



BKSZC Pogány Frigyes Technikum

# CarRepair - E-szervizkönyv

szoftverfejlesztő és -tesztelő  
vizsgaremek

2022. május

Készítette:  
Hollander Norbert  
Nagy Zsolt  
Németh Máté

Konzulens:  
Tóth József

# A feladat leírása

Autószerbizünk a CarRepair olyan felületet szeretne létrehozni, ahol a látogatók regisztrálhatnak és autóikat online módon tudják bejelenteni szervizelésre, majd a már kész munkákat autónként visszanézhetik, mint egy elektronikus szervizkönyvet.

A feladat olyan webhely és mobilalkalmazás létrehozása, amelyen a látogatók regisztrálhatnak az autószerbiz adatbázisába, ahol saját adataik mellett regisztrálhatják autóikat flottájukba, majd ezt a flottát kezelhetik, törléssel vagy szervizbe jelentkezéssel. A flottáról egyenként lekérdezhetik az adott autók szerviz előzményeit, mint egy elektronikus szervizkönyvben.

Az adatokat egy online adatbázisban kell tárolni, amelyet egy API-n keresztül lehet elérni. Az elkészített weblapok, illetve mobil alkalmazás erről az API-ról töltik le és jelenítik meg az adatokat.

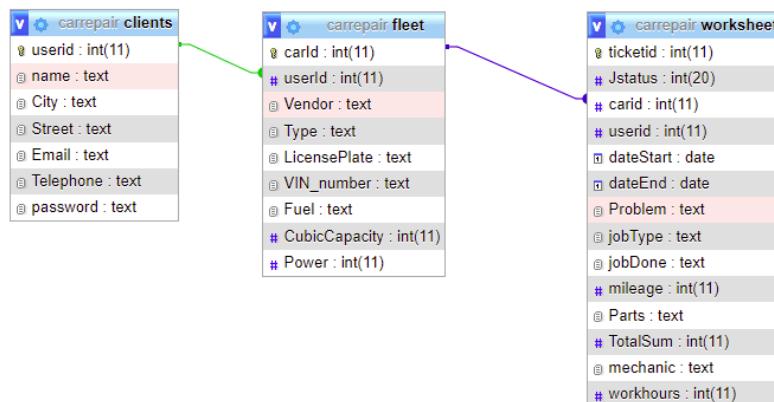
A felhasználói felületen kívül van még egy adminisztrátori felület, amely nem külön bejelentkezási hely mellett elérhető, hanem a felhasználónév és jelszó pontos megadásával egy más felületre irányít minket, ahol a bejelentkezett autók munkalapjait láthatjuk, azokat törölhetjük, vagy kitölthetjük. Továbbá van egy olyan fül ahol az összes munkalap megjelenik.

## GitHub repository

Az alkalmazások kódjai ebben a repository-ban érhetők el:  
[https://github.com/unisiar33/Car-Repair\\_VizsgaRemek.git](https://github.com/unisiar33/Car-Repair_VizsgaRemek.git)

## Az adatbázis elkészítése

Az adatbázist a MySQL adatbázis-kezelővel és a PHPMyAdmin programmal készítettük el. Az adatokat három táblára bontottuk:



## A clients(felhasználók) tábla adatai

Minden felhasználónak /client -nek kell egy azonosító (userid), amely az elsődleges kulcs lesz. Ezen kívül szükség van :

- Névre (name) ez szolgál majd bejelentkezéshez
- Város (City) felhasználó saját oldala betöltésekor visszaköszönő adat
- Utca(Street) felhasználó saját oldala betöltésekor visszaköszönő adat
- Email (Email) felhasználó saját oldala betöltésekor visszaköszönő adat, további fejlesztési lehetőségeknél fontos adat például : Email küldés szerviz munkalappal, visszaigazoló email bejelentkezéskor
- Telefonszám(Telephone) felhasználó saját oldala betöltésekor visszaköszönő adat
- Jelszó(password) hash kód amely a megadott jelszó és a bicrypttel készült

Ennek megfelelően a képzesek tábla szerkezete:

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
1	userid	int(11)			Nem	Nincs		AUTO_INCREMENT	Módosítás  Eldobás ▾ Több
2	name	text	utf8mb4_hungarian_ci		Nem	Nincs			Módosítás  Eldobás ▾ Több
3	City	text	utf8mb4_hungarian_ci		Nem	Nincs			Módosítás  Eldobás ▾ Több
4	Street	text	utf8mb4_hungarian_ci		Nem	Nincs			Módosítás  Eldobás ▾ Több
5	Email	text	utf8mb4_hungarian_ci		Nem	Nincs			Módosítás  Eldobás ▾ Több
6	Telephone	text	utf8mb4_hungarian_ci		Nem	Nincs			Módosítás  Eldobás ▾ Több
7	password	text	utf8mb4_hungarian_ci		Nem	Nincs			Módosítás  Eldobás ▾ Több

Az alkalmazás teszteléséhez az alábbi tesztadatokat vittük be.:

userid	name	City	Street	Email	Telephone	password
2	Norbert Hollander	Érd	Porcsinrőzsa u.	norbert.hollander@gmail.com	telephone	\$2b\$10\$UD45bk.kJ5wz2VYcScluupgCUZUDCvX.V3wZRseTF1...
12	Admin	Budapest	Porcsinrőzsa utca	admin@admin.hu	06202238988	\$2b\$10\$KriXiGjGyeneUi6FhwTOfSndVzD49E4wFGuSGvwowX...
27	Kovács István	Budapest	Kalapka utca 74.	kovacs.i@gmail.com	+36304598466	\$2b\$10\$27nDDl3L0WsttzuuCltud3yRPqly.A6SuDSyUWr...
28	Mendi Bertalan	Budapest	Vádrózsa utca 12.	mendib@gmail.com	+36706189523	\$2b\$10\$R0lzzXhO/Ci69evjQ72Vj.u7aSz88sml.Hv0MRxx7X...
29	Kékesi Tamás	Budapest	Csokonai utca 1/b	kekesitamas22@gmail.com	+36203438658	\$2b\$10\$I.omnia/XsR.8TbEaB2XpuS13fHeLygOU2kzO.XLiU...
30	Kelemen Géza	Veszés	Booskai utca 49.	gezakelemen@freemail.com	+36301318456	\$2b\$10\$GkeVuuzZhqq0.VVgshVsgTOBkpTZmkRivWObRi0wlP9...
31	Gádos Ferenc	Budapest	Gólya utca 31.	fgados@gmail.com	+36305678485	\$2b\$10\$AhQuRHo82bi9C09bfOHa.7zTMFL1Y1kuGiZLnIfA...
32	Gádosné Kalocsai Ilona	Budapest	Gólya utca 62.	ilona88@gmail.com	+36308589904	\$2b\$10\$dfx1qm15ITeaM29i7rsodO1NwTyVd3FGD8GFixAoIKX...
33	Németh László	Gyál	Erdősor utca 43.	nemeth@freemail.com	+36702345659	\$2b\$10\$aV86z0vo5/4zR4U/5FdJGemVMPLaHfYdsdaNQnq3vL...
34	Klózs Renáto	Budapest	Ady Endre út 11.	kloszi@gmail.com	+36703578468	\$2b\$10\$wPShFugAbfjS.E8Lwmmy3EORkGMKLg76SpP1aY1SRU7...
35	Kaposvári Imre	Veszés	Juhász Gyula utca 73.	imike32@gmail.com	+36705490507	\$2b\$10\$4pb47G.M2jyjH24lg.zHOZm1.ya6C5ObLpk85927M...
36	Juhász István	Üllő	Bercsényi utca 9.	juhistvan@freemail.com	+36204648584	\$2b\$10\$9ao4pT29daKCoPMTmifcOVsC6wel6fqUtmBR1DVrB...
37	Horváth Bence	Alsónémedi	Kápolna utca 63.	bence8742@gmail.com	+36204139854	\$2b\$10\$3yqpmAnsHyG80un.Iu18uDm4U57rTdx3qjcpsQqO...
38	Szentpéteri Gizella	Pécel	Felsőszor utca 55.	gizella77@gmail.com	+36201389451	\$2b\$10\$sk1Av2prFA8iN8DW7IQCydejTxdk0Vxjt0MT07tZN3L...
39	Bede László	Budapest	Lajos utca 8.	laszlo78@gmail.com	+36703239438	\$2b\$10\$XNqANRu7ywFOkkyBkrTcdm95R4vscksleeBaW8sb...
40	Pere László	Budapest	Bojtár utca 3.	pere@freemail.com	+36709946512	\$2b\$10\$SLZEx.Qgp5M58JHM0z58tVOEjqmq02u.DY3Ti5MNWpL...
41	Góbolyós Zsolt	Budapest	Csillaghégyi út 57.	zsolt@gmail.com	+36301229497	\$2b\$10\$Lzhxnnon5gGEeKTfLeMKIOtkUY9pUH7YHFGymFBfALB...
42	Támadai Viktor	Budapest	Gödöllői utca 77.	viktor33@gmail.com	+36304648482	\$2b\$10\$SMvoHiuj0vuRXxgvjebtYFeLqz8hc2Brc1CxoxynOcf...
43	Szalanka Gergely	Budapest	Szép utca 32.	gergely@freemail.com	+36204563138	\$2b\$10\$irJog1RL120VgTKuRnwxeCP3nh2koz7G7lohk2unT...
44	Pál András	Budapest	Csóka utca 69.	andris55pal@gmail.com	+36302899488	\$2b\$10\$rlg1myqy90hIO1x4bZsEeQeu.MPh/2ZBFbWh7baHa...
45	Faragó Ábel	Csömör	Anna utca 4.	abelfarago@gmail.com	+36705466563	\$2b\$10\$ik2.mTxnjPp9/kBnrnLBOEIGuqRef4X8W8vMMZx.Pt...
46	Fülöp Dániel	Budapest	Lipót utca 73.	danielfulop@freemail.com	+36703238483	\$2b\$10\$sk74dsKz4X08aE2Qm3Oui0W3jE5Di8GMUwlJ.UE...

## A fleet /fotta tábla adata

Ez a tábla a felhasználók autóinak életszerű jellemzőit tartalmazza:

- carId: autó azonosítója (elsődleges kulcs),
- userId: felhasználó azonosítója (idegen kulcs),
- Vendor: a gyártó neve,
- Type: a típus neve
- LicensePlate: rendszámtábla
- VIN\_number: alvázszám
- Fuel : üzemanyag típusa
- CubicCapacity:lökettérfogat
- Power : teljesítmény

Az elkészített tábla szerkezete:

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
1	carId	int(11)			Nem	Nincs		AUTO_INCREMENT	Módosítás  Eldobás  Több
2	userId	int(11)			Nem	Nincs			Módosítás  Eldobás  Több
3	Vendor	text	utf8mb4_hungarian_ci		Nem	Nincs			Módosítás  Eldobás  Több
4	Type	text	utf8mb4_hungarian_ci		Nem	Nincs			Módosítás  Eldobás  Több
5	LicensePlate	text	utf8mb4_hungarian_ci		Nem	Nincs			Módosítás  Eldobás  Több
6	VIN_number	text	utf8mb4_hungarian_ci		Nem	Nincs			Módosítás  Eldobás  Több
7	Fuel	text	utf8mb4_hungarian_ci		Nem	Nincs			Módosítás  Eldobás  Több
8	CubicCapacity	int(11)			Nem	Nincs			Módosítás  Eldobás  Több
9	Power	int(11)			Nem	Nincs			Módosítás  Eldobás  Több

A userId mező idegen kulcs beállítása :

Megszorítás tulajdonságai		Oszlop	Idegen kulcs megszorítás (INNODB)		
			Adatbázis	Tábla	Oszlop
fleet_ibfk_1		ON DELETE RESTRICT	userId	carrepair	usersend
		ON UPDATE RESTRICT			
		+ Oszlop hozzáadása			

## Tesztadatok:

carId	userId	Vendor	Type	LicensePlate	VIN_number	Fuel	CubicCapacity	Power
9	2	Peugeot	307 Break	IMN 624	FR307161234NFUTU5121424535	Gasoline	1596	110
26	2	Alfa Romeo	156	JTD865	IT1323245JTD198V2321425356	Diesel	1896	85
30	27	Skoda	Octavia	MTC-905	4NUDT13S962700984	Gasoline	1984	230
31	27	Toyota	Yaris	SKE-704	KM8JT3AC2DU583865	Gasoline	998	65
32	28	Toyota	Prius	SDW-264	1HGEM21292L047875	Gasoline	1798	99
33	29	Ford	Mustang	PDX-400	1G8ZH1277XZ105148	Gasoline	4951	421
34	29	Porsche	911	PRS-911	2CNBJ1365W8902635	Gasoline	3614	345
35	30	Alfa Romeo	GT	SAP-932	1C3BF88P0GX570598	Gasoline	1970	166
36	31	Mercedes-Benz	CLA	RRH-050	1GTEK19RXVE536195	Gasoline	1332	136
37	31	Mercedes-Benz	GLC	RSE-384	JH4DA3453GS008314	Gasoline	1991	245
38	32	Mercedes-Benz	A 180	RWA-888	JL5TA3453GS256378	Gasoline	1332	136
39	33	Chevrolet	Aveo	LCW-341	1FMCU04112KA71263	Gasoline	1150	72
40	34	BMW	318i	MZP-215	1GCHK29U87E198893	Gasoline	1995	143
41	35	Fiat	Tipo	NZW-843	KMH DU4AD5AU138970	Gasoline	1598	120
42	35	Fiat	500	PHE-572	JN8MD06S2BW031939	Gasoline	1272	89
43	36	Tesla	Model 3	TBZ-177	KLATA52871B611178	Electric	0	513
44	37	BMW	i3	SRA-102	1J4GW48S94C420221	Electric	0	170
45	37	Skoda	Octavia	TBJ-815	WD5WD641525381291	Diesel	1968	150
46	38	Citroen	C4	LOA-887	2HGES15252H603204	Diesel	1560	90
47	39	Peugeot	208	KRG-792	2C3KA73WX8H237747	Gasoline	1398	88
48	39	Nissan	Navara	PPX-995	3B7HF13Y81G193584	Diesel	2298	190
49	40	Daewoo	Matiz	HDA-359	WDBHA33G2XF844170	Gasoline	796	52
50	41	Volkswagen	Golf	SLK-780	ZFFEW58A680144998	Diesel	1968	150
51	42	Volvo	S60	INF-229	1FMZK05135GAGG488	LPG	2435	140
52	43	Volvo	XC40	SUZ-835	JH4DA9480PS008002	Electric	0	408

## A worksheet /munkalap tábla

Ebbe a táblába a szervizbe bejelenkezett illetve az elvégzett munkalapok (ticketek) és azok adatait tartalmazza.

