



**SZÉCHENYI
EGYETEM**
UNIVERSITY OF GYŐR



PackX csomagküldő szolgáltatás

Projektmunka I.-II.

GKNB_INTM005

Friedrich Artúr (IGKAXE), Szakály Károly (M9H5JA), Székely Dániel (JAXC3C),
Szilágyi Dominik (TJ1WI2), Tőrek Zsombor (FD19GJ)

Győr, 2023

Tartalom

PackX csomagküldő szolgáltatás	1
Tartalom	2
Bevezetés	4
Alapinformációk	4
Csapat	4
Üzleti cél	4
Követelmények	4
Felhasználói dokumentáció	6
Regisztráció, Bejelentkezés	6
Rólunk, kontakt	7
Csomag követése	9
Felhasználói nézet	11
Csomagfeladás	11
Csomagtörténet, előzmények	12
Felhasználói eszközök	12
Hűségrendszer	13
Adatmódosítás	14
Jelszó visszaállítás	14
Fiók törlése	15
Futár felhasználói nézet	15
Csomagok kezelése	16
Automaták nézete	16
Admin nézet	17
Felhasználók kezelése	17
Automaták kezelése	18
Csomagok kezelése	19
Technikai leírás	21
Frontend	21
Alapinformációk, használt technológiák	21
Az API	21
Backend	24
Alapinformációk, használt technológiák	24
Adatbázis	25
API	26
Authentication	26
Csomagok kezelése	28
Email küldés	29
CO₂ kibocsátás számolás	29
DevOps	30
Automatizált infrastruktúra	30
Terraform	30
Ansible	31

<u>CI/CD</u>	<u>31</u>
<u>Backend</u>	<u>31</u>
<u>Frontend</u>	<u>32</u>
<u>Folyamatos kiadás (CD, continuous deployment)</u>	<u>32</u>
<u>Monitoring</u>	<u>33</u>
<u>Exporterek</u>	<u>33</u>
<u>VMAgent és VictoriaMetrics</u>	<u>33</u>
<u>Grafana</u>	<u>33</u>

Bevezetés

Alapinformációk

A projektünk célja, hogy egy átfogó csomagkövető rendszert hozzunk létre. Ezzel a megoldással szeretnénk kielégíteni egy csomag életútjában résztvevő szereplők igényeit, melybe beletartoznak:

- felhasználók
- kézbesítők
- automaták

A rendszerünk segítségével a fenti szereplők környezetbarát módon, hatékonyan képesek részt venni a csomagszállítás folyamatában.

Csapat

Az alábbi táblázatban összefoglaltuk a csapattagok adatait, illetve a projekten belüli feladatkörüket.

név	neptun-kód	e-mail	feladatkör
Friedrich Artúr	IGKAXE	artur.friedrich.harka@gmail.com	frontend
Szakály Károly	M9H5JA	karoly.szakaly2000@gmail.com	infra, devops
Székely Dániel	JAXC3C	szekelydani5g@gmail.com	frontend
Szilágyi Dominik	TJ1WI2	szilagydaminik918@gmail.com	backend
Töreky Zsombor	FD19GJ	toreky.zsombor@gmail.com	backend

Üzleti cél

A rendszer megoldást nyújthat a napjainkban felmerülő problémákra. Egyik fő célunk a karbonsemlegesség, melyet teljesen elektromos szállítóeszközök biztosítanak. Az automaták 24/7 rendelkezésre állása a kényelmet biztosítja, hogy ügyfeleink bármikor, akadálymentesített helyen férjenek hozzá a küldeményükhöz.

Követelmények

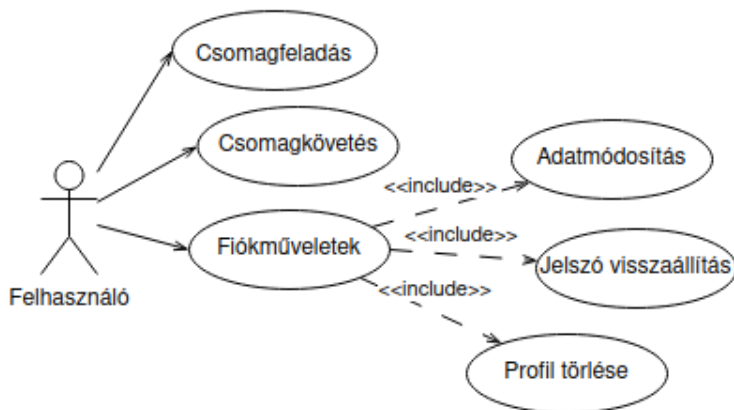
A projekttel szemben támasztott követelményeket az alábbiakban részletezzük.

