***DOCUMENTATIE RENT A CAR***

*Stanciu Darius-Gabriel*

*Sîrbu Cosmin-Claudiu*

*Grupa 30231(2)*

Introducere

Această documentație descrie arhitectura și funcționalitățile unei aplicații web dezvoltate în Node.js, folosind framework-ul Express și ORM-ul Sequelize. Aplicația gestionează două entități principale: mașini (Car) și închirieri (Rental), oferind o interfață RESTful pentru operațiuni CRUD (Create, Read, Update, Delete).

Arhitectura Aplicației

Model-View-Controller (MVC)

Aplicația adoptă modelul MVC, un pattern de design arhitectural care separă aplicația în trei componente principale:

Model: Reprezentat prin entitățile Car și Rental, gestionând structura datelor și operațiunile cu baza de date.

View: În contextul unui API RESTful, view-urile sunt reprezentate de răspunsurile JSON trimise înapoi la client.

Controller: Modulele carController și rentalController manipulează logica de afaceri și coordonează fluxul de date între model și view.

Structura Codului

Codul este structurat modular, cu fiecare entitate și funcționalitate împărțită în fișiere și foldere specifice. Aceasta include modele pentru baza de date, controllere pentru logica de afaceri și rute pentru gestionarea cererilor HTTP.

Tehnologii Folosite

Node.js: Platforma de bază pentru server.

Express: Framework-ul web folosit pentru a construi API-ul.

Sequelize: ORM utilizat pentru a interacționa cu baza de date.

Nodemailer: Bibliotecă pentru trimiterea de email-uri, configurată cu Mailtrap pentru testare.

Funcționalități

Gestionarea Mașinilor

Listare Mașini: Utilizatorii pot vizualiza toate mașinile disponibile.

Detalii Mașină: Vizualizarea detaliată a unei mașini folosind ID-ul său.

Creare Mașină: Adăugarea unei noi mașini în sistem.

Ștergere Mașină: Eliminarea unei mașini din sistem.

Gestionarea Închirierilor

Listare Închirieri: Utilizatorii pot vedea toate închirierile efectuate.

Detalii Închiriere: Accesul la informații despre o anumită închiriere.

Creare Închiriere: Inițierea unei noi închirieri, cu trimiterea de email-uri de confirmare.

Actualizare Închiriere: Modificarea detaliilor unei închirieri.

Ștergere Închiriere: Anularea unei înch

irieri existente.

Securitatea Aplicației

Securitatea este o preocupare majoră, iar aplicația adoptă diverse măsuri pentru a proteja datele și interacțiunile utilizatorilor:

Helmet: Utilizat pentru a adăuga antete de securitate HTTP pentru a proteja împotriva unor vulnerabilități comune.

CORS (Cross-Origin Resource Sharing): Permite cererile între domenii diferite, cu posibilitatea de a restricționa și controla accesul.

Validarea Datelor: Asigură că datele de intrare sunt verificate pentru a preveni injectarea de cod și alte atacuri.

Interacțiunea cu Baza de Date

Sequelize: Permite modelarea entităților și gestionarea relațiilor dintre acestea. Operațiunile CRUD sunt simplificate prin intermediul metodelor Sequelize.

Tranzacții: Pentru a asigura consistența datelor, în special în cazul operațiunilor compuse, aplicația utilizează tranzacții Sequelize.

Trimiterea de Email-uri

Nodemailer cu Mailtrap: Folosit pentru testarea funcționalității de email într-un mediu de dezvoltare fără a trimite email-uri reale.

Rutele API

Aplicația expune un API RESTful prin care clienții pot interacționa cu resursele:

Rute pentru Mașini (/cars): Include rute pentru listarea, crearea, actualizarea și ștergerea mașinilor.

Rute pentru Închirieri (/rentals): Permite utilizatorilor să listeze, să creeze, să actualizeze și să șteargă închirieri.

Concluzie

Această aplicație Node.js cu Express și Sequelize reprezintă un exemplu solid de arhitectură MVC, oferind funcționalități robuste și o structură clară. Prin utilizarea tehnologiilor și practicilor moderne de dezvoltare, aplicația este bine poziționată pentru a fi extinsă și scalată, asigurând în același timp securitatea și eficiența.