

Slovenská technická univerzita v Bratislave  
Fakulta informatiky a informačných technológií

## **Mobilná aplikácia - Autoservis**

Mobilné technológie a aplikácie

**Autori:** Matej Lánik, Jakub Sorád

**Cvičenia:** Štvrtok 8:00

**Prednášajúci:** doc. Ing. Peter Trúchly, PhD.

**Akademický rok:** 2021/22

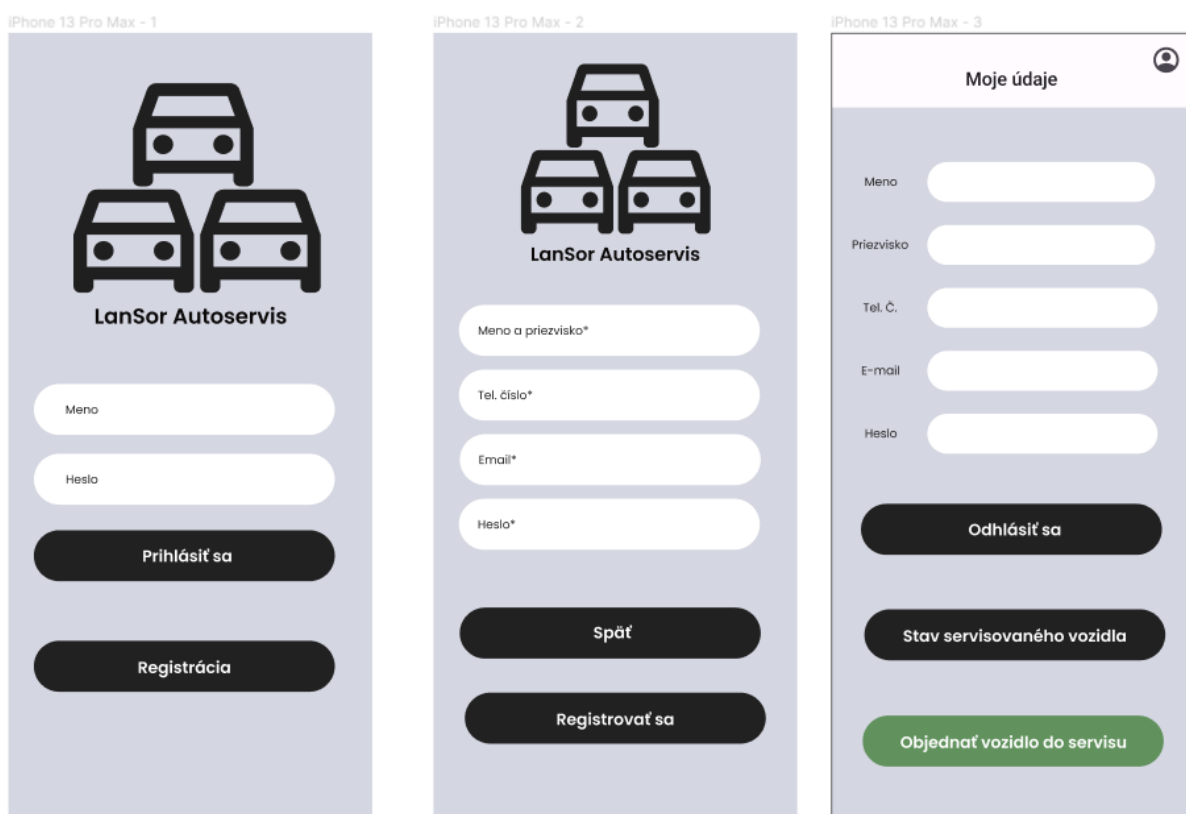
# Obsah

<b>Návrh wireframov</b>	<b>2</b>
<b>Návrh databázy</b>	<b>5</b>
<b>Návrh API endpointov</b>	<b>6</b>
<b>Akceptačné testy</b>	<b>7</b>
Frontend aplikácie	7
Kladné testy	7
Záporné testy	10
Backend	12
Kladné testy	12
Záporné testy	14
<b>Backend aplikácie</b>	<b>15</b>
Backend technológie	15
Štruktúra backendu a jeho funkcionalita	15
Testovanie backendu	16
Change log oproti Milestone 1	17

