "string"

                    },

                    "notification": {

                        "type": "boolean"

                    },

                    "photo": {

                        "type": "string"

                    },

                    "role": {

                        "type": "string",

                        "enum": [

                            "ADMIN",

                            "USER",

                            "MODERATOR",

                            "TECHNICIAN"

                        ]

                    }

                },

                "required": [

                    "email",

                    "name",

                    "password",

                    "phoneNumber",

                    "bonusAccount",

                    "notification",

                    "photo",

                    "role"

                ]

            },

            "UpdateUserDto": {

                "type": "object",

                "properties": {

                    "email": {

                        "type": "string"

                    },

                    "name": {

                        "type": "string"

                    },

                    "password": {

                        "type": "string"

                    },

                    "phoneNumber": {

                        "type": "string"

                    },

                    "bonusAccount": {

                        "type": "string"

                    },

                    "photo": {

                        "type": "string"

                    },

                    "notification": {

                        "type": "boolean"

                    },

                    "role": {

                        "type": "string",

                        "enum": [

                            "ADMIN",

                            "USER",

                            "MODERATOR",

                            "TECHNICIAN"

                        ]

                    }

                },

                "required": [

                    "role"

                ]

            },

            "RentalDto": {

                "type": "object",

                "properties": {

                    "isActive": {

                        "type": "boolean"

                    },

                    "dateRented": {

                        "type": "string"

                    },

                    "dateReturned": {

                        "type": "string"

                    },

                    "userId": {

                        "type": "string"

                    },

                    "distance": {

                        "type": "number"

                    },

                    "avgSpeed": {

                        "type": "number"

                    },

                    "maxSpeed": {

                        "type": "number"

                    },

                    "energyConsumed": {

                        "type": "number"

                    }

                },

                "required": [

                    "isActive",

                    "dateRented",

                    "dateReturned",

                    "userId",

                    "distance",

                    "avgSpeed",

                    "maxSpeed",

                    "energyConsumed"

                ]

            },

            "UpdateRentalDto": {

                "type": "object",

                "properties": {

                    "isActive": {

                        "type": "boolean"

                    },

                    "dateRented": {

                        "type": "string"

                    },

                    "dateReturned": {

                        "type": "string"

                    },

                    "userId": {

                        "type": "string"

                    },

                    "distance": {

                        "type": "number"

                    },

                    "avgSpeed": {

                        "type": "number"

                    },

                    "maxSpeed": {

                        "type": "number"

                    },

                    "energyConsumed": {

                        "type": "number"

                    }

                }

            },

            "PaymentDto": {

                "type": "object",

                "properties": {

                    "paymentMethod": {

                        "type": "string",

                        "enum": [

                            "VISA",

                            "PAYPAL",

                            "MASTERCARD",

                            "OTHER"

                        ]

                    },

                    "amount": {

                        "type": "string"

                    },

                    "date": {

                        "type": "string"

                    },

                    "rentalId": {

                        "type": "string"

                    }

                },

                "required": [

                    "paymentMethod",

                    "amount",

                    "date",

                    "rentalId"

                ]

            },

            "UpdatePaymentDto": {

                "type": "object",

                "properties": {

                    "paymentMethod": {

                        "type": "string",

                        "enum": [

                            "VISA",

                            "PAYPAL",

                            "MASTERCARD",

                            "OTHER"

                        ]

                    },

                    "name": {

                        "type": "string"

                    },

                    "amount": {

                        "type": "string"

                    },

                    "date": {

                        "type": "string"

                    },

                    "rentalId": {

                        "type": "string"

                    }

                },

                "required": [

                    "paymentMethod"

                ]

            },

            "VehicleDto": {

                "type": "object",

                "properties": {

                    "status": {

                        "type": "string",

                        "enum": [

                            "FREE",

                            "INUSE",

                            "NOTAVAILABLE",

                            "BROKEN",

                            "REPAIR"

                        ]

                    },

                    "runnedDistance": {

                        "type": "number"

                    },

                    "releaseDate": {

                        "type": "string"

                    },

                    "currentLocation": {

                        "type": "string"

                    }

                },

                "required": [

                    "status",

                    "runnedDistance",

                    "releaseDate",

                    "currentLocation"

                ]

            },

            "UpdateVehicleDto": {

                "type": "object",

                "properties": {

                    "status": {

                        "type": "string",

                        "enum": [

                            "FREE",

                            "INUSE",

                            "NOTAVAILABLE",

                            "BROKEN",

                            "REPAIR"

                        ]

                    },

                    "runnedDistance": {

                        "type": "number"

                    },

                    "releaseDate": {

                        "type": "string"

                    },

                    "currentLocation": {

                        "type": "string"

                    }

                },

                "required": [

                    "status"

                ]

            },

            "BatteryDto": {

                "type": "object",

                "properties": {

                    "status": {

                        "type": "string",

                        "enum": [

                            "INUSE",

                            "NOTINUSE",

                            "BROKEN",

                            "CHARGING",

                            "REPAIR"

                        ]

                    },

                    "chargeLevel": {

                        "type": "number"

                    },

                    "condition": {

                        "type": "string"

                    },

                    "type": {

                        "type": "string",

                        "enum": [

                            "LithiumIon",

                            "LithiumManganese",

                            "LeadAcid"

                        ]

                    },

                    "capacity": {

                        "type": "number"

                    }

                },

                "required": [

                    "status",

                    "chargeLevel",

                    "condition",

                    "type",

                    "capacity"

                ]

            },

            "UpdateBatteryDto": {

                "type": "object",

                "properties": {

                    "chargeLevel": {

                        "type": "number"

                    },

                    "status": {

                        "type": "string",

                        "enum": [

                            "INUSE",

                            "NOTINUSE",

                            "BROKEN",

                            "CHARGING",

                            "REPAIR"

                        ]

                    },

                    "condition": {

                        "type": "string"

                    },

                    "type": {

                        "type": "string",

                        "enum": [

                            "LithiumIon",

                            "LithiumManganese",

                            "LeadAcid"

                        ]

                    },

                    "capacity": {

                        "type": "number"

                    }

                },

                "required": [

                    "status",

                    "type"

                ]

            },

            "RentalVehicleDto": {

                "type": "object",

                "properties": {

                    "vehicleId": {

                        "type": "string"

                    },

                    "rentalId": {

                        "type": "string"

                    }

                },

                "required": [

                    "vehicleId",

                    "rentalId"

                ]

            },

            "UpdateRentalVehicleDto": {

                "type": "object",

                "properties": {

                    "vehicleId": {

                        "type": "string"

                    },

                    "rentalId": {

                        "type": "string"

                    }

                }

            },

            "BatteryVehicleDto": {

                "type": "object",

                "properties": {

                    "vehicleId": {

                        "type": "string"

                    },

                    "batteryId": {

                        "type": "string"

                    }

                },

                "required": [

                    "vehicleId",

                    "batteryId"

                ]

            },

            "UpdateBatteryVehicleDto": {

                "type": "object",

                "properties": {

                    "vehicleId": {

                        "type": "string"

                    },

                    "batteryId": {

                        "type": "string"

                    }

                }

            }

        }

    }

}