- ticketid: a munkalap azonosítója,elsődleges kulcs
- carid: az azonosítója,másodlagos kulcs
- userid: felhasználó azonosítója, másodlagos kulcs
- Jstatus: munka állapota ( kész/ nincs kész)
- dateStart: kezdés időpontja
- dateEnd: befejezés időpontja
- Problem: Hibaleírás
- jobType: munkatípusa
- JobDone: megoldás a hibára
- Mileage: km állás
- Parts: felhasznált alkatrészek további

- Totalsum: összköltség
- Mechanic: felelős szerelő
- Workhours: munkaóra

A tábla szerkezete:

Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
ticketid	int(11)			Nem	Nincs		AUTO_INCREMENT	Módosítás  Eldobás  Több
Jstatus	int(20)			Nem	0			Módosítás  Eldobás  Több
carid	int(11)			Nem	Nincs			Módosítás  Eldobás  Több
userid	int(11)			Nem	Nincs			Módosítás  Eldobás  Több
dateStart	date			Nem	Nincs			Módosítás  Eldobás  Több
dateEnd	date			Nem	Nincs			Módosítás  Eldobás  Több
Problem	text	utf8mb4_hungarian_ci		Nem	Nincs			Módosítás  Eldobás  Több
jobType	text	utf8mb4_hungarian_ci		Nem	Nincs			Módosítás  Eldobás  Több
jobDone	text	utf8mb4_hungarian_ci		Nem	Nincs			Módosítás  Eldobás  Több
mileage	int(11)			Nem	Nincs			Módosítás  Eldobás  Több
Parts	text	utf8mb4_hungarian_ci		Nem	Nincs			Módosítás  Eldobás  Több
TotalSum	int(11)			Nem	Nincs			Módosítás  Eldobás  Több
mechanic	text	utf8mb4_hungarian_ci		Nem	Nincs			Módosítás  Eldobás  Több
workhours	int(11)			Nem	Nincs			Módosítás  Eldobás  Több

Idegen kulcsok beállításai:

Megszorítás tulajdonságai		Oszlop	Idegen kulcs megszorítás (INNODB)		
		Adatbázis	Tábla	Oszlop	
worksheet_ibfk_2	ON DELETE RESTRICT  ON UPDATE RESTRICT	carid	carrepair	fleet	carid
	+ Oszlop hozzáadása				
worksheet_ibfk_3	ON DELETE RESTRICT  ON UPDATE RESTRICT	userid	carrepair	clients	userid
	+ Oszlop hozzáadása				

A tesztadatok:

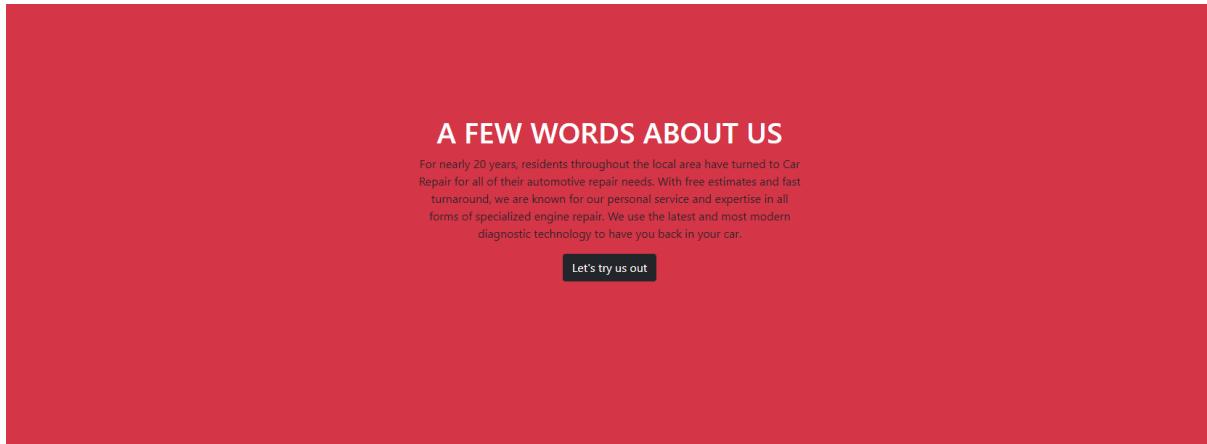
ticketid	Jstatus	carid	userid	dateStart	dateEnd	Problem	jobType	jobDone	mileage	Parts	TotalSum	mechanic	workhours
16	1	26	2	2023-03-28	2022-03-28	Balról bala húz	General mechanics	Futómű beállítás	171622	-	28623	Bill Right	2
22	1	30	27	2022-04-15	2022-04-15	Kopott gumiabroncsok	Tyre change	Resolved job	162351	4x LingLong Greenmax All Season	110000	Eng. Thomas Doll	1
23	1	30	27	2022-05-15	2022-05-15	Időszakos olajcsere	General mechanics	Planned job	0	5,5L 5W-30 motorolaj, olajszűrő, pollenszűrő, leve...	50000	Eng. Thomas Doll	2
24	1	30	27	2022-05-14	2022-04-14	Bal tömpített fényszóró nem ég	General electronics	Planned job	0	Osim H4 izzó (pár)	8000	Kurt Alfredson	1
25	1	31	27	2022-03-18	2022-03-17	Karosszéria hibák javítása	General mechanics	Resolved job	144274	Első lökhárító (csere), jobb első sárvédő (javítás...	130000	Joe Cocker	8
26	1	32	28	2022-03-11	2022-03-11	Egyenletesen útjárás	General mechanics	Első stabilizátor rúdak és szilentek cseréje	387181	Első stabilizátor rúdak és szilentek	85000	Eng. Thomas Doll	4
27	1	32	28	2022-03-17	2022-03-17	Általános átvizsgálás	General mechanics	Resolved	387243	-	15000	Bill Right	1
28	1	32	28	2022-03-17	2022-03-17	MOT	MOT	Resolved	387279	-	22000	Bill Right	1
29	1	33	29	2021-12-10	2021-12-10	Kopott gumiabroncsok	Tyre change	Resolved	34980	4x Michelin Pilot Sport 4S	550000	Eng. Thomas Doll	1
30	1	33	29	2021-12-17	2021-12-17	MOT	MOT	Resolved	35020	-	22000	Eng. Thomas Doll	1
31	1	33	29	2022-04-01	2022-04-01	Csikorgó, fémes hang félkészénél a kocsi elejéből	General mechanics	Első fékbesítek valamint fékfürkészák cseréje	35320	2x fékberet, 2x fékfürkész	370000	Bill Right	3
32	1	34	29	2021-12-10	2021-12-10	Kopott gumiabroncsok	Tyre change	Resolved	677344	4x Pirelli Pzero Winter	345000	Eng. Thomas Doll	1
33	1	34	29	2022-01-20	2022-01-20	Repedt szélvédő	General mechanics	Unresolved	677814	Első szélvédő	165000	Bill Right	9
34	1	35	30	2021-11-09	2021-11-09	Autóvillamosági problémák	General electronics	Meghibásodott akkumulátor cseréje	185344	Vára akkumulátor	55000	Eng. Thomas Doll	3
35	1	35	30	2021-11-26	2021-11-26	Check engine lámpa, ABS problémák	General mechanics	hibákód olvasás és ABS vezérlő cseréje	185491	ABS vezérlőegység és jeladó	488000	Bill Right	4
36	1	35	30	2021-11-29	2021-11-29	MOT	MOT	Unresolved	185528	Nem jelent meg a tulajdonos a pk-val	22000	Bill Right	1
37	1	35	30	2021-12-01	2021-12-01	MOT	MOT	Resolved	185568	-	22000	Bill Right	1
38	1	36	31	2022-01-05	2022-01-05	Időszakos olajcsere	General mechanics	Resolved	17000	5,5L 5W-30 motorolaj, olajszűrő, pollenszűrő, leve...	50000	Eng. Thomas Doll	2
39	1	36	31	2022-02-21	2022-02-21	Téli abroncsok nyárira cserélése	Tyre change	Resolved	17822	-	15000	Eng. Thomas Doll	1
40	1	36	31	2022-03-03	2022-03-03	Általános átvizsgálás	General mechanics	Resolved	17918	-	15000	Eng. Thomas Doll	1
41	1	36	31	2022-03-17	2022-03-17	MOT	MOT	Resolved	17948	-	22000	Bill Right	1
42	1	37	31	2021-10-08	2021-10-08	Kopott gumiabroncsok	Tyre change	Resolved	42600	4x Firestone Winterhawk 4	180000	Joe Cocker	1
43	1	37	31	2021-11-01	2021-11-01	Defektes bal első gumiabroncs	General mechanics	Resolved	42751	1x Firestone Winterhawk 4	45000	Joe Cocker	1

44	1	37	31	2021-11-01	2021-11-01	Az autó jobbra kanyarodáskor bugó hangot ad elő	General mechanics	Futómű átvizsgálás es einasznalcoott akárteszek c...	42751	Futómű beállítás, bal és jobb első kerékagy csapág...	75000	Eng. Thomas Doll	2
45	1	38	32	2022-04-11	2022-04-11	Vásárlás előtti átvizsgálás	General mechanics	Resolved	55000	-	15000	Kurt Alfredson	1
46	1	38	32	2022-04-11	2022-04-11	Olajcsere	General mechanics	Resolved	55002	4L 5W-30 motorolaj, olajszűrő, pollenszűrő, levegő...	45000	Kurt Alfredson	1
47	1	38	32	2022-04-11	2022-04-11	Vezérlés és folyadékok cseréje	General mechanics	Resolved	55002	Vezérlés szett, ablakmosó folyadék, fákola, hűtőv...	180000	Kurt Alfredson	4
48	1	38	32	2022-04-11	2022-04-11	Automatikálóban olajcsere	General mechanics	Resolved	55002	váltóolaj csere	25000	Kurt Alfredson	2
49	1	38	32	2022-04-11	2022-04-11	Egyéb kiegészítők vásárlása	General mechanics	Resolved	55002	gumiszöveng, csomagtérláda, izzókészlet	43750	Joe Cocker	1
50	1	39	33	2019-07-28	2019-07-28	MOT	MOT	Resolved	197421	-	25000	Kurt Alfredson	1
51	1	39	33	2019-11-15	2019-11-15	Nyári abroncsok téli cserélése	Tyre change	Resolved	200429	-	15000	Bill Right	1
52	1	39	33	2020-03-10	2020-03-10	Téli abroncsok nyári cserélése	General mechanics	Resolved	205108	-	15000	Joe Cocker	1
53	1	39	33	2020-07-14	2020-07-14	Időszakos olajcsere	General mechanics	Resolved	208244	4L 5W-30 motorolaj, olajszűrő, pollenszűrő, levegő...	40000	Bill Right	2
54	1	39	33	2021-09-28	2021-09-28	Fura, csúszó hang a motortéről	General mechanics	Vezérlés csere	216589	Vezérlés szett	120000	Kurt Alfredson	2
55	1	40	34	2021-08-06	2021-08-06	MOT	MOT	Nem felelt meg a vizsgán.	345988	-	22000	Bill Right	1
56	1	40	34	2021-08-09	2021-08-30	Nagy füst Jon a kipufogóból, gyakran fogyd hűtővíz	General mechanics	Motor felújítása	346001	Dugattuk és blokk felújítása, tömíthet valamit s...	568000	Kurt Alfredson	30
57	1	40	34	2021-08-31	2021-08-31	MOT	MOT	Resolved	346001	-	22000	Joe Cocker	1
58	1	41	35	2021-07-04	2021-07-04	Eszterkai hibák kijavítása	General mechanics	első és hátsó lökhárítók fényezése, gépházirő...	7719	Gépházirő	290000	Joe Cocker	20
59	1	41	35	2021-09-19	2021-09-19	A pötök lámpa nem világít	General electronics	Kickerrelve	78428	Pötök lámpa modul	27000	Eng. Thomas Doll	3
60	1	41	35	2021-12-02	2021-12-02	Előregedett nyári gumiabroncsok cseréje	General mechanics	Resolved	79711	4x Viking 4Season	130000	Eng. Thomas Doll	1
61	1	42	35	2022-01-11	2022-01-11	Vásárlás előtti átvizsgálás	General mechanics	Resolved	27581	-	16000	Eng. Thomas Doll	1
62	1	42	35	2022-01-13	2022-01-13	A tornolt fényszórók ereje gyenge	General electronics	Az tornolt fényszóró izzóna cserélésre, verhet...	27590	2x H7 Osram Nightbreaker emelt fényerejű izzó	18000	Eng. Thomas Doll	1
63	1	42	35	2022-01-13	2022-01-13	Vásárlás után, az autóban található folyadékok	General mechanics	Folyadékok lecsere és olajcsere	27698	Hűtővíz, fákola, hajtóműalj, motorolaj, olajszűr...	86900	Eng. Thomas Doll	5
64	1	42	35	2022-01-14	2022-01-14	A hátsó kilátás növelése érdekében tolatókamera	General electronics	Utólagos univerzális tolatókamera beszerelése	27635	1x Nagy látószögű tolatókamera	30000	Bill Right	4
65	1	43	36	2021-10-05	2021-10-05	Előregedett négyvászakos gumiabroncsok	Tyre change	Előregedett gumiabroncsok lecsere	47028	4x Continental Conti/WinterContact	510000	Bill Right	1
66	1	43	36	2022-03-30	2022-03-30	Új nyári gumiabroncsok vásárlása és felük lecser...	Tyre change	Resolved	54656	4x Bridgestone Potenza Sport	280000	Bill Right	1
67	1	43	36	2022-04-27	2022-04-27	Kavicsfelverődések a gépházaton	General mechanics	Planned job	56217	-	50000	Joe Cocker	6
68	1	44	37	2021-05-13	2021-05-13	MOT	MOT	Resolved	0	-	22000	Kurt Alfredson	1
71	1	44	37	2021-08-19	2021-08-19	Felülvizsgálat rendszer használata lassú	General electronics	Az elérhető szoftverfrissítések telephelye	32091	Gyártó által publikált szoftvercsomag	10000	Joe Cocker	1
72	1	45	37	2022-02-14	2022-02-14	Check engine lámpa világít	General mechanics	Hibákód olvasás, lerakódott koksz kitakarítása a m...	495203	Motorizáló folyadék	27000	Bill Right	5
73	1	45	37	2022-02-15	2022-02-15	Check engine lámpa világít	General mechanics	Hibákód olvasás, katalizátor cserélésre	488249	Bontott katalizátor	110000	Bill Right	7
74	1	45	37	2022-02-17	2022-02-17	Nehézséges váltás, erős remegés, alapjáraton zajos m...	General mechanics	Kettős tömegű lendkerék valamint kuplung alkatrész...	498267	Kettős tömegű lendkerék, kuplung szett	230000	Eng. Thomas Doll	13
75	1	45	37	2022-04-29	2022-04-29	Kavicsfelverődések, repedések találhatók a szélvédőn...	General mechanics	Planned job	0	-	37000	Eng. Thomas Doll	6
76	1	46	38	2022-04-01	2022-04-01	A rádió recsegő hangot ad, olykor el is nézik	General electronics	Rádió antennájának kicsere	130209	Rádió antenna	29500	Joe Cocker	2
77	1	46	38	2022-05-19	2022-05-19	Zajok utiháton való áthaladáskor	General mechanics	Planned job	0	-	15000	Kurt Alfredson	1
78	1	46	38	2022-05-20	2022-05-20	MOT	MOT	Planned job	0	-	22000	Eng. Thomas Doll	1
79	1	47	39	2021-09-30	2021-09-30	Az üzemanyag szintjelző rossz mennyiséget mutat	General mechanics	Lambda szonda kicsere	86324	Lambda szonda	31600	Eng. Thomas Doll	8
80	1	47	39	2021-10-18	2021-10-18	Zörgő hang a futómű felől	General mechanics	Lengőkarok, stabilizátor rúdak, szilentek	87597	Lengőkarok, stabilizátor rúdak és szilentek	55000	Joe Cocker	5
81	1	47	39	2021-10-19	2021-10-19	Futómű hibáinak javítása	General mechanics	Leszakadt provédő gumiabroncsok lecsere	87609	provédő gumiabroncsok	30000	Joe Cocker	4
82	1	47	39	2022-02-09	2022-02-09	MOT	MOT	Resolved	88938	-	22000	Eng. Thomas Doll	1
83	1	48	39	2022-03-14	2022-03-14	Az autó elakadt a sárlan valamint nehéz terepen	Tyre change	Terel gumiabroncs jánála és felcsere	14009	4x Goodyear UltraGrip gumiabroncs	432000	Eng. Thomas Doll	1
84	1	48	39	2022-03-17	2022-03-17	Alcoscan az autó használata	General mechanics	Hasmagasság megemelése állítható tekeresrúrokkal	14071	4x Nissan Navara D40 +40 tekeresrúgó	480000	Bill Right	10
85	1	48	39	2022-03-18	2022-03-18	Kiegészítő világítást szerezte az autó elejére val...	General electronics	Emelt fényerejű, univerzális LED-es lámpatestek be...	14102	4x 20db LED-es, emelt fényerejű lámpatest	77500	Kurt Alfredson	4
86	1	49	40	2021-08-17	2021-08-17	MOT	MOT	Resolved	175024	-	22000	Joe Cocker	1
87	1	49	40	2021-10-07	2021-10-07	A klima nem hűt megfelelően	General mechanics	Elrepedt levegőcsövek kicsere	175257	3x Levegőcső	23830	Eng. Thomas Doll	3
88	1	49	40	2021-11-03	2021-11-03	Elhasználódott gumiabroncsok	Tyre change	Elhasználódott gumiabroncsok lecsere	175257	4x LingLong GreenMax AllSeason négyvászakos gumiab...	65000	Eng. Thomas Doll	1
89	1	49	40	2022-02-08	2022-02-08	Az autó nehezen indul	General mechanics	Előregedett gyertyákkelések kicsere	175688	Gyertyákkelé szett	36000	Eng. Thomas Doll	3
90	1	50	41	2021-11-04	2021-11-04	Az autó rágat indulásnál valamint menet közben	General mechanics	Megháztámadt benzinpumpa és gyűjtőrúrok kicsere...	252800	Benzinpumpa, 4x Bosch gyűjtőrúró	130000	Bill Right	8
91	1	50	41	2021-12-15	2021-12-15	Fékzavarra nyikorgó hangot ad az autó	General mechanics	Az első valamint hátsó fékberétek és féktárcsák cs...	262948	4x Féktárcsa, 4x fékberétek	128500	Joe Cocker	3
92	1	50	41	2022-01-14	2022-01-14	A helyzetjelző izzók rendszereken kiégett	General electronics	Megháztámadt biztosíték kicsere	252998	10A-es biztosíték, 2x helyzetjelző izzó	7850	Kurt Alfredson	2
93	1	50	41	2022-03-28	2022-03-28	Kopott téli gumiabroncsok	General mechanics	Kopott téli gumiabroncsok lecsere	253921	4x Hankook Ventus Prime4	92000	Eng. Thomas Doll	1
94	1	51	42	2022-01-25	2022-01-25	MOT	MOT	Resolved	286411	-	22000	Joe Cocker	1
95	1	51	42	2022-03-23	2022-03-23	Kavicsfelverődés található a szélvédőn	General mechanics	Kavicsfelverődés kijávítása	286752	-	42000	Bill Right	4
96	1	51	42	2022-04-19	2022-04-19	Kopogó hang az autó háttulja felől	General mechanics	Jobb hátsó törett tekeresrúgó kicsere	288091	tekeresrúgó	40000	Choose...	3
97	1	52	43	2022-04-04	2022-04-04	A fedélzeti rendszer elavult	General electronics	Gyártó által jóváhagyott frissítések letölthése és ...	8725	-	10000	Eng. Thomas Doll	1
98	1	52	43	2022-04-05	2022-04-05	Hátsó parkolószensor hibás működése	General electronics	Hibás szensor kicsere	8788	Hátsó parkolóradar szensor	26000	Joe Cocker	2
99	1	52	43	2022-05-08	2022-05-08	MOT	MOT	Planned job	0	-	22000	Eng. Thomas Doll	1
100	1	53	44	2022-04-20	2022-04-20	Általános átvizsgálás	General mechanics	Autó átvizsgálása	13452	-	15000	Bill Right	1
101	1	53	44	2022-04-20	2022-04-20	Sport gumiabroncsok vásárlása	Tyre change	Gyári gumiabroncsok lecsere	13452	4x Michelin Pilot Sport 4 SUV	267000	Joe Cocker	1
102	1	53	44	2022-04-20	2022-04-20	Kiegészítők vásárlása	General mechanics	Resolved	13452	1x Type 2 gyorstöltő kábel, 1x műanyag csomagtér...	55000	Joe Cocker	1
103	1	54	45	2022-04-12	2022-04-12	Nyikorgó, fémes hang fékzavarok	General mechanics	Kopott fékberétek és tárcák lecsere	63842	4x Brembo fékberétek, 4x Brembo tárcás	100000	Eng. Thomas Doll	3
104	1	54	45	2022-04-12	2022-04-12	Új nyári abroncsok vásárlása	Tyre change	Téli abroncsok lecsere	63842	4x Continental ContiSportContact 4	432000	Joe Cocker	1
105	1	54	45	2022-05-02	2022-05-02	MOT	MOT	Planned job	0	-	22000	Joe Cocker	1
106	1	55	46	2022-04-21	2022-04-21	Eszterkai hibák kijavítása	General mechanics	Hátsó lökhárító lecsere és fényezése	87130	1x Hátsó lökhárító	97500	Joe Cocker	4
107	1	55	46	2022-04-22	2022-04-22	Váltóolaj időszakos cseréje	General mechanics	Váltóolaj lecsere	87396	Hajtóműalj	40000	Bill Right	1
108	1	55	46	2022-05-09	2022-05-09	Időszakos motorolaj és vezérléscsere	General mechanics	Planned job	0	-	169500	Kurt Alfredson	5

# Kliens oldal -FRONTEND HTML /CSS készítése

A kliens oldalon egy reszponzív weboldalt készítünk a Bootstrap 5 segítségével. Az elkészítés során minimális saját .css fájl alkalmazás volt a cél.

Az oldal széles monitoron:





## BEST SERVICES

Car Repair stands for expertise, value, and professionalism - and has from the day the company began. We were the first automotive aftermarket repair company to offer a lifetime guarantee on select services, and today we guarantee that the results of our work will meet your expectations.

[Book an inspect](#)

## EXCELLENT QUALITY

Car Repair can solve almost any problem that occurs with your vehicle. From engine repairs and oil change to regular maintenance and diagnostics, you will always get reliable services from our ASE certified technicians who use the latest in automotive equipment and diagnostic software.

[Join us](#)



## Our Advantages

### All Car Makes



We provide a variety of repair and maintenance services for all car makes and models, even for exotic and vintage ones.

### Service Variety



The main principle of our work is to offer a wide range of quality car repair services and we've been doing it since our first day.

### Quality Support



Car Repair offers quality support programs for any vehicles that allow them to always stay fully functional.

[Register now](#)

### Accessories



At our car repair center, you can also buy any accessories you need for your vehicle, including car tires and filters.

## Our Services



### Card title

Some quick example text to build on the card title and make up the bulk of the card's content.



### Card title

Some quick example text to build on the card title and make up the bulk of the card's content.



### Card title

Some quick example text to build on the card title and make up the bulk of the card's content.



### Card title

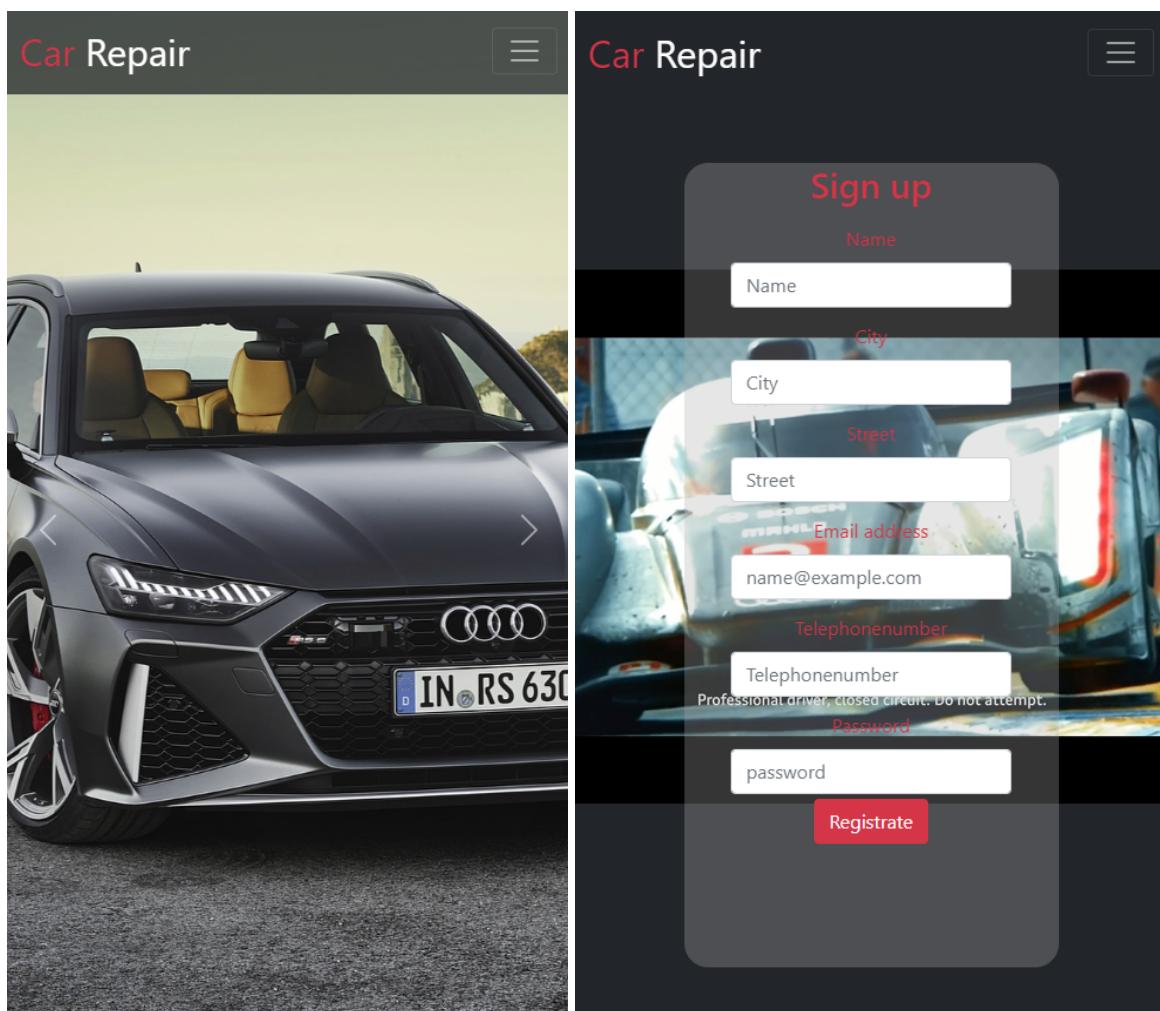
Some quick example text to build on the card title and make up the bulk of the card's content.