# Návrh wireframov

Použili sme nástroj [Figma](#), kde sme vytvorili jednotlivé wireframy a tiež aj user flows, takže náš návrh je aj interaktívny. Po konzultácií sme zapracovali na pripomienkach a možných zlepšeniach, medzi ne patria aj tieto dve:

- Pri scenári, keď sa prihlasuje technik sme použili inú úvodnú obrazovku, ale dopredu nevieme, kto sa bude prihlasovať, či zákazník alebo technik. Preto bude platiť pre oboch typov používateľov jednotná prihlasovacia obrazovka.
- Keď technik opraví auto, potvrdí to v aplikácii a následne aj zákazník potvrdí opravu auta, tak sme vymazali dané auto z databázy. V našej aplikácii by bolo veľmi vhodné si práve tieto opravené autá ukladať do histórie opravených áut. Túto históriu si potom technik môže pozrieť po prihlásení do aplikácie.



iPhone 13 Pro Max - 4

Objednať vozidlo do servisu

☐ Želám si aby servis môjho vozidla vykonával konkrétny technik.

Späť

iPhone 13 Pro Max - /

Zvoľte požadované úkony

☒ Výmena motorového oleja

☒ Výmena filtrov

☒ Výmena pneumatík

☒ Servis motora

Späť

Nahrať fotografie vozidla

iPhone 13 Pro Max - 8

Nahrať fotografiu vozidla

Späť

Objednať vozidlo do servisu

iPhone 13 Pro Max - 11

Stav vozidla

Servis vozidla bol dokončený

Potvrdiť vyzdvihnutie vozidla


iPhone 13 Pro Max - 6

Stav vozidla

Začať videohovor s technikom

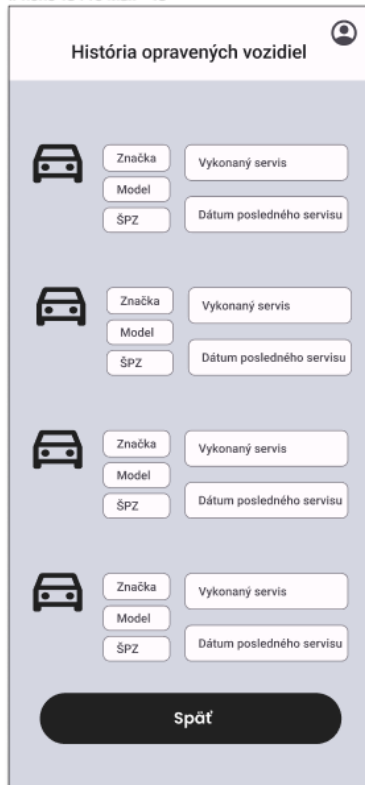
Pokračovať

login

  
LanSor Autoservis

Prihlásiť sa

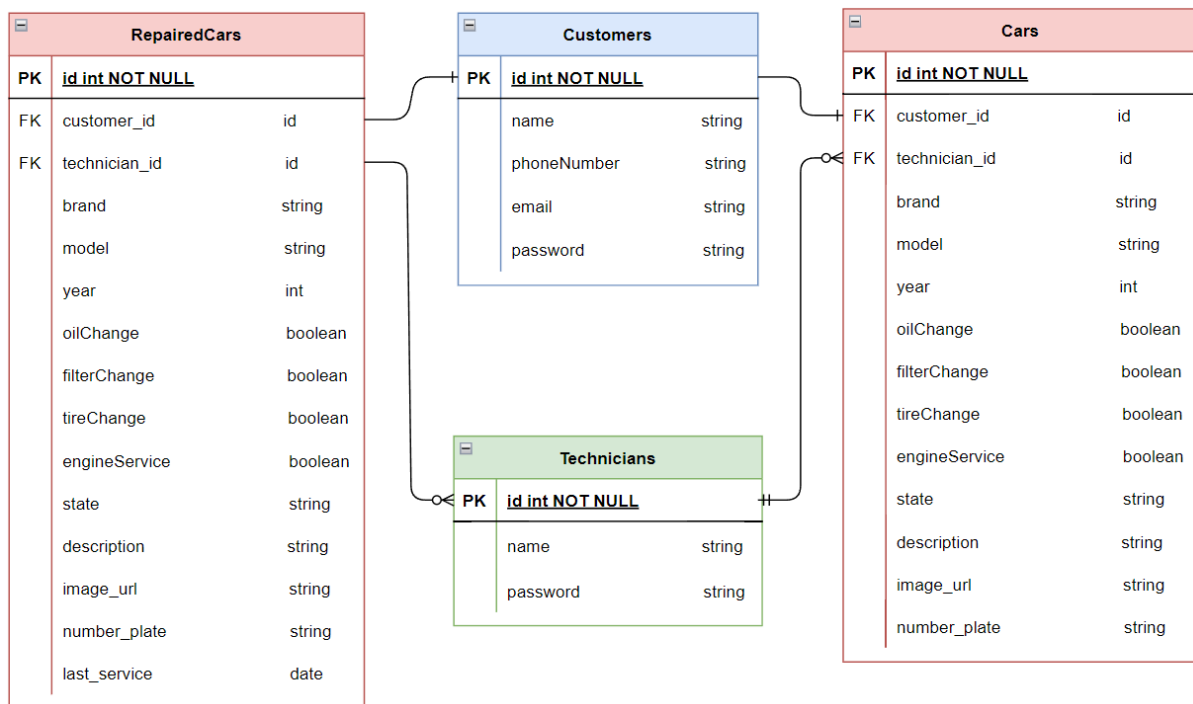
Registrácia



# Návrh databázy

Súčasťou projektu mobilnej aplikácie je návrh fyzického modelu databázy, ktorú použijeme. Na vytvorenie návrhu sme použili nástroj [draw.io](https://draw.io), kde sme jednotlivé schémy vedeli navrhnuť a tiež aj vzťahy medzi nimi.

