**Autóbérlési Weboldal Projektdokumentáció**

# 1. Bevezetés

## 1.1 Projekt célja

Ez a projekt egy autóbérlési weboldalt és a hozzá tartozó API-t tartalmazza. A cél egy reszponzív és felhasználóbarát webalkalmazás létrehozása, amely lehetővé teszi a felhasználók számára a bérelhető járművek keresését, szűrését és megtekintését. Az API biztosítja az adatok kezelését és az adatbázis működését.

## 1.2 Alkalmazás tervezése

Az alkalmazás két fő komponensből áll:

* **Frontend**: HTML, CSS, JavaScript segítségével készült reszponzív felület.
* **Backend**: ASP.NET Core alapú API, amely az adatbázist kezeli és kiszolgálja a frontend kéréseit.

# 2. Fejlesztői Dokumentáció

## 2.1 Technológiai stack

* **Frontend**: HTML, CSS, JavaScript
* **Backend**: ASP.NET Core
* **Adatbázis**: MariaDB
* **Adatbázis kezelő**: Entity Framework Core
* **API**: RESTful
* **Konténerizálás (tervezett)**: Docker

## 2.2 Adatbázis struktúra

Az adatbázis az alábbi főbb táblákat tartalmazza:

### Beállítások

* **Szerver**: localhost
* **Port**: 3306
* **Felhasználónév**: api\_user
* **Jelszó**: admin
* **Adatbázis neve**: rent\_a\_car
* **Jogosultságok**: teljes hozzáférés

### Táblák

#### Cars

* Id: int
* ModelId: int
* FuelTypeId: int
* CategoryId: int
* Km: int
* PricePerKm: int
* Year: int
* HorsePower: int
* Description: varchar(400)

#### Models

* Id: int
* ManufacturerId: int
* Name: longtext

#### Manufacturers

* Id: int
* Name: varchar(50)

#### Categories

* Id: int
* Name: varchar(30)

#### Fuel\_Types

* Id: int
* Name: varchar(30)

## 2.3 API Endpointok

### Cars

* **Összes autó lekérése**: GET /cars
* **Autó id alapján való lekérdezése**: GET /cars/{id}
* **Autó hozzáadása**: POST /cars
* **Autó frissítése**: PUT /cars/{id}
* **Autó törlése**: DELETE /cars/{id}

### Models

* **Összes modell lekérése**: GET /models
* **Modell id alapján való lekérdezése**: GET /models/{id}
* **Modell hozzáadása**: POST /models
* **Modell frissítése**: PUT /models/{id}
* **Modell törlése**: DELETE /models/{id}

### Categories

* **Összes kategória lekérése**: GET /categories
* **Kategória id alapján való lekérdezése**: GET /categories/{id}
* **Kategória hozzáadása**: POST /categories
* **Kategória frissítése**: PUT /categories/{id}
* **Kategória törlése**: DELETE /categories/{id}

### Manufacturers és FuelTypes

* **Gyártók**: /manufacturers
* **Üzemanyag típusok**: /fuel\_types

# 3. Telepítési útmutató

## 3.1 Frontend telepítése

1. Klónozd a repót: git clone https://github.com/Bandus8/RentACarWeboldal
2. Nyisd meg a projektet egy böngészőben.

## 3.2 Backend telepítése

1. Klónozd a repót: git clone https://github.com/PlatypusHE22/RentACarApi
2. Telepítsd a függőségeket: dotnet restore
3. Indítsd el a szervert: dotnet run

# 4. Használati útmutató

* A felhasználó beléphet a weboldalra, kereshet autókat, szűrhet márka, évjárat vagy üzemanyag típus szerint.
* A frontend az API-ból szerzi be az adatokat JSON formátumban.

# 5. Jövőbeli fejlesztési lehetőségek

* Felhasználói regisztráció és bejelentkezés
* Foglalási rendszer
* Fizetési integráció
* Teljes konténerizálás Docker segítségével

# 6. Tesztelési dokumentáció

## 6.1 Tesztelés célja

A rendszer fő funkcióinak ellenőrzése, különös tekintettel az API megfelelő válaszaira és az adatbázis helyes kezelésére.

## 6.2 Tesztelési módszerek

* **Funkcionális tesztelés**: Ellenőrizni az API végpontok helyes működését.
* **Adatbázis tesztelés**: Biztosítani, hogy a lekérdezések megfelelően működnek.
* **Teljesítmény tesztelés**: Az oldal válaszidejének mérése különböző terhelések mellett.

# 7. Köszönetnyilvánítás

* **Fejlesztői csapat**: Azok, akik hozzájárultak a projekt fejlesztéséhez.
* **Oktatók és mentorok**: Akik támogatást nyújtottak a projekt során.
* **Felhasználók**: Akik teszteléssel és visszajelzéseikkel segítettek.

Ez a dokumentáció a projekt jelenlegi állapotát tükrözi, és a fejlesztés előrehaladtával frissíteni fogjuk.