Az oldal eleje és a form mobil kijelzőn:

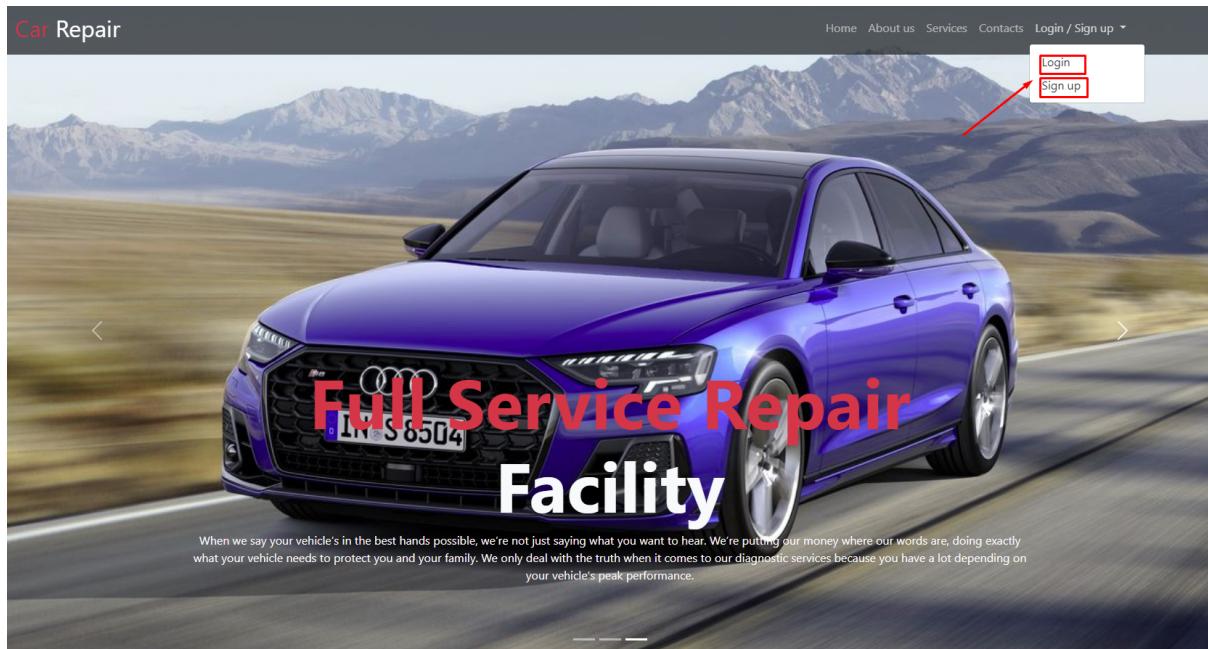
A navbar mobil méreten megváltozik, úgynevezett 'szendvics' menü lesz.

A szöveg és a kép egy oszlopba kerül, és csak az első kép töltődik be. A form is egy oszlopos lesz. A menü helyén a három csíkos gomb jelenik meg

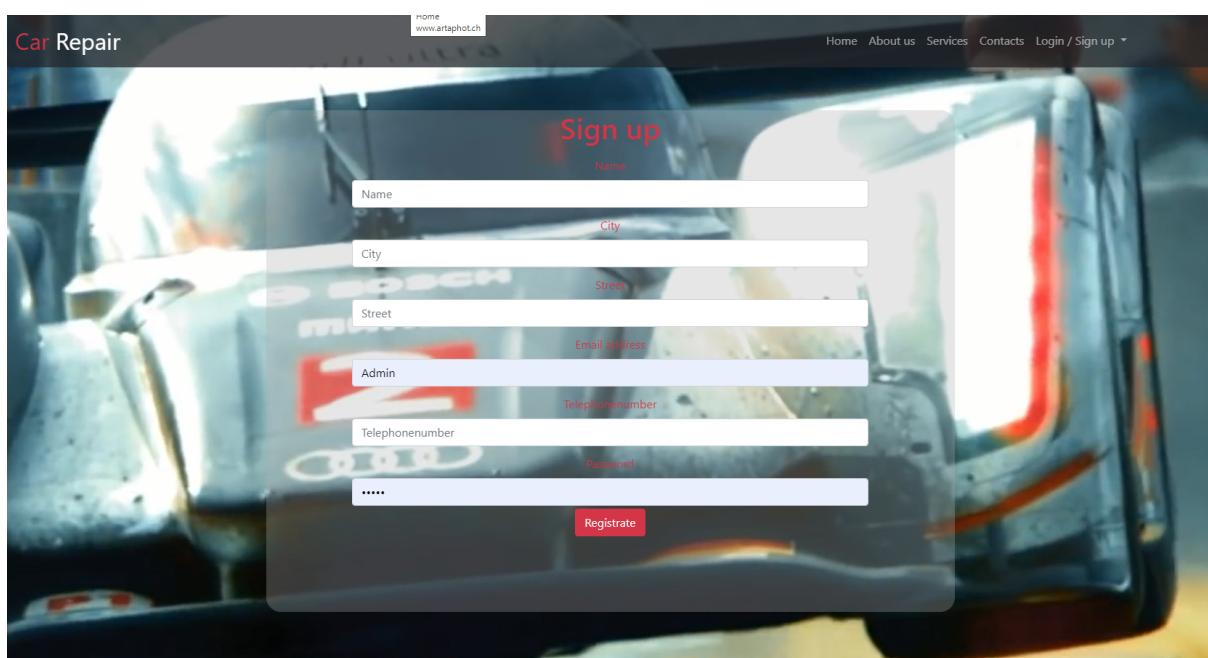


# Kliens oldal működése - FRONTEND programozás

Az index.html oldalhoz nem kapcsolódik javascript file, azonban innen tudjuk elérni mind a sign up mind a login felületet.



A login.html oldalon a /reg POST API végpontra tudunk adatot küldeni és várni a választ a szervertől. Mindezt a signup.js fájlban végezzük el



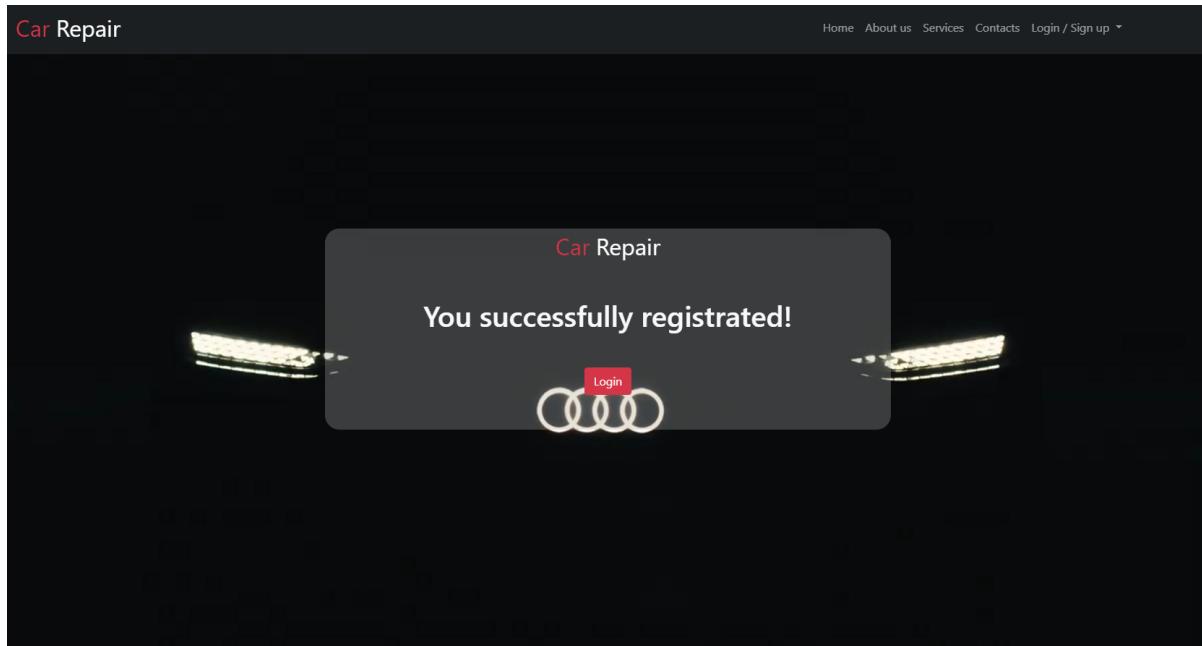
A lekérdezéshez a fetch promise-t használjuk. Elküldjük a bodyban található tárolandó adakat json formátumban. 'Aztán' szerver válaszát megkapjuk, ha 'ok' akkor választ is json formátumba alakítjuk. A szerver válaszát megjelenítjük ( ezek a válaszok a szerver/backend oldali API végponton már deklarálva vannak már) . Majd ha a szerver válasza 'ok' , a felhasználó felé visszajelzést adunk. Bár a egy üzenet megjelenik az oldal alján az 'uzenet' id html részen, a sikeres visszajelzést egy success oldalra történő irányítással szeretnénk a felhasználó számára egyértelművé tenni.

Amennyiben mégsem lenne sikeres a fetch promise, a .catch -el kezeljük, és a consol-ra kiíratjuk a hibát.

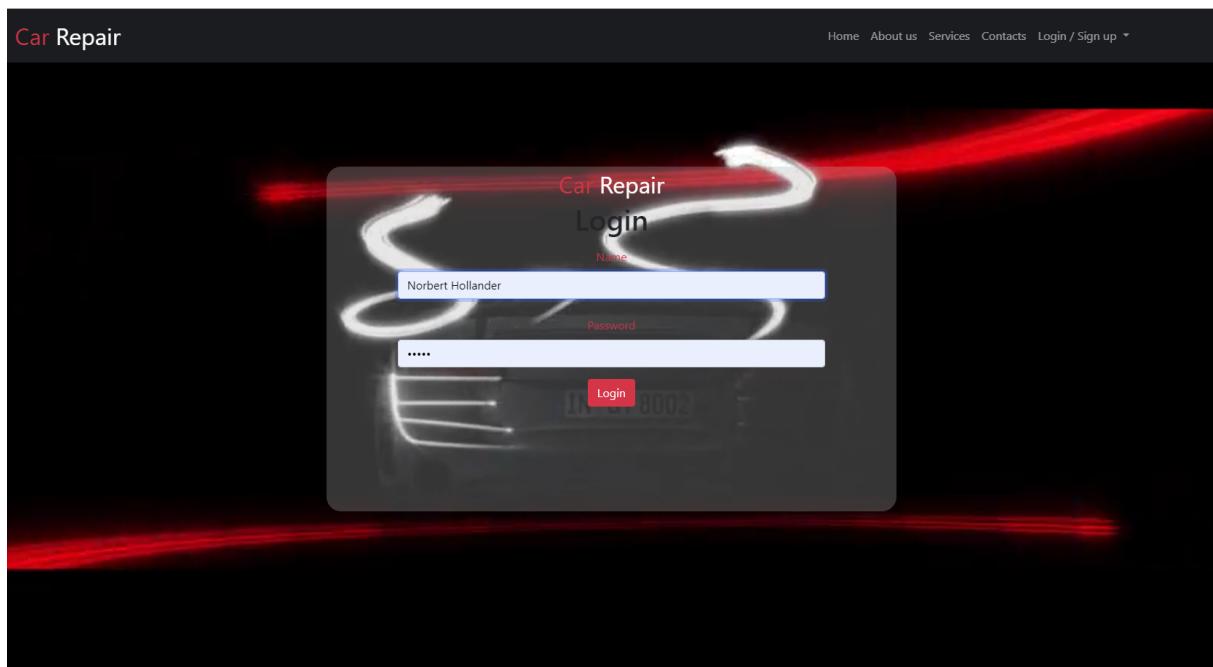
```
document.getElementById("button").onclick = function (e) {
    e.preventDefault();
    const url = 'http://localhost:5050/reg';
    fetch(url, {
        method: 'POST',
        headers: {
            'Content-type': 'application/json; charset=utf-8'
        },
        body: JSON.stringify({
            "name": document.getElementById("name").value,
            "city": document.getElementById("city").value,
            "street": document.getElementById("street").value,
            "email": document.getElementById("email").value,
            "telephone": document.getElementById("telephone").value,
            "password": document.getElementById("password").value
        })
    })

    .then((response) => {
        ok = response.ok
        return response.json()
    })
    .then(json => {
        document.getElementById("uzenet").innerHTML = json.message
        if (ok) document.location = "success.html"
    })
    .catch(err => console.log(err));
}
```

A sikeres regisztrációt megerősítő honlap:



A login.html oldalon a /login POST API végpontra tudunk adatot küldeni és várni a választ a szervertől. Mindezt a login.js fájlban végezzük el



A lekérdezéshez a fetch promise-t használjuk. Elküldjük a bodyban található name és password adakat json formátumban. 'Aztán' szerver válaszát megkapjuk, ha 'ok' akkor választ is json formátumba alakítjuk. 'Aztán' ebből a json a sessionStorage.token-t is használjuk, hogy a böngészés során fontos adatokat majd tudjunk tárolni és használni.