Po zapracovaní na pripomienke ohľadom histórie opravených áut sme upravili náš databázový model. Použijeme ďalšiu schému, ktorá slúži na uloženie už opravených áut. Bude tam aj pridaný údaj kedy prebehol posledný servis alebo oprava auta, čo môže pomôcť technikovi pri riešení problémov s autom. Taktiež prebehli aj drobné úpravy v jednotlivých schémach, kde u zákazníka ani technika nemali uložené heslo pre prístup od aplikácie. Ďalej sme pridalí evidenčné číslo pre každé auto.



# Návrh API endpointov

Použili sme nástroj [Swagger](#), pre správu API dokumentácií. Poskytuje množstvo funkcionalít, ktoré nám umožnia v prehľadnej forme poskytovať údaje a detaily o API volaniach a endpointoch, ktoré budú súčasťou nášho projektu.

Prihlásenie používateľa		^
POST	/login Skontroluje zadané používateľské údaje	▼
Zoznam používateľov		^
GET	/customers Vráti zoznam všetkých používateľov.	▼
Vkladanie používateľa do DB		^
POST	/customers Vytvorí nový záznam používateľa v databáze	▼
Zobrazenie informácií o používateli		^
GET	/customers/{id} Vráti informácie o používateli	▼
Zoznam vozidiel		^
GET	/cars Vráti zoznam všetkých vozidiel	▼
Vloženie auta do DB		^
POST	/cars Vytvorí nový záznam používateľa v databáze	▼
Zoznam opravených vozidiel		^
GET	/repairedCars Vráti zoznam všetkých vozidiel	▼
Vloženie auta do histórie DB		^
POST	/repairedCars Vytvorí nový záznam používateľa v databáze	▼
Zobrazenie informácií o vozidle		^
GET	/cars/{id} Vráti údaje o vozidle	▼
Odstránenie vozidla z DB		^
DELETE	/cars/{id} Odstráni vozidlo z DB	▼
Úprava stavu vozidla		^
PUT	/cars/{id} Upraví údaje o Vozidle	▼
Zobrazenie technických vozidiel		^
GET	/technicianCars/{id} Vráti všetky vozidlá na ktorých pracuje konkrétny technik	▼
Zobrazenie informácií o zákazníkovi vozidle		^
GET	/customerCar/{id} Vráti údaje o vozidle podľa prislúchajúceho zákazníka	▼
Zoznam technikov		^
GET	/technicians Vráti zoznam všetkých technikov	▼
Zobrazenie informácií o technikovi		^
GET	/technicians/{id} Vráti údaje o technikovi	▼

Každý z API endpointov obsahuje popis funkcionality, detaily parametrov a odpovedí. Na spodnom obrázku je príklad detailov po rozkliknutí endpointu.

## Zobrazenie informácií o vozidle

**GET** `/cars/{id}` Vráti údaje o vozidle

Toto volanie využívame na získanie informácií o aute, na ktorom sa práve vykonáva servis.

**Parameters** Try it out

Name	Description
<b>id</b> * <i>required</i>	unikátny identifikátor vozidla
number (path)	<input type="text" value="id"/>

**Responses**

Code	Description	Links
200	Informácie o vozidle  <div>Media type <div>application/json</div><div>Controls: Accept: header.</div><div>Example Value   Schema</div><pre>{   "id": 1,   "customer_id": 1,   "technician_id": 1,   "brand": "BMW",   "model": "X3",   "year": 2011,   "oilChange": true,   "filterChange": true,   "tyreChange": true,   "engineService": true,   "state": "not_finished",   "description": "velmi pokazene",   "image_url": "obrazok.jpg",   "number_plate": "TT816DX" }</pre></div>	No links
404	Vozidlo neexistuje  <div>Media type <div>application/json</div><div>Example Value   Schema</div><pre>{   "message": "Vozidlo neexistuje." }</pre></div>	No links



# Akceptačné testy

Vytvorili sme dokopy 10 akceptačných testov pre frontendovú časť aplikácie a tiež pre backend aplikácie. Rozdelili sme testy na dva typy, kedy správanie používateľa je kladné a kedy je záporné. Podľa toho dostávame rôzne výstupy pri interakcii s našou aplikáciou.

## Frontend aplikácie

### Kladné testy

Test 1: Prihlásenie sa	
Vstupné podmienky:	Používateľ, ktorý má vytvorený účet a nachádza sa v databáze.
Výstupné podmienky:	Používateľ sa úspešne prihlási do svojho účtu.
Postup:	<ol style="list-style-type: none"><li>1. Používateľ zadá svoje prihlasovacie údaje.</li><li>2. Používateľ stlačí tlačidlo PRIHLÁSIŤ SA.</li><li>3. Používateľovi sa zobrazí obrazovka s jeho údajmi a možnosťou objednania sa do servisu.</li></ol>
Výsledok: PASS / FAIL	

<b>Test 2: Vyplnenie požadovaných servisných úkonov.</b>	
<b>Vstupné podmienky:</b>	Používateľ, ktorý zadal základné informácie o jeho aute a stlačil tlačidlo POKRAČOVAŤ.
<b>Výstupné podmienky:</b>	Zákazník vyplnil formulár s požiadavkami o servise a prechádza na posledný krok objednávky do servisu.
<b>Postup:</b>	<ol style="list-style-type: none"> <li>1. Používateľ zaškrtnie checkboxy všetkých požadovaných úkonov.</li> <li>2. Ak používateľ nemá žiadne konkrétnejšie požiadavky k servisu, preskočiť na krok 4.</li> <li>3. Ak používateľ má ďalšie servisné požiadavky, napíše ich do poľa DOPLŇUJÚCE INFORMÁCIE.</li> <li>4. Po vyplnení všetkých požadovaných servisných úkonov klikne používateľ na tlačidlo NAHRAŤ FOTOGRAFIE VOZIDLA.</li> <li>5. Používateľovi sa zobrazí obrazovka, v ktorej musí nahrať fotografiu/fotografie vozidla.</li> </ol>
<b>Výsledok: PASS / FAIL</b>	

<b>Test 3: Vloženie fotografií servisovaného vozidla</b>	
<b>Vstupné podmienky:</b>	Používateľ, ktorý vyplnil všetky potrebné informácie o servisnom úkone.
<b>Výstupné podmienky:</b>	Používateľove auto bude zaradené do databázy. Servisovanie auta následne bude pridelené požadovanému technikovi(ak si takého používateľ zvolil) a zákazník si bude môcť pozrieť stav opravy vozidla.
<b>Postup:</b>	<ol style="list-style-type: none"> <li>1. Používateľ klikne na tlačidlo VYBRAŤ OBRÁZOK.</li> <li>2. Používateľ vloží fotografiu/fotografie jeho vozidla.</li> <li>3. Používateľovi sa zobrazia vložené fotografie na obrazovke.</li> <li>4. Používateľ klikne na OBJEDNAŤ VOZIDLO DO SERVISU.</li> <li>5. Používateľovi je zobrazená začiatočná obrazovka s jeho údajmi, kde už môže skontrolovať stav jeho servisovaného vozidla.</li> </ol>
<b>Výsledok: PASS / FAIL</b>	

## Záporné testy

Test 4: Vytvorenie nového účtu	
Vstupné podmienky:	Používateľ, ktorý už je zaregistrovaný sa nachádza v login obrazovke aplikácie.
Výstupné podmienky:	Používateľ sa bude musieť vrátiť na prihlasovaciu obrazovku, keďže účet už vytvorený mal.
Postup:	<ol style="list-style-type: none"><li>1. Používateľ stlačí tlačidlo REGISTRÁCIA.</li><li>2. Zobrazí sa registračný formulár.</li><li>3. Používateľ vyplní všetky potrebné polia, no zadá tel.č. alebo e-mail, ktorý už existuje v databáze.</li><li>4. Používateľ klikne na tlačidlo REGISTROVAŤ.</li><li>5. Používateľ bude upozornený, že tento účet už existuje.</li><li>6. Používateľ stlačí tlačidlo Späť.</li></ol>
Výsledok: PASS / FAIL	

<b>Test 5: Vyplnenie základných informácií o vozidle</b>	
<b>Vstupné podmienky:</b>	Používateľ, ktorý je registrovaný a chce objednať svoje auto do servisu.
<b>Výstupné podmienky:</b>	Zákazník úspešne informácie o jeho vozidle, a prejde na druhý krok objednávky do servisu. V prípade nevyplnenia jedného z povinných polí je používateľ upozornený.
<b>Postup:</b>	<ol style="list-style-type: none"> <li>1. Používateľ stlačí tlačidlo OBJEDNAŤ VOZIDLO DO SERVISU.</li> <li>2. Používateľovi sa zobrazí formulár, v ktorom musí vyplniť informácie o jeho aute.</li> <li>3. Používateľ vyplní formulár, s tým že zabudne zadať niektoré z povinných polí.</li> <li>4. Ak používateľ nemá požiadavku vybrať si konkrétneho technika, preskočiť na krok 7.</li> <li>5. Ak si používateľ žiada konkrétneho technika, musí zaškrtnúť checkbox.</li> <li>6. Zákazník si zo zobrazeného dropdown menu zvolí preferovaného technika, ktorému bude servis pridelený.</li> <li>7. Zákazník klikne na tlačidlo POKRAČOVAŤ.</li> <li>8. Zákazník bude upozornený, že zabudol vyplniť všetky povinné údaje.</li> <li>9. Zákazník po upozornení vyplní potrebné údaje, preskočiť na krok 7.</li> </ol>
<b>Výsledok: PASS / FAIL</b>	