Három felhasználói kör igényeinek kívánunk megfelelni:

- felhasználók
- kiszállítók/futárok

- adminisztrátorok

A legkritikusabb követelményeket a felhasználóval szemben támasztottunk, melyeket az alábbi use-case diagram szemléltet.



Látható, hogy a felhasználói funkciók közül elsődlegesen a csomagfeladás, illetve annak nyomon követése a prioritás. A különböző személyi adatokkal kapcsolatos rendelkezéseket figyelembe véve lehetőséget adunk a felhasználónak a fiókadatok módosítására, jelszavának visszaállítására, illetve profiljának törlésére is.

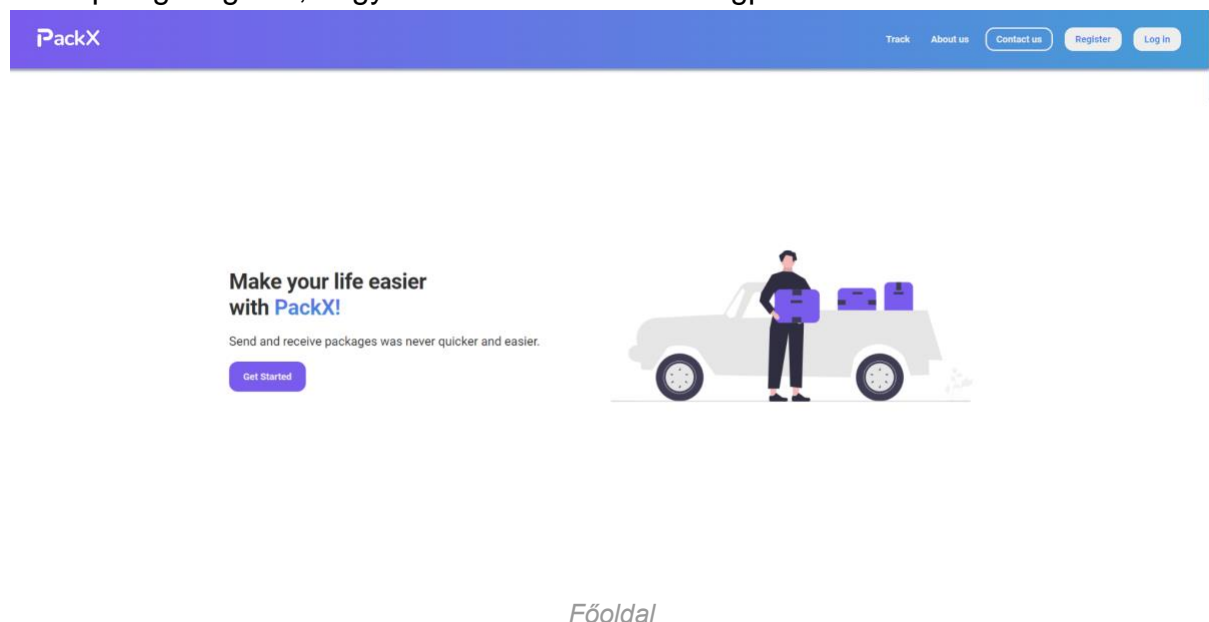
A futárok esetében kulcsfontosságú a csomagok állapotának frissítésének lehetősége. A futárok ezen felül láthatják az egyes automaták alapvető adatait.

Az adminisztrátori szinten láthatóak a csomagadatok, valamint a felhasználói adatbázis is. Az adminisztrátornak joga van ezen adatok szerkesztésére. Új automatát szintén csak ezen a felhasználói szinten van lehetőség hozzáadni.