A szerver válaszát megjelenítjük ( ezek a válaszok a szerver/backend oldali API végponton már deklarálva vannak már) . Majd ha a szerver válasza 'ok' a felhasználó névtől függően irányít a megfelelő oldalra, vagy a user.html vagy az admin.html oldalra.

```
document.getElementById("button").onclick = function (e) {
    e.preventDefault();
    let ok = false
    const url = 'http://localhost:5050/login';
    fetch(url, {
        method: 'POST',
        headers: {
            'Content-type': 'application/json; charset=utf-8'
        },
        body: JSON.stringify({
            "name": document.getElementById("name").value,
            "password": document.getElementById("password").value
        })
    })
    .then((response) => {
        ok = response.ok
        return response.json()
    })

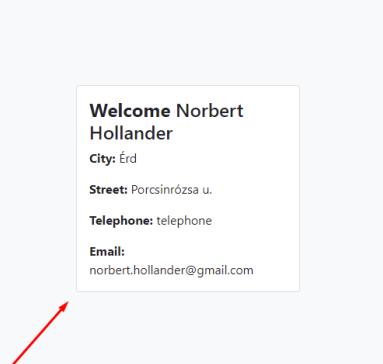
    .then(json => {
        sessionStorage.token = json.token
        document.getElementById("uzenet").innerHTML = json.message;
        if (ok){
            if (document.getElementById("name").value=="Admin") document.location = "admin.html";
            else document.location="user.html";
        }
    })
    .catch(err => console.log(err));
}

e.preventDefault();
```

A fenti sorral azt érjük el, hogy a függvény során nem tölti be az oldalt újra be, hanem a változások, az oldal változása nélkül jelenik meg.

## Felhasználói felület

A user.html oldalon a /user GET API végpontról tudunk adatot lekérni a user.js fileban található load() függvénnyel.



```

function load() {
  const url = 'http://localhost:5050/user'
  fetch(url, {
    method: 'GET',
    headers: {
      'Authorization': token
    }
  })
  .then((response) => response.json())
  .then(json => {
    let user = json[0]
    document.getElementById("username").innerHTML = "<b>Welcome</b> " + user.name
    document.getElementById("Address1").innerHTML = "<b>City: </b>" + user.city
    document.getElementById("Address2").innerHTML = "<b>Street: </b>" + user.street
    document.getElementById("Telephone").innerHTML = "<b>Telephone: </b>" + user.telephone
    document.getElementById("Email").innerHTML = "<b>Email: </b>" + user.email
  })
  .catch(err => console.log(err));
}

```

A függvényben a fetch promise részeként elküldjük a headersben a tokent- amit kaptunk a bejelentkezéskor és tárolunk a böngészőben.

```
const token = 'Bearer: ' + sessionStorage.token
```

A sikeres lekérdezéskor a kapott adatokat a card-body html részre töltjük be. A beöltés állandó legyen ezért a user.js file-ban meghívjuk a load() függvényt.

A card-body html része

```

<div class="card-body">
  <h4 class="card-title" id="username"></h5>
  <p class="card-text" id="Address1"></p>
  <p class="card-text" id="Address2"></p>
  <p class="card-text" id="Telephone"></p>
  <p class="card-text" id="Email"></p>
</div>
:
```

Autónkat a /addCar POST API végpontra történő parancssal tudjuk feltölteni melyet a Add felirattal ellátott ‘buttonCar’ id-jű gomb megnyomásával indítunk el egy függvény segítségével. Az adatokat egy formban adjuk meg.

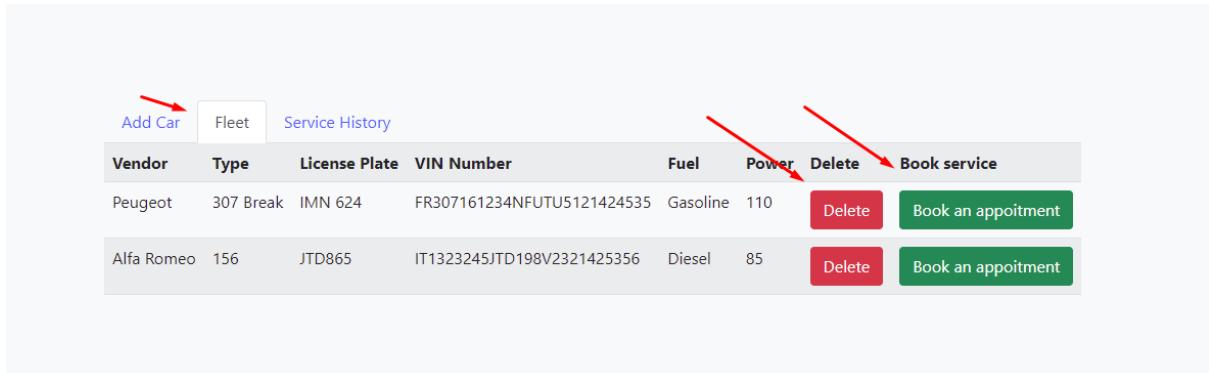
The screenshot shows a web-based car management system. At the top, there are three navigation tabs: 'Add Car' (which is active and highlighted in red), 'Fleet', and 'Service History'. Below these tabs is a form for entering car details. The form consists of several input fields and dropdown menus. The 'Vendor' field is empty. The 'Type' field is also empty. The 'License Plate' field is empty. The 'Cubic Capacity' field is empty. The 'Power' field is empty. The 'Fuel' dropdown menu is set to 'Gasoline'. The 'VIN Number' field is empty. At the bottom left of the form area, there is a red rectangular button labeled 'Add Car'. A red arrow points from the bottom left towards this button, indicating where the user should click to trigger the function.

A függvény lefutása végén a form adatait töröljük.

```
// Autó hozzáadása
document.getElementById("buttonCar").onclick= function(e) {
    e.preventDefault();
    const url = 'http://localhost:5050/addcar';
    const token = 'Bearer: ' + sessionStorage.token
    fetch(url, {
        method: 'POST',
        headers: {
            'Content-type': 'application/json; charset=utf-8',
            'Authorization': token
        },
        body: JSON.stringify({
            "vendor": document.getElementById("vendor").value,
            "Type": document.getElementById("Type").value,
            "LicensePlate": document.getElementById("LicensePlate").value,
            "VIN_number": document.getElementById("VIN_number").value,
            "Fuel": document.getElementById("Fuel").value,
            "CubicCapacity": document.getElementById("CubicCapacity").value,
            "Power": document.getElementById("Power").value
        })
    })
    .then((response) => {
        return response.json()
    })

    .catch(err => console.log(err));
    document.getElementById("addCarForm").reset();
}
```

A flottához már hozzáadott autók adait user.html oldalon a /fleet GET API végpontról tudunk adatot lekérni a user.js fileban található függvény segítségével.A függvény a Fleet tab fül megnyomásával indul el.



Add Car	Fleet	Service History					
Vendor	Type	License Plate	VIN Number	Fuel	Power	Delete	Book service
Peugeot	307 Break	IMN 624	FR307161234NFUTU5121424535	Gasoline	110	<button>Delete</button>	<button>Book an appointment</button>
Alfa Romeo	156	JTD865	IT1323245JTD198V2321425356	Diesel	85	<button>Delete</button>	<button>Book an appointment</button>

```

// Flotta lekéréés
document.getElementById("profile-tab").onclick= function(e) {
  e.preventDefault();
  const url = 'http://localhost:5050/fleet';
  const table = document.getElementById("cars")
  const token = 'Bearer: ' + sessionStorage.token
  fetch(url, {
    method: 'GET',
    headers: {
      'Authorization': token
    }
  })
  .then((response) => response.json())
  .then(json => {
    table.innerHTML = "<tr><th>Vendor</th><th>Type</th><th>License Plate</th><th>VIN Number</th><th>Fuel</th><th>Power</th><th>Delete</th>"
    json.forEach(car => {
      table.innerHTML += "<tr><td>" + car.vendor + "</td><td>" + car.Type + "</td>" +
        "</td><td>" + car.LicensePlate + "</td><td>" + car.VIN_number + "</td><td>" + car.Fuel + "</td><td>" + car.Power + "</td>" +
        "<td><button class='button btn btn-danger text-white' " +
        "onclick='deletecar(" + car.carId + ")'>Delete</button></td>" +
        "<td><button class='button btn btn-success text-white' " +
        "onclick='servicecar(" + car.carId + ')'>Book an appointment</button></td></tr>"
    });
  })
  .catch(err => console.log(err));
}

```

A flotta adait tartamazó tábla méretét, a json formátumban kapott adatok hossza határozza meg, mivel a json.forEach addig halad és készíti a tábla elemeit amíg van adat. A 'cars' id tábla html része, a könnyebb láthatóság és értelmezhetőség miatt a tábla felválta színes:

```





```

Az autó törlése a html táblázatból illetve a carrepair/fleet adatbázis táblából a /fleet/deletecar/:id DELETE API végpontra történő parancssal tudjuk elérni , ennek a deletecar() függvénye amely ezt a folyamatot végrehatja, ha DELETE feliratú gombot megnyomjuk:

```

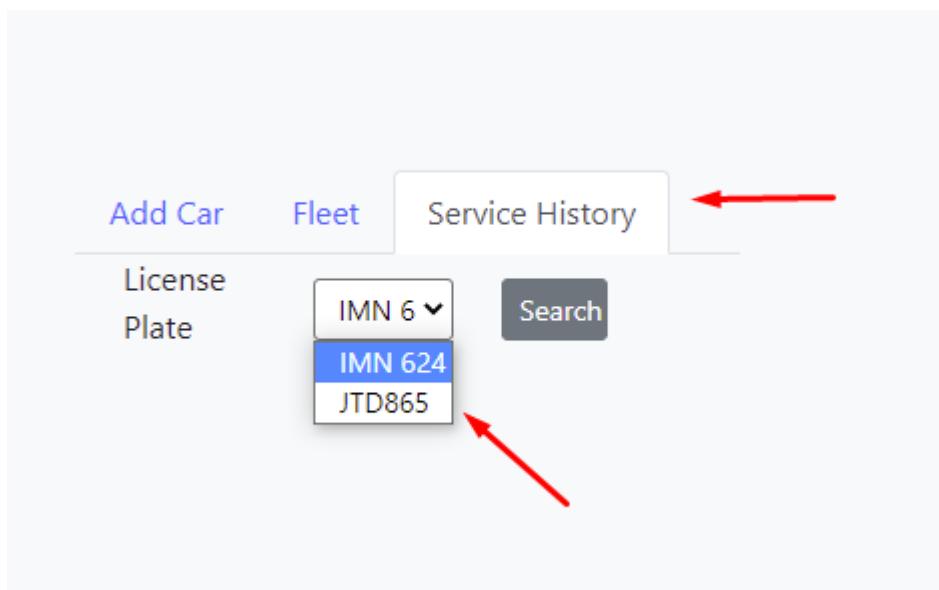
// Törölni az autót a flottából
function deletecar(carId) {
  if (confirm("Are you sure to delete this car from your Fleet?")) {
    deleteservice(carId);
    fetch('http://localhost:5050/fleet/deletecar/' + carId, {
      method: 'DELETE',
      headers: {
        'Authorization': token
      }
    })
    .then(res => {
      document.location = "user.html"
    })
    .catch(err => console.log(err));
  }
}

```

Az autó szervizre történő bejelentése Book an appointment gomb megnyomásával történik , ezzel a /service/:id POST API végpontot érjük el parancs küldéssel. Ezt a servicecar(carID) függvénytel valósítjuk meg. A függvény futása során a fetch promise elküldése után egy alert ablakban visszajelzik a felhasználónak hogy a szerviz bejelentkezés megtörtént.

```
// Szervizbe egy találkozó regisztrálása
function servicecar(carId) {
    const token = 'Bearer: ' + sessionStorage.token
    if (confirm("Are you sure to book an service appoitment for this car?")) {
        fetch('http://localhost:5050/service/' + carId, {
            method: 'POST',
            headers: {
                'Authorization': token
            }
        })
        .then(res => {
            alert("Service appoitment registered");
        })
        .catch(err => console.log(err));
    }
}
```

A Service History fül megnyomásával a /history GET API végpontra küldött kéréssel a rendszámtáblákat tartalmazó select elem feltöltése történik a szerverről lekért adatokkal.



A függvény amely ezt elvégzi:

```

// Szerviz bejegyzésekhez rendszám választás ( autó )

document.getElementById("profile-tab2").onclick= function (e) {
    e.preventDefault;
    const url = 'http://localhost:5050/history';
    const select = document.getElementById("historyPlate");
    select.innerHTML="";
    const token = 'Bearer: ' + sessionStorage.token
    fetch(url, {
        method: 'GET',
        headers: {
            'Content-type': 'application/json;charset=utf-8',
            'Authorization': token
        }
    })
    .then((response) => response.json())
    .then(json => {
        json.forEach(car => {
            i=0;
            select.innerHTML += "<option value='"+ car.carId +"'>"+ car.licensePlate+""
        });
    })
    .catch(err => console.log(err));
}

```

A select elem option elemei a json formátumban kapott adat mennyiségtől függ. Itt is a json.forEach-t használjuk.

A html részlet

```

<select id="historyPlate" class="form-select-sm w-100 " aria-label=".form-select-md example">

</select>

```

A select elem kiválasztása és a 'Search' gomb megnyomásával a /history/car POST API végpontra küldött paranccsal kapjuk meg a kiválasztott autó szerviz előzményeit. Ez a elektronikus szervizkönyv lelke.

A függvény:

```

// Szerviz történet lekérés rendszám alapján

document.getElementById("btnHistory").onclick= function(e) {
    e.preventDefault();
    const url = 'http://localhost:5050/history/car';
    const table = document.getElementById("History");
    const token = 'Bearer: ' + sessionStorage.token
    fetch(url, {
        method: 'POST',
        headers: {
            'Content-type': 'application/json;charset=utf-8',
            'Authorization': token
        },
        body: JSON.stringify({
            "historyPlate": document.getElementById("historyPlate").value
        })
    })
    .then((response) => response.json())
    .then(json => {
        table.innerHTML = "<tr><th>Date</th><th>Mileage</th><th>History ID</th><th>Vendor</th><th>Type</th><th>License Plate</th><th>Job Type</th><th>Workhours</th><th>Problem</th><th>Resolution</th><th>Mechanic</th><th>Parts</th><th>Total</th></tr>";
        json.forEach(history => {
            table.innerHTML += "<tr><td>" + history.dateStart + "</td><td>" + history.mileage + "</td><td>" + history.ticketId + "</td><td>" +
            "<td>" + history.jobType + "</td><td>" + history.workhours + "</td><td>" + history.problem + "</td><td>" + history.jobDone + "</td>" +
            "<td>" + history.parts + "</td><td>" + history.totalSum + "</td></tr>";
        });
    })
    .catch(err => console.log(err));
}

```

Itt is egy üres táblát töltünk fel a json formátumban kapott adatokkal a json.forEach metódussal.

A felhasználók a logout() függvénytelével tudnak kilépni. A kiléptetés során a login oldalra irányít, és törli a böngészőből az előző belépés token-jét.

```
function logout() {
    delete sessionStorage.token;
    document.location = "login.html";
}
```

## Admin felület

The screenshot shows a web application titled 'Car Repair' with a dark header bar. On the right side of the header is a red 'Logout' button. Below the header, there are three tabs: 'Live Tickets' (which is selected), 'Worksheet', and 'Search History'. The main content area contains a table with the following columns: Ticket ID, Name, Vendor, Type, LicensePlate, VIN\_number, and Delete. There is one row of data in the table:

Ticket ID	Name	Vendor	Type	LicensePlate	VIN_number	Delete
109	Szentpéteri Gizella	Citroen	C4	LOA-887	2HGES15252H603204	<button>Delete</button>

A felületet bejelentkezés során Admin /admin user/password kombinációval tudjuk elérni. A betöltött admin.html oldal működését és az API végpontok elérését a admin.js javascript file segíti.

A Live Tickets fül töltődik be amely a /service GET API végpontra történő kéréssel kapjuk meg a még le nem zárt , azaz várakozó munkalapokat és azok adatait. Ezt a load() függvény meghívása az admin.js file elején. A load() függvény:

```
//Élő munkalapok betöltése

function load () {
    const url = 'http://localhost:5050/service';
    const table = document.getElementById("livetickets");
    const token = 'Bearer: ' + sessionStorage.token
    fetch(url, {
        method: 'GET',
        headers: {
            | 'Authorization': token
        }
    })
    .then((response) => response.json())
    .then(json => {
        table.innerHTML = "<tr><th>Ticket ID</th><th>Name</th><th>Vendor</th><th>Type</th><th>LicensePlate</th><th>VIN_number</th><th>Delete</th>"
        json.forEach(ticket => {
            table.innerHTML += "<tr><td>" + ticket.ticketId + "</td><td>" + ticket.name + "</td>" +
                "<td>" + ticket.vendor + "</td><td>" + ticket.type + "</td><td>" + ticket.Licenseplate + "</td><td>" + ticket.vin_number +
                "<td><button class='button btn btn-danger text-white' '"
                + 'onclick="deleteticket(' + ticket.ticketId + ')">Delete</button></td></tr>"
        });
    })
    .catch(err => console.log(err));
}
```

A függvény futása során a json formátumban kapott adatokkal feltöltünk egy üres táblázatot.

A ‘DELETE’ gomb megnyomásával az adott le nem zárt munkalapot ki tudjuk törölni, ezt a /fleet/deleteticket/:id DELETE API végpontra küldött parancssal érjük el, ezt a deleteticket(ticketId) függvényel érjük el.

```
//Ticket törlése

function deleteticket(ticketId){
    fetch('http://localhost:5050/fleet/deleteticket/' + ticketId, {
        method: 'DELETE',
    })
    .then(res => {
    })
    .catch(err => console.log(err));
    load();
}
```

A Worksheet fül megnyomásával az alábbi felület érjük el:

The screenshot shows the 'Worksheet' tab of a service management application. The form fields include:

- Ticket Number: Ticket No. (input field)
- Date Range: Start (mm/dd/yyyy input field) and End (mm/dd/yyyy input field)
- Workhour: Workhour (input field)
- Job Type: Choose... (dropdown)
- Mileage: Mileage (input field)
- Service Technician: Service Technician (Choose... dropdown)
- Required job: Problems, errors reported by Client (text area)
- Resolution: Resolved job (text area)
- Parts: (input field)
- Total Charge: (input field)
- Buttons: Done (red button) and a red arrow pointing to it.

A látható formot kitölve , majd a ‘DONE’ gombot megnyomva a /worksheet POST API végpontra küldött adatokkal lezárjuk a munkalapot. A függvény a ‘Done’ gomb megnyomásával indul el. A függvény lefutása elején egy megerősítést kér, hogy biztosan kitöltöttünk-e minden részletet.

```
// Worksheet kitöltése

document.getElementById("buttonTicket").onclick= function(e) {
    e.preventDefault();
    if (confirm("Are you sure about everything is filled?")) {
        const url = 'http://localhost:5050/worksheet';
        const token = 'Bearer: ' + sessionStorage.token
        fetch(url, {
            method: 'POST',
            headers: {
                'Content-type': 'application/json; charset=utf-8',
                'Authorization': token
            },
            body: JSON.stringify({
                "Ticketnumber": document.getElementById("Ticketnumber").value,
                "startDate": document.getElementById("startDate").value,
                "endDate": document.getElementById("endDate").value,
                "Workhour": document.getElementById("Workhour").value,
                "jobType": document.getElementById("jobType").value,
                "Mileage": document.getElementById("Mileage").value,
                "Technician": document.getElementById("Technician").value,
                "Problem": document.getElementById("Problem").value,
                "Job": document.getElementById("Job").value,
                "Parts": document.getElementById("Parts").value,
                "Total_Charge": document.getElementById("Total_Charge").value
            })
        })
        .then((response) => {
            return response.json()
        })
    }
}
```

Az összes szerviz munkalapot a Service history fülel érjük el, a fül megnyomásával töltődik be a függvény amely a /servicehistory GET API végpontról kér le adatokat. A függvény neve loadhistory().

```
//Összes munkalapok betöltése

function loadhistory () {
    const url = 'http://localhost:5050/servicehistory';
    const table = document.getElementById("historytable")
    const token = 'Bearer: ' + sessionStorage.token
    fetch(url, {
        method: 'GET',
        headers: {
            'Authorization': token
        }
    })
    .then((response) => response.json())
    .then(json => {
        table.innerHTML = "<tr><th>Ticket ID</th><th>Name</th><th>Job Type</th><th>Vendor</th><th>Type</th><th>License Plate</th><th>Mileage</th><th>Date</th><th>Workhours</th><th>Problem</th><th>Resolution</th><th>Mechanic</th><th>Parts</th><th>Total Sum</th></tr>" + json.forEach(ticket => {
            table.innerHTML += "<tr><td>" + ticket.ticketId + "</td><td>" + ticket.name + "</td>" + "<td>" + ticket.jobType + "</td><td>" + ticket.vendor + "</td><td>" + ticket.type + "</td><td>" + ticket.licenseplate + "</td>" + "<td><td>" + ticket.mileage + "</td><td>" + ticket.dateStart + "</td><td>" + ticket.workhours + "</td><td>" + ticket.problem + "</td><td>" + ticket.mechanic + "</td><td>" + ticket.parts + "</td><td>" + ticket.totalSum + "</td></tr>"
        });
    })
    .catch(err => console.log(err));
}
```

A felület amit kapunk a táblázat json formátumban kapott adatokkal feltöltéssel json.forEach után.

Worksheet													
Ticket ID	Name	Job Type	Vendor	Type	License Plate	Mileage	Date	Workhours	Problem	Resolution	Mechanic	Parts	Total
16	Norbert Hollander	General mechanics	Alfa Romeo	156	JTD865	171522	2022-03-27T22:00:00.000Z	2	Futómű balra húz	Futómű beállítás	Bill Right	--	25623
22	Kovács István	Tyre change	Skoda	Octavia	MTC-905	162351	2022-04-14T22:00:00.000Z	1	Kopott abroncsok	Resolved job	Eng. Thomas Doll	4x LingLong Greenmax All Season	110000
23	Kovács István	General mechanics	Skoda	Octavia	MTC-905	0	2022-05-14T22:00:00.000Z	2	Időszakos olajcsere	Planned job	Eng. Thomas Doll	5.5L 5W-30 motorolaj, olajszűrő, pollenszűrő, levegőszűrő	50000
24	Kovács István	General electronics	Skoda	Octavia	MTC-905	0	2022-05-13T22:00:00.000Z	1	Bal tompított fényszóró nem ég	Planned job	Kurt Alfredson	Osram H4 izzó (pár)	8000
25	Kovács István	General mechanics	Toyota	Yaris	SKE-704	144274	2022-03-15T23:00:00.000Z	8	Karosszéria hibák javítása	Resolved job	Joe Cocker	Első lökhárító (cseré), jobb első sárvédő (javítás), rendszámtábla keret (cseré)	130000
26	Mendi Bertalan	General mechanics	Toyota	Prius	SDW-264	367161	2022-03-10T23:00:00.000Z	4	Egyenetlen úttartás	Első stabilizátor rúdak és szilentek cseréje	Eng. Thomas Doll	Első stabilizátor rúdak és szilentek	85000
27	Mendi Bertalan	General mechanics	Toyota	Prius	SDW-264	367243	2022-03-16T23:00:00.000Z	1	Általános átvízsgálás	Resolved	Bill Right	-	15000

Az admin felületről a felhasználói felületen megismert logout() függvényteljesen történik.

//Logout funkció

```
function logout() {
    delete sessionStorage.token;
    document.location = "login.html";
}
```

# Az API elkészítése

Az API 14 végponttal rendelkezik

	GET	POST	DELETE
/reg		Regisztráció	
/login		Bejelentkezés	
/user	Felhasználói adatok lekérése		
/addCar		Autó hozzáadása	
/fleet	Flotta betöltése Fleet fülön		
/fleet/deletecar/:id			Autó flottából való törlése
/fleet/deleteservice/:id			Autó törlése a szerviztörténetből
/fleet/deleteticket/:id			Ticket törlése
/service/:id		Szervizre történő bejelentkezés	
/service	Élő ticketek betöltése		
/worksheet		Munkalap adatainak mentése	
/servicehistory	Összes szervíztörténet lekérése		
/history	Legördülő lista készítése		
/history/car		Legördülő lista alapján újabb adatok lekérése	

Az API-t Node.js alkalmazásként az Express keretrendszer segítségével készítettük el. Kódja az server mappában, a server.js fájlban található. Érdekesség hogy, mivel server.js néven van elmentve ezért a node server.js indítás helyett a npm start is megfelel a terminálban történő indításkor.

Az alkalmazásba először a következő modulokat importáltuk:

- express
- dotenv
- mysql
- cors
- jsonwebtoken jwt / autentikáció használatához
- bcrypt / hash kód létrehozása - védett api

A dotenv modulra azért volt szükség, hogy az adatbázis eléréséhez szükséges adatokat a forráskódon kívül, a .env nevű fájlban adhassuk meg. A JWT token előállításához szükséges titkos kulcsot is generáltuk, és bírtuk a .env fájlba:

```
HOST='localhost'  
PORT=3306  
DATABASE='carrepair'  
USER='root'  
PASSWORD=''  
TOKEN_SECRET=79956727caf048afda0d613e525706
```

A használathoz a dotenv modult az alkalmazás elején be kell tölteni és konfigurálni:

```
const express = require("express");  
const app = express();  
app.use(express.json());  
const cors = require("cors");  
app.use(cors());  
require('dotenv').config();
```

Ahhoz, hogy az API-t másik, tetszőleges címről (origin) el lehessen érni, a cors modult így kell beállítani:

```
const express = require("express");  
const app = express();  
app.use(express.json());  
const cors = require("cors");  
app.use(cors());  
require('dotenv').config();
```

A MySQL adatbázis használatához be kell importálni a mysql modult, és létrehozni egy poolt, amely kezeli a kapcsolatokat az adatbázissal:

```
const mysql = require('mysql');

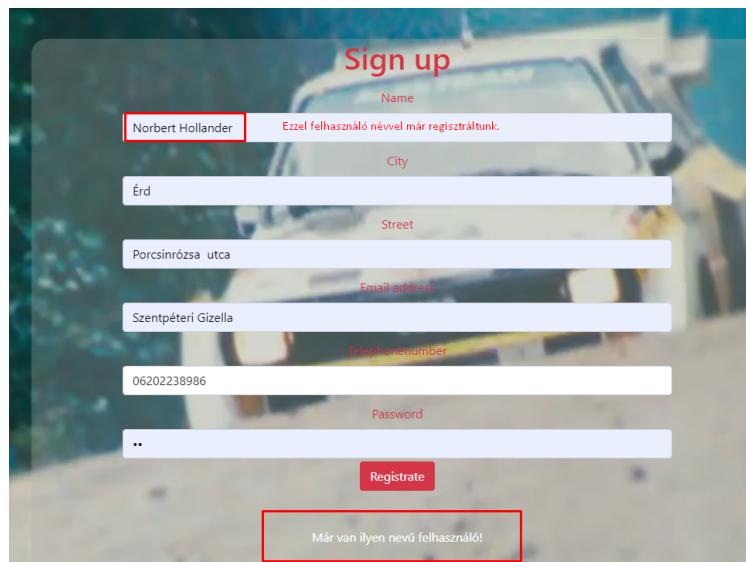
const pool = mysql.createPool({
  host: process.env.HOST,
  port:process.env.PORT,
  user: process.env.USER,
  password: process.env.PASSWORD,
  database: process.env.DATABASE
});
```

Az authentikációhoz jsonwebtoken modult importáljuk be illetve a titkosításhoz használt bcrypt hash kódot előállító modult importáljuk be.

```
const bcrypt = require("bcrypt");
const jwt = require('jsonwebtoken');
```

A regisztráció során egy POST parancsot használunk. Regisztráció során először egy olyan lekérdezést /SELECT/ végzünk mely során azt ellenőrizzük, hogy van-e már ilyen felhasználó, amennyiben ez igaz, hibaüzenettel tér vissza a parancs. Még nem létező felhasználó esetén, hashkód előállítást fogjuk elvégezni a felhasználó által megadott jelszóval. További MySQL lekérdezés/ INSERT/ során rögzítjük /felvisszük a felhasználó adatait.

Hibaüzenet már meglévő felhasználó esetén



## Regisztráció kódja

```
// Regisztráció
app.post('/reg', function(req, res) {
  const { name, city, street, email, telephone, password } = req.body;
  const q1 = "SELECT * FROM clients WHERE name = ?";
  pool.query(q1, [req.body.name],
    function (error, result) {
      if (error){
        return res.status(500).json({ message: "Adatbázis hiba!" });
      }
      else if(result.length > 0) {
        return res.status(400).json({ message: "Már van ilyen nevű felhasználó!" });
      }
      else{
        const hash = bcrypt.hashSync(password, 10);
        const q2 = "INSERT INTO clients (name,city,street,email,telephone,password) VALUES (?, ?, ?, ?, ?, ?)";
        pool.query(q2, [name,city,street,email,telephone, hash],
          function (error, result) {
            if (!error) {
              res.status(201).json({ message: "Sikeres regisztráció" })
            } else {
              res.send(error);
            }
          });
      }
    });
});
```

Bejelentkezéskor egy POST parancs során egy SELECT lekérdezést végzünk el, hogy ellenőrizzük, hogy van-e ilyen felhasználó. Amennyiben a lekérdezés sikeres a jelszó hashkód vizsgálata hajtjuk végre.

```
// Bejelentkezés
app.post("/login", function (req, res) {
  const { name, password } = req.body;
  const q = "SELECT * FROM clients WHERE name = ?";
  pool.query(q, [req.body.name],
    function (error, result) {
      if (error)
        return res.status(500).json({ message: "Adatbázis hiba!" });
      else if (result.length == 0) {
        return res.status(400).json({ message: "Nincs ilyen nevű felhasználó!" })
      } else{
        user = JSON.parse(JSON.stringify(result[0]));
        if (!bcrypt.compareSync(password, user.password))
          return res.status(401).json({ message: "Hibás jelszó!" })
        const token = jwt.sign(user, process.env.TOKEN_SECRET, { expiresIn: 3600 })
        res.json({ token, message: "Sikeres bejelentkezés." })
      }
    });
});
```

A védett végpontokra érkező kéréseknél meg kell vizsgálni a felhasználó által küldött tokent. Ehhez készítettünk egy middleware függvényt.

```

// Token ellenőrzése middleware-rel (forma: BEARER token)
function authenticateToken(req, res, next) {
  const authHeader = req.headers['authorization']
  const token = authHeader && authHeader.split(' ')[1]
  if (!token)
    return res.status(401).send({message: "Azonosítás szükséges!"})
  jwt.verify(token, process.env.TOKEN_SECRET, (err, user) => {
    if (err)
      return res.status(403).send({message: "Nincs jogosultsága!"})
    req.user = user
    next()
  })
}

```

Először ellenőrizzük az authorization fejlécet, és szétvágjuk a szóköznél, majd a második felét vesszük. Ezzel levágjuk az elejéről a Bearer szót és a szóközt.

Ha nem küldték a fejlécet vagy abban a tokent, akkor 401-es kódot és hibaüzenetet küldünk.

Ha megkaptuk a tokent, akkor a titkos kulcs segítségével ellenőrizzük. Ha az ellenőrzés során hiba történik (a token hibás), akkor 403-as hiba kódot küldünk.

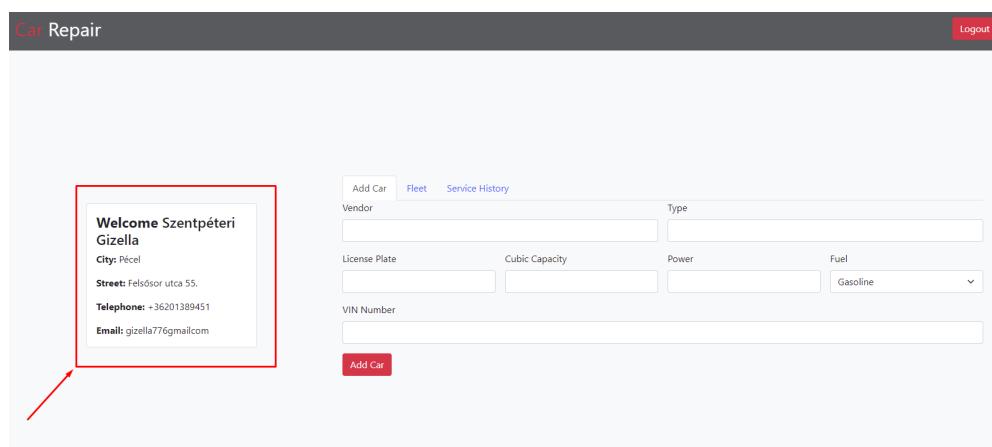
Ha jó token, akkor az abban tárolt felhasználói adatokat hozzaadjuk a kéréshez, és meghívjuk a következő függvényt.

A felhasználói felületen lekért a regisztrációkor megadott adatok betöltéséhez egy SELECT lekérdezést használunk, a GET kérésnél.

```

//Felhasználói adatok lekérése
app.get("/user", authenticateToken, function (req, res) {
  const q = "SELECT name, city, street, telephone, email "
  + "FROM clients WHERE name=?";
  pool.query(q, [req.user.name], function (error, results) {
    if (!error) {
      res.send(results);
    } else {
      res.send(error);
    }
  });
});

```



Az autó flottához történő hozzáadás egy POST parancs során INSERT MySQL lekérdezés.

```
// Auto hozzáadása

app.post("/addcar", authenticateToken, function (req, res) {
  const q = "INSERT INTO fleet (userid, Vendor, type, licenseplate, vin_number,fuel,cubiccapacity,power) "
  + "VALUES(?, ?, ?, ?, ?, ?, ?, ?)"
  pool.query(q,
    [req.user.userid,
    req.body.vendor,
    req.body.Type,
    req.body.LicensePlate,
    req.body.VIN_number,
    req.body.Fuel,
    req.body.CubicCapacity,
    req.body.Power],
    function (error, results) {
      if (!error) {
        res.send(results);
      } else {
        res.send(error);
      }
    });
})
```

Autóflotta betöltése egy GET kéréssel történik, ahol egy SELECT lekérdezés által kapjuk meg az adatokat.

```
// Autó flota

app.get("/fleet", authenticateToken, function (req, res) {
  const q = "SELECT vendor,Type,LicensePlate,VIN_number,Fuel,Power,carId FROM Fleet where userid=?";
  pool.query(q,[req.user.userid], function (error, results) {
    if (!error) {
      res.send(results);
    } else {
      res.send(error);
    }
  });
});
```

Vendor	Type	License Plate	VIN Number	Fuel	Power	Delete	Book service
Citroen	C4	LOA-887	2HGES15252H603204	Diesel	90	<button>Delete</button>	<button>Book an appointment</button>

Ezen a felületen további végpontokat tudunk elérni. Tudjuk törlni az autókat, vagy az autókat bejelenti szervizre.

Ezek kódja

```

// Auto törlése a flottából

app.delete("/fleet/deletecar/:id", authenticateToken, function (req, res) {
  const q = "DELETE FROM fleet WHERE carId=?";
  pool.query(q, [req.params.id], function (error, results) {
    if (!error) {
      res.send(results);
    } else {
      res.send(error);
    }
  });
});

```

Amennyiben az autót töröljük, törölnünk kell a szerviztörténetből is az adott autóhoz tartozó szerviz bejegyzéseket. Ez a két MySQL tábla ( fleet és worksheet tábla) idegen kulcs beállításai miatt kötelező, ellenkező esetben SQL hiba történik. Ezeket a parancsokat DELETE parancs során DELETE lekérdezést hajtunk végre.

```

// Auto törlése a szerviztörténetből

app.delete("/fleet/deleteservice/:id", function (req, res) {
  const q = "DELETE FROM worksheet where carId=?";
  pool.query(q, [req.params.id], function (error, results) {
    if (!error) {
      return
    } else {
      return
    }
  });
});

```

Autó bejelentése szervizre, egy POST parancs során INSERT lekérdezéssel tudjuk elvégezni:

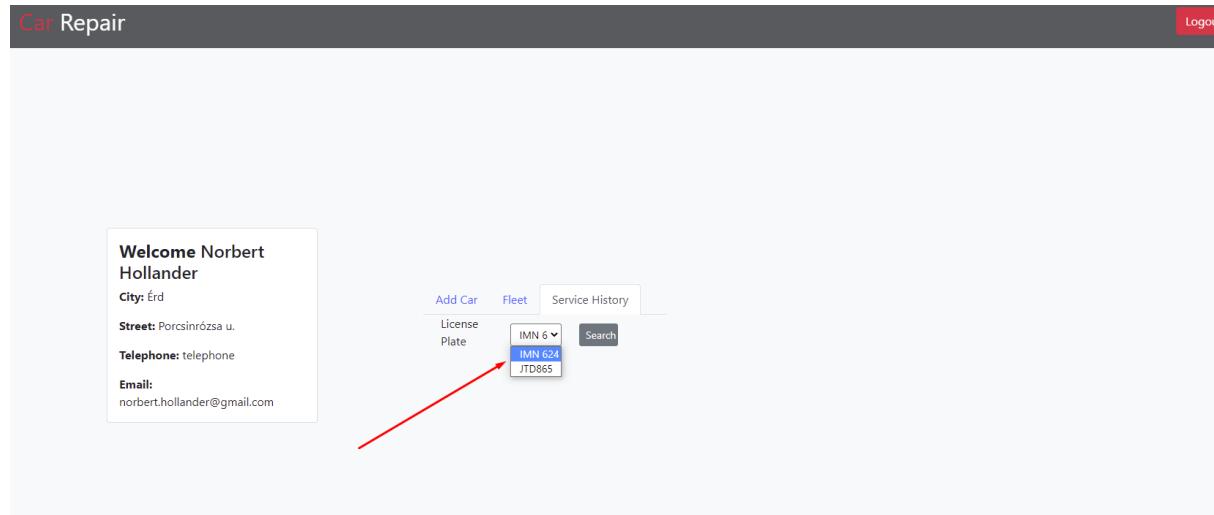
```

// Auto regisztrálása szervizre

app.post("/service/:id", authenticateToken, function (req, res) {
  const q = "INSERT INTO worksheet (userid,carId) "
    + "VALUES(?,?)"
  pool.query(q,
    [req.user.userid,
     req.params.id],
    function (error, results) {
      if (!error) {
        res.send(results);
      } else {
        res.send(error);
      }
    });
});

```

Autók szerviz történetének lekérdezése előtt ki kell választanunk egy listából a megfelelő autót. Több autó esetén ez a lista több autót tartalmaz. Ez már egy GET kérés során SELECT lekérdezés eredménye.




---

//Felhasználó autó szerviz története-rendszámok lekérdezése legördülő listához

```
app.get("/history", authenticateToken, function (req, res) {
  const q = "SELECT LicensePlate,carId FROM Fleet where userid=?";
  pool.query(q,[req.user.userid], function (error, results) {
    if (!error) {
      res.send(results);
    } else {
      res.send(error);
    }
  });
});
```

Amennyiben kiválasztottuk a rendszámot, egy újabb GET kérés során egy több táblás összetett SELECT lekérdezéssel megkapjuk az autó szerviz történetét.

---

// Autó szerviz története rendszámonként

```
app.post("/history/car", authenticateToken, function (req, res) {
  const q = "SELECT worksheet.dateStart,worksheet.mileage,worksheet.ticketId, fleet.vendor,fleet.type,fleet.Licenseplate,worksheet.jobType, "
  +" worksheet.workhours, worksheet.problem, worksheet.jobDone, worksheet.mechanic, worksheet.parts, worksheet.totalSum "
  +" FROM worksheet JOIN fleet ON fleet.carid=worksheet.carid"
  +" WHERE fleet.carId=?"
  pool.query(q, [req.body.historyPlate],function (error, results) {
    if (!error) {
      res.send(results);
    } else {
      res.send(error);
    }
  });
})
```

## Adminisztrátor beléptetése

Az adminisztrátor bejelentkezése nem egy külön bejelentkezési felületen történik, hanem a felhasználói nevek között az 'Admin' név kiemelt szerep és admin jelszóval betud lépni. Ezt a kliens oldali login.js fájlban végezzük, és egyben átirányítjuk az admin.html oldalra.

```
document.getElementById("button").onclick = function (e) {
    e.preventDefault();
    let ok = false
    const url = 'http://localhost:5050/login';
    fetch(url, {
        method: 'POST',
        headers: {
            'Content-type': 'application/json; charset=utf-8'
        },
        body: JSON.stringify({
            "name": document.getElementById("name").value,
            "password": document.getElementById("password").value
        })
    })
    .then((response) => {
        ok = response.ok
        return response.json()
    })

    .then(json => {
        sessionStorage.token = json.token
        document.getElementById("uzenet").innerHTML = json.message;
        if (ok){
            if (document.getElementById("name").value=="Admin") document.location = "admin.html";
            else document.location="user.html";
        }
    })
    .catch(err => console.log(err));
}
```

Az adminisztrátori felület legelső részlete egy újabb API végpont.

Az összes élő azaz még nem lezárt munkalap/ticket lekérdezése illetve ezek törlése, lemondott szervizelés esetén.

Ticket Data						
Ticket ID	Name	Vendor	Type	LicensePlate	VIN_number	Delete
109	Szentpéteri Gizella	Citroen	C4	LOA-887	2HGES15252H603204	<button>Delete</button>

A ticketek lekérdezése egy GET kérés során egy SELECT lekérdezéssel érjük el.

```
// Elő ticketek mutatása adminnak
app.get("/service", authenticateToken, function (req, res) {
  const q = "SELECT worksheet.ticketId, clients.name, fleet.vendor,fleet.type,fleet.Licenseplate,fleet.vin_number "
    +" FROM worksheet JOIN fleet ON fleet.carid=worksheet.carid"
    +" JOIN clients ON clients.userId = worksheet.userId"
    +" WHERE worksheet.Jstatus =0";
  pool.query(q, function (error, results) {
    if (!error) {
      res.send(results);
    } else {
      res.send(error);
    }
  });
});
```

Az élő ticket törlését DELETE kéréssel és DELETE MySQL parancsal érjük el.

```
// Ticket törlése a szerviztörténetből

app.delete("/fleet/deleteticket/:id", function (req, res) {
  const q = "DELETE FROM worksheet where ticketid=?";
  pool.query(q, [req.params.id], function (error, results) {
    if (!error) {
      return
    } else {
      return
    }
  });
});
```

A Worksheet fülön további API végponttal tudjuk a munkalapokat lezárni. A lezáráshoz a megfelelő adatokat ki kell tölteni. Végül egy gombnyomással egy POST parancsal egy UPDATE SET MySQL parancsot hajtunk végre. A parancs során nem csak az adatokat változtatjuk meg, hanem a ticket státuszát is átállítjuk 1'-re, az a már nem élő ticket, így az ott nem jelenik meg.

```
// Munkalap kitöltés

app.post("/worksheet", authenticateToken, function (req, res) {
  const q = "UPDATE worksheet SET Jstatus='1', dateStart=?, dateEnd=?, workhours=?,jobType=?,mileage=?,mechanic=?,Problem=?,jobDone=?,Parts=?,
  pool.query(q,
  [
    req.body.startDate,
    req.body.endDate,
    req.body.Workhour,
    req.body.jobType,
    req.body.Mileage,
    req.body.Technician,
    req.body.Problem,
    req.body.Job,
    req.body.Parts,
    req.body.Total_Charge,
    req.body.Ticketnumber
  ],
  function (error, results) {
    if (!error) {
      res.send(results);
    } else {
      res.send(error);
    }
  });
});
```

Az összes munkalap megtekintését egy API végpont amelyet egy GET kéréssel érünk el , amely során egy több táblás összetett SELECT MySQL lekérdezéssel kapunk meg.Ezáltal nem csak a munkalap adatokat kapjuk meg, hanem az autó tulajdonosi és flotta részleteit tudhatjuk meg.

// Összes munkalap keresés

```
app.get("/servicehistory", authenticateToken, function (req, res) {
  const q = "SELECT worksheet.ticketId, clients.name, worksheet.jobType, fleet.vendor,fleet.type,fleet.Licenseplate, worksheet.mileage,
  +" worksheet.dateStart,worksheet.workhours, worksheet.problem, worksheet.jobDone, worksheet.mechanic, worksheet.parts,
  +" worksheet.totalSum "
  +" FROM worksheet JOIN fleet ON fleet.carid=worksheet.carid"
  +" JOIN clients ON clients.userId = worksheet.userId;";
  pool.query(q, function (error, results) {
    if (!error) {
      res.send(results);
    } else {
      res.send(error);
    }
  });
});
```

A szervert a helyi gépen az 5050-es porton futtatjuk, hogy ne ütközzön a kliensekkel:

```
app.listen(5050, () => console.log("Server started on port 5050"))
```

# Az API tesztelése

A honlap és API és Kliens oldal tervezése és programozása agilis módszerrel történt lépésről lépésre. Ezért a kliens program funkcióit “kézzel” próbáltuk ki. minden megfelelően működött. Az API végpontokat a POSTMAN programmal teszteltük. A tesztelés minden végponttal megtörtént. A tesztelés sok esetben hibakeresésre is alkalmas volt, amikor valami nem sikerült, a POSTMAN-ban a szervertől kapott válaszokkal sikerült megállapítani a hibákat.

## POSTMAN tesztelés

### Bejelenkezés:

Admin felületre történő bejelentkezés. JSON formátumban küldünk a szerverre adatot, majd választ válunk token és message.

The screenshot shows the POSTMAN interface with the following details:

- Request URL:** http://localhost:5050/login
- Method:** POST
- Body (JSON):**

```
1 {"name": "Admin",  
2   "password": "admin"}  
3  
4
```
- Response Status:** 200 OK
- Response Headers:** Status: 200 OK, Time: 92 ms, Size: 713 B
- Response Body (Pretty JSON):**

```
1 {  
2   "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  
3     eyJlc2VyavQ10jEyLCJyV11jjojQWRtaW4iLC3DaXR5IjoiQnVkyXBlc3QiLCJTdHJlZXQioiJQb3Jjc2lucsOzenNhICB1dGNhIiwiRW1haWwiOiJhZG1pbk8hZG1pbis0dSIsIlRlbGVwaG9uZSI  
4     6IJjA2MjAyMjM40Tg21iwigFZc3dvcvmQioiIkMmIkMTAkS3IuaW5YR2lHeWVuZVVpNkVmaHdUT2ZTbmRWekQ00UU0d0ZHdVNhd93WEtQSdmYkk3M1ciLCJpYXQiOjE2NTA20Tg0MjAsImV4cCI6MT  
      Y1MDcwMjAyMH0.2ujjqzH7g7UxaYjXOA05KIAsmWIGM-bhYZzw-5PDg44",  
    "message": "Sikeress bejelentkezés."
```

A tokent fell kell használnunk az authorization-hez, mivel csak ezzel tudjuk elérni a további API végpontokat.

http://localhost:5050/login

POST http://localhost:5050/login

Authorization: Bearer Token

Type: Bearer Token

Token: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.

Body:

```

1 {
2   "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
3   ey31c2vyaWQ1ojeYLCJuYWl1joiOWtaW41LCJ0aXR5IjoiQnVkyBlc3QilC3TdhJ1ZXq1oijQb3Jjc2lucsOzenNhICB1dGNhIiwiRw1haWwiOijhZG1pbkBhZG1pbisodSisIlRlbGVwaG9uZSI
4   6IjA2MjAyMjM40Tg2IwiCgFzc3dvcnQ10iIkMmIkMTAkS31uaW5YR21HeWVuZVppnkvmaHdUT22TbmRWekQ00UU0d0ZhdVNhd93WEtQSzdmykk3M1c1lCjpYXQ1oje2NTA20Tg0MjAsImV4cCI6MT
      "message": "Sikeres bejelentkezés."

```

Status: 200 OK Time: 92 ms Size: 713 B Save Response

## Admin felületen összes szerviz történet lekérése:

http://localhost:5050/servicehistory

GET http://localhost:5050/servicehistory

Authorization: Bearer Token

Type: Bearer Token

Token: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.

Body:

```

1 [
2   {
3     "ticketId": 16,
4     "name": "Norbert Hollander",
5     "jobType": "General mechanics",
6     "vendor": "Alfa Romeo",
7     "type": "156",
8     "licenseplate": "JT0866",
9     "mileage": 171522,
10    "dateStart": "2022-03-27T22:00:00.000Z",
11    "workhours": 2,
12    "problem": "Futómű balra húz",
13    "jobDone": "Futómű beállítás",
14    "mechanic": "Bill Right",
15    "status": "In Progress"

```

Status: 200 OK Time: 23 ms Size: 30.56 KB Save Response

## Sikertelen bejelentkezés

### Nem létező felhasználó

The screenshot shows a POST request to `http://localhost:5050/login`. The request body is JSON with fields `name` and `password`. The response status is 400 Bad Request, with the message "Nincs ilyen nevű felhasználó!" (No such user).

```
1 {"name": "Azanka",  
2 ... "password": "Benny Hill"  
3 }  
4
```

```
1 {  
2   "message": "Nincs ilyen nevű felhasználó!"  
3 }
```

### Nem jó jelszó:

The screenshot shows a POST request to `http://localhost:5050/login`. The request body is JSON with fields `name` and `password`. The response status is 401 Unauthorized, with the message "Hibás jelszó!" (Wrong password).

```
1 {"name": "Nozbert Hollander",  
2 ... "password": "Pamela Anderson"  
3 }  
4
```

```
1 {  
2   "message": "Hibás jelszó!"  
3 }
```

# Csapatmunka

Nagy Zsolt segített az adatbázis tervezésében és az adatbázis teszt adatokkal való feltöltése illetve az egész program tesztelése általa történt. Németh Máté a dokumentációt készítette el. A Frontend és Backend elkészítését Hollander Norbert végezte.

## Továbbfejlesztési lehetőségek

- Npm nodemailer csomag telepítésével olyan lehetőségek vannak mint pl:
  - A frontend részen a contact részen emailes üzenet küldési lehetőség bármilyen felhasználó/látogató részére
  - Szervizre történő bejelentkezéskor a regisztráció során megadott email címre megerősítő email küldése
  - Munkalap lezárása során egy email küldése a felhasználó számára a munkalap adataival
- Szűrők beállítása az adminisztrátor / összes szerviztörténet részen.
- Felhasználói oldal betöltésekor a regisztrációkor megadott adatok módosíthatósága
- Felhasználói oldalon az adatok felett avatar kép beállítási lehetőség
- Adatbázis fejlesztés: a felhasznált alkatrészek külön táblába való szervezése. Ezzel az alkatrész raktárkészlet is kezelhető lenne.

# Tartalomjegyzék

<b>A feladat leírása</b>	<b>2</b>
<b>GitHub repository</b>	<b>2</b>
<b>Az adatbázis elkészítése</b>	<b>2</b>
A clients(felhasználók) tábla adatai	3
A fleet /fotta tábla adata	4
A worksheet /munkalap tábla	5
<b>Kliens oldal -FRONTEND HTML /CSS készítése</b>	<b>8</b>
<b>Kliens oldal működése - FRONTEND programozás</b>	<b>11</b>
<b>Az API elkészítése</b>	<b>24</b>
<b>Az API tesztelése</b>	<b>35</b>
<b>Tartalomjegyzék</b>	<b>39</b>