## Backend

### Kladné testy

Test 6: Správna registrácia	
Vstupné podmienky:	Pri registrácii užívateľ užívateľ polia.
Výstupné podmienky:	Užívateľské údaje sú zapísané v databáze.
Postup:	<ol style="list-style-type: none"><li>1. Je odoslaná POST požiadavka na endpoint <code>/customers</code>, ktorá obsahuje request body s vyplnenými údajmi</li><li>2. V backende sa spracujú údaje a skontroluje sa, či povinné polia majú správny tvar a boli vyplnené</li><li>3. Ak boli správne vyplnené, tak vraciame response s pôvodným body a so status kódom 201, inak príde odpoveď vo formáte JSON s informáciou, aká nastala chyba so status kódom 400</li></ol>
Výsledok: PASS / FAIL	

Test 7: Dokončený servis auta	
<b>Vstupné podmienky:</b>	Používateľ má auto v servise, je opravené a pripravené na vyzdvihnutie.
<b>Výstupné podmienky:</b>	Opravené auto sa vymaže z databázy áut, ktoré sú servisované.
<b>Postup:</b>	<ol style="list-style-type: none"> <li>1. Je odoslaná DELETE požiadavka na endpoint <code>/cars/:id</code></li> <li>2. Obsahuje request body s údajmi o konkrétnom aute, ktoré databáza spracuje</li> <li>3. Následne príde response s kódom 200 a informačným body, že auta sa vymazalo zo databázy v prípade úspešného odstránenia. Inak príde informácia v response body o chybe, že dané aute nie je evidované</li> </ol>
<b>Výsledok:</b> PASS / FAIL	

Test 8: Získanie údajov o aute zákazníka	
<b>Vstupné podmienky:</b>	Používateľ má aute momentálne v servise.
<b>Výstupné podmienky:</b>	Používateľ vidí stav jeho auta, aké úkony sa vykonali na jeho aute.
<b>Postup:</b>	<ol style="list-style-type: none"> <li>1. Je odoslaná GET požiadavka na endpoint <code>/cars/:id</code>.</li> <li>2. V backende sa request spracuje a hľadá sa auto s daným ID.</li> <li>3. Ak sa auto s konkrétnym a jeho údaje sa vrátia v reponse body spolu s kódom 200. V prípade ak také auto nie je v databáze, vracia sa chybový kód 500 a response body s ohlásenou chybou, že auto nie je v databáze</li> </ol>
<b>Výsledok:</b> PASS / FAIL	

## Záporné testy

Test 9: Nepridelené autá technikovi	
Vstupné podmienky:	Technik chce vidieť stav pridelených áut, ale nemá žiadne auto pridelené.
Výstupné podmienky:	Technik neuvidí pridelené autá, zobrazí sa informačná chybová hláška.
Postup:	<ol style="list-style-type: none"><li>1. Je odoslaná GET požiadavka na endpoint <code>/TechnicianCars/:id</code>, a request body s obrázkom</li><li>2. Na backende sa zistí, pri vyhľadávaní áut v databáze je odpoveďou prázdne pole</li><li>3. Posiela sa response body s chybovým kódom 404 a správou, že daný technik nemá pridelené ani jedno auto</li></ol>
Výsledok: PASS / FAIL	

Test 10: Nekompletné vyplnenie údajov auta	
Vstupné podmienky:	Používateľ zadáva objednávku, ale zabudol vyplniť správne rok výroby auta, čo je povinný údaj.
Výstupné podmienky:	Používateľovi sa zobrazí chybová správa o tom, že je potrebné tento údaj vyplniť.
Postup:	<ol style="list-style-type: none"><li>1. Je odoslaná POST požiadavka na endpoint <code>/cars</code>.</li><li>2. Na backende sa zistí, že nie sú vyplnené všetky povinné polia pri pridávaní auta a jeho údajov do databázy.</li><li>3. Posiela sa response body s chybovým kódom 400 a správou, údaje o aute neboli úspešne registrované.</li></ol>
Výsledok: PASS / FAIL	



# Backend aplikácie

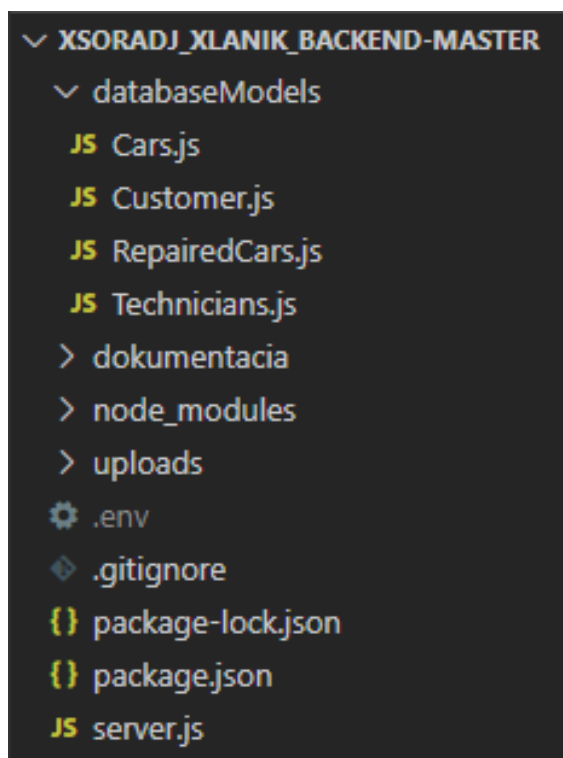
Súčasťou projektu je samotný backend aplikácie. Programovali sme v programovacom prostredí Visual Studio Code. Zdrojový kód aj s jeho súčasťami je dostupný v [Github repozitári](#).

## Backend technológie

Pre naprogramovanie backendu a funkčných endpointov sme použili [Node.js](#) spolu s frameworkom [Express.js](#). Spoločná kombinácia poskytuje množstvo funkcionalít, na programovanie webových a mobilných aplikácií. Použili sme databázu [MongoDB](#) a tiež aj “[mongoose](#)” pre prehľadné objektové modelovanie.

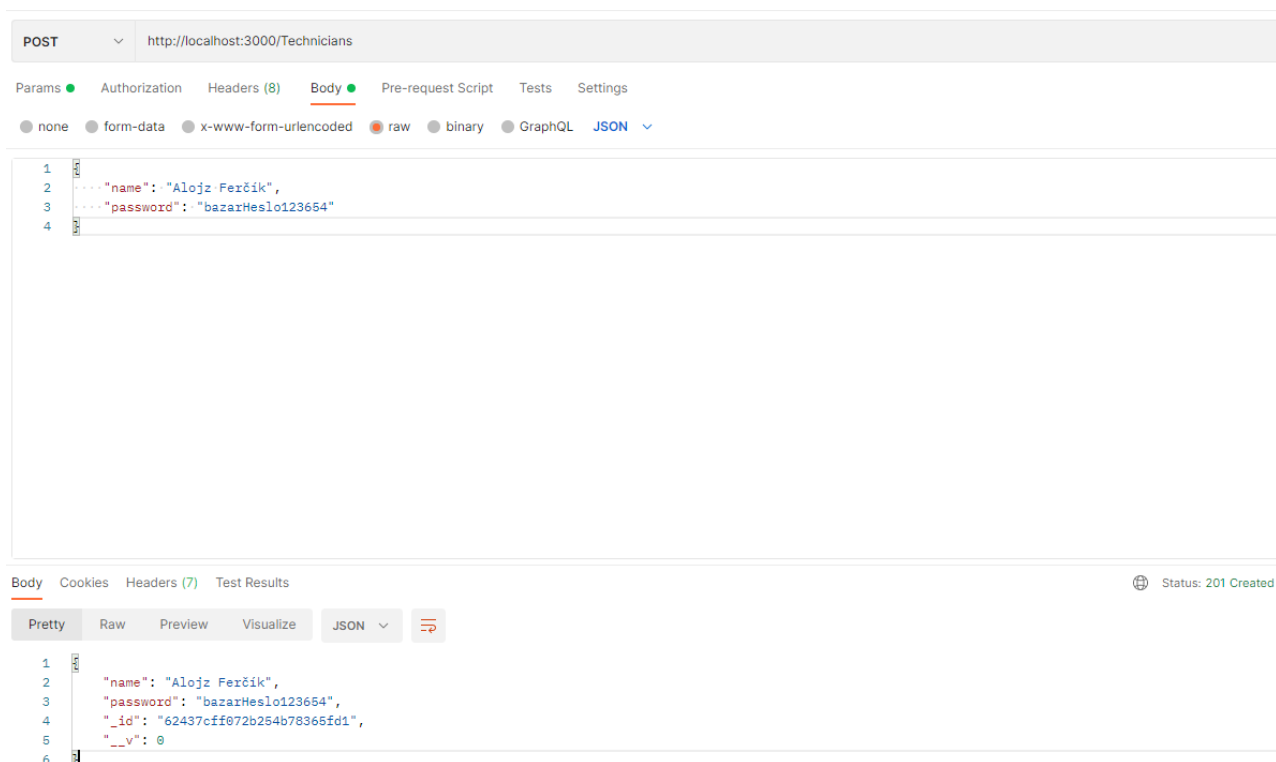
## Štruktúra backendu a jeho funkcionalita

Štruktúra projektu obsahuje dôležité súčasti nato, aby sme mohli spúšťať projekt (*node\_modules*, *package.json*, *package-lock.json*). Hlavná logika programu je naprogramovaná v súbore *server.js*, kde je základná funkcionalita, definícia API endpointov, pripojenie na databázu a samotné operácie s JSON objektami a databázou. Priečink *databaseModels* obsahuje modely, pre každú tabuľku nášho fyzického databázového modelu. V ňom sú definované štruktúry/modely každej tabuľky databázy.



## Testovanie backendu

Pre overenie funkčnosti API endpointov sme použili nástroj alebo aplikáciu [Postman](#), ktorým sme sa správali ako HTTP klient a posielali sme HTTP request na náš backend. Následne sme správanie backendu mohli overovať odpoveďami priamo v Postman-ovi alebo tiež aj v samotnej databáze v prípade pridania, upravenia alebo odstránenia položky v databáze.



Tu vidíme príklad testovania funkčnosti nášho backendu, kde sme poslali POST so vstupnými údajmi vo formáte JSON. Dostali sme aj odpoveď so status kódom a v databáze sa takýto technik zapísal do príslušnej tabuľky technikov. Testovali sme všetky metódy a to GET, POST, PUT (PATCH) a DELETE. K dispozícii v repozitári je aj kolekcia volaní v nástroji Postman. Všetky API endpointy sú funkčné tak, ako sme pôvodne chceli a sme pripravení na prácu na frontendovej časti projektu.

## Change log oproti Milestone 1

V milestone 1 odovzdaní projektu a teda oproti samotnému návrhu nastalo pár zmien, ktoré boli urobené v záujme zvýšenia kvality a konzistentnosti projektu:

- správne upravené názvy API endpointov, niektorých premenných v API dokumentácii nástroja Swagger.
- upravené detaily pri akceptačných testoch (status kódy, typ response body)
- pridaný POST endpoint */login*, na prihlásenie zákazníka alebo technika do systému. Súčasťou je aj validácia vstupných údajov.
- pridaný GET endpoint */CustomerCar/:id*, pre získanie auta príslušného zákazníka
- pridaný GET endpoint */TechnicianCars/:id*, pre získanie všetkých áut, ktoré má daný technik pridelené
- pridaná funkcionálna uloženia obrázku do databázy, kde sa zapíše celý obsah obrázka vo formáte base64
- prerobený akceptačný test č.9, kde sme použili jeden z nových endpointov.