Az említett használati eseteket bemutató folyamatokat alább, a 'Felhasználói dokumentáció' ide vonatkozó részeiben részletezzük.

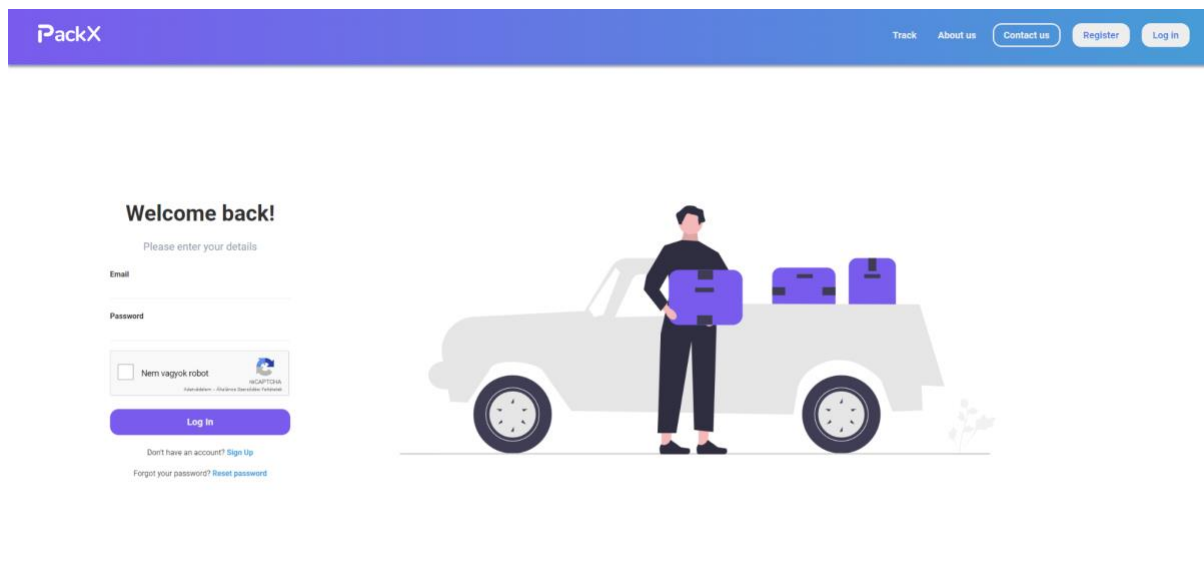
Felhasználói dokumentáció

A felhasználó egy modern, letisztult főoldalra érkezik a PackX domain címét megnyitva. A főoldalon röviden és tömören megtalálható minden kulcsfontosságú dolog a szolgáltatásról. Tudunk szállítási útiköltség díjat lekérni csomagpontok között, csomag mérettől függetlenül. Láthatjuk, hogy mennyi szén dioxiddal mentettük meg a bolygónkat azzal, hogy a PackX szolgáltatását vette igénybe, mivel 100%-ban elektromos járműveket és megújuló energia alapú infrastruktúrát használ a cég. Mindezekon kívül az oldal aljára görgetve megtekinthetjük a beágyazott Google Térkép segítségével, hogy hol találhatóak a csomagpontok.



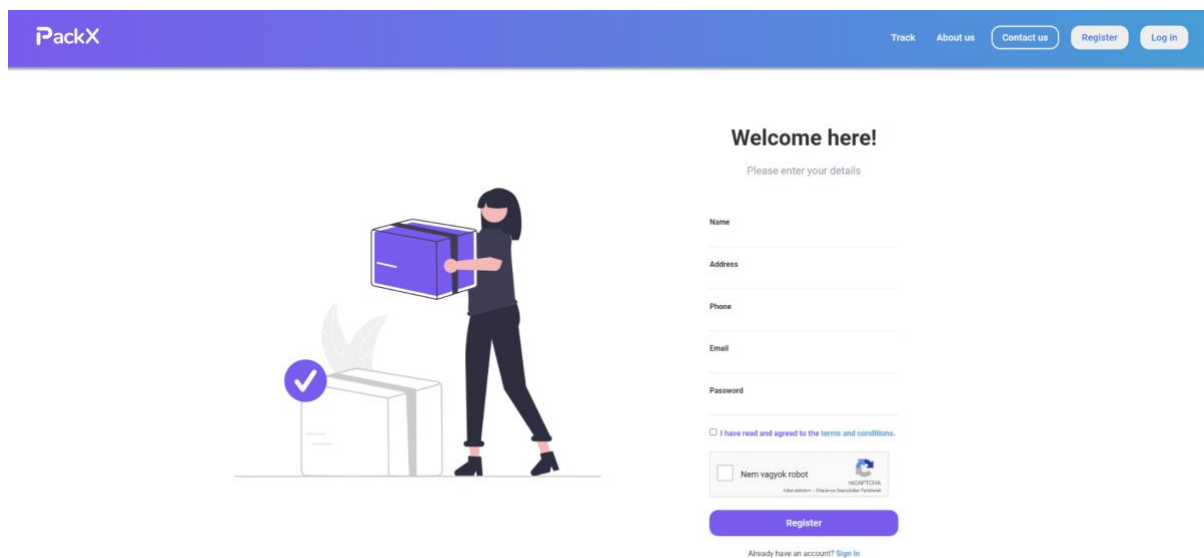
Regisztráció, Bejelentkezés

A csomagfeladás regisztrációhoz kötött. Ha már rendelkezik PackX fiókkal és csomagot szeretne feladni kérem jelentkezzen be a "Login" menüpont kiválasztásával a navigációs panelen.



“Login” page

Amennyiben csomagot szeretne feladni és még nem használta szolgáltatásunkat, kérjük regisztráljon a regisztráció gombra kattintva a navigációs panelen, vagy a “Get Started” gombra való kattintással.



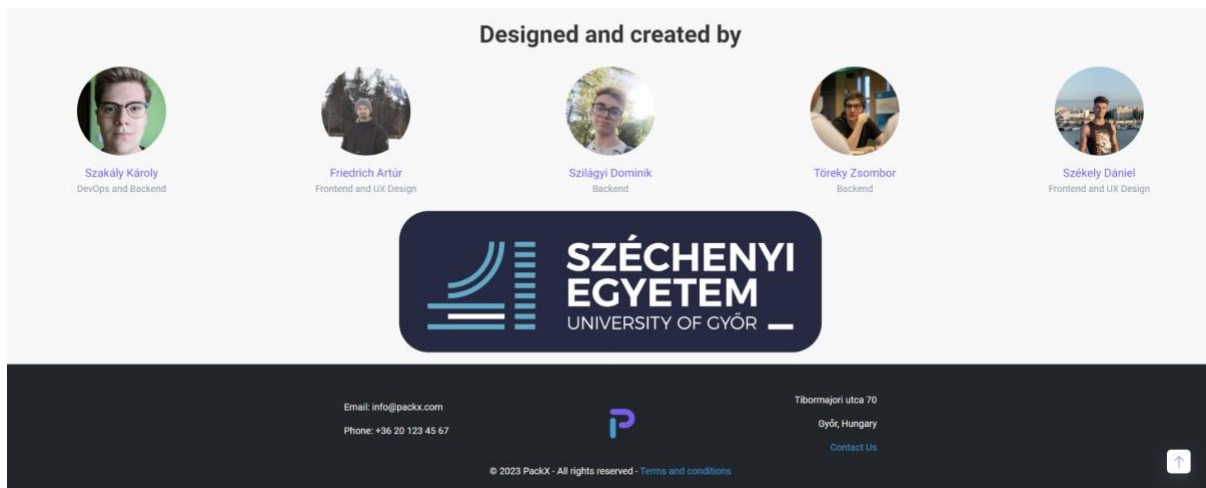
“Register” page

Rólunk, kontakt

Bejelentkezés nélküli állapotban is megtekintheti az “About us” és a “Contact us” oldalakat, valamint az oldal adatvédelmi tájékoztatóját.

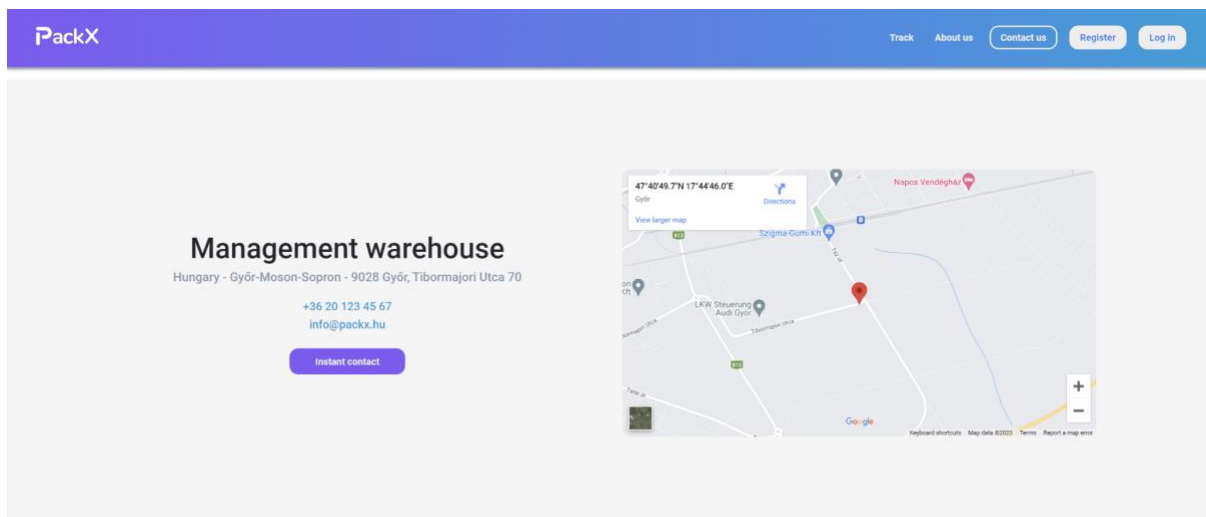
Az “About us” menüpontra kattintva a navigációs panelen a projekt leírását és összefoglalását olvashatják. Ezen az oldalon megtalálható a készítőik felsorolása és

az egyes résztvevők beosztása a projektben. Az oldalon egy rejtett “Easter Egg” is található.



“About us” oldal utolsó panele

A “Contact us” oldalon a felhasználó megtalálja a céggel kapcsolatos fő információkat, valamint intuitív módon beágyazott Google Térkép segítségével megtekinthető a cég székhelye. Az “Instant contact” gombra kattintva a egy digitális “.vcf” kiterjesztésű kontakt kártyát tudunk letölteni/betölteni a készülékünkön, megspórolva így a kontakt kártya készítést a végfelhasználónak. Így egy szempillantás alatt a készülékünkön találhatjuk a cég elérhetőségeit.



“Contact us” oldal

Az európai GDPR szabályozás rendszernek megfelelően minden felhasználó számára elérhető az adatvédelmi irányelvek oldal. Ezen oldal egy GDPR-nak megfelelő [ingyenes sablon](#) segítségével készült.

Terms and Conditions



Welcome to PackX!

These terms and conditions outline the rules and regulations for the use of PackX's Website, located at packx.hu.

By accessing this website we assume you accept these terms and conditions. Do not continue to use PackX if you do not agree to take all of the terms and conditions stated on this page.

The following terminology applies to these Terms and Conditions, Privacy Statement and Disclaimer Notice and all Agreements: "Client", "You" and "Your" refers to you, the person log on this website and compliant to the Company's terms and conditions, "The Company", "Ourselves", "We", "Our" and "Us", refers to our Company. "Party", "Parties", or "Us", refers to both the Client and ourselves. All terms refer to the offer, acceptance and consideration of payment necessary to undertake the process of our assistance to the Client in the most appropriate manner for the express purpose of meeting the Client's needs in respect of provision of the Company's stated services, in accordance with and subject to, prevailing law of hu. Any use of the above terminology or other words in the singular, plural, capitalization and/or he/she or they, are taken as interchangeable and therefore as referring to same.

Cookies

We employ the use of cookies. By accessing PackX, you agreed to use cookies in agreement with the PackX's Privacy Policy.

Most interactive websites use cookies to let us retrieve the user's details for each visit. Cookies are used by our website to enable the functionality of certain areas to make it easier for people visiting our website. Some of our affiliate/advertising partners may also use cookies.

License

Adatvédelmi irányelvek oldal

Csomag követése

A csomagkövetés nem igényel regisztrációt. Amennyiben csomagot szeretne követni, kattintson a "Track" menüpontra a navigációs panelen. A csomaghoz tartozó csomagazonosító segítségével nyomon követhetjük csomagunkat. A szürke mezőbe írja be az azonosítót és nyomjon a "Track" gombra.

Package Tracking

You can track your package status by entering your tracking number, which we send you via email.


[Track](#)

"Track" oldal

Ezután a weboldal átirányítást követően el juttat minket a csomag részletes adatait tartalmazó oldalra, ahol a csomaggal kapcsolatos információkat megtalálhatjuk. Az alábbi információkat kapjuk ezen az oldalon:

- Milyen fázisban van jelenleg a csomag

- Mi a csomag száma
- Mekkora a csomag mérete
- Mennyibe kerül a kiszállítás
- Mikorra várható a kiszállítás
- Mennyi szén-dioxidot takarítottunk meg



[Track](#)
[About us](#)
[Contact us](#)
[Register](#)
[Log in](#)

Transit

December 8, 2023

HUF ▾

Dispatch

Transit

In Warehouse

In Delivery

Delivered

Tracking number: b0y08XRZ25

Size: Large

Price: 2570 HUF

Est. delivery date: December 6, 2023

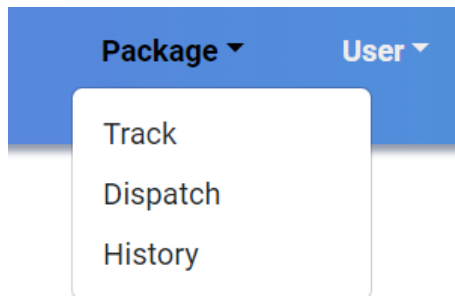
CO2 savings: 20.98 kg

Csomagkövetés oldal

Felhasználói nézet

A navigációs panelen bejelentkezést követően megjelenik két extra legördülő menüpont.

A "Package" alatt a csomagkövetésen kívül adhatunk fel csomagot a "Dispatch" gombra kattintva, valamint a múltbéli küldeményeket is megtekinthetjük.



"Package" navigációs panel almenü

Csomagfeladás

A "Dispatch" menüpontra való kattintás után a csomagfeladó oldalra jutunk, ahol a szükséges adatok megadásával adhatunk fel csomagot. Lehetőségünk van az ár 3 valutában való megjelenítésére. (*HUF, USD, EUR*) A választott valutát az oldal megjegyzi, és a továbbiakban a kiválasztott valutában mutatja az összegeket.

Az alábbi adatokat tudjuk megadni:

- A csomag induló helye
- A csomag cél helye
- A címzett neve
- A címzett e-mail címe
- A csomag mérete
- Esetleges extra információ vagy megjegyzés a futárnak
- Milyen fajta kiszállítást szeretnénk (normál, gyors, extra gyors, aznapos)

PackX

Package

User

About us

Contact us

Log out, Test

Send Package

HUF

Your email address: test1@test.hu

Your name: Test

Sender Locker

Select Sender Locker

Receiver Locker

Select Receiver Locker

Receiver Name

Receiver Email

Package Size

Small

Medium

Large

Package Size: Max size: 20x20x20 centimeters

Note

"Dispatch" oldal

Csomagtörténet, előzmények

A "History" gombra kattintva, ha már adtunk fel csomagot azelőtt, láthatjuk a csomagokat egy helyen. Ha a csomagunk még nem lépett státuszt, azaz még csak rögzítve lett a rendszerben, a felhasználó 24 órán belül még tudja törölni. A csomagokról modern letisztult kártyák segítségével kapunk információkat. A "Track" gombra kattintva pedig a csomag részletes adatait tartalmazó oldalra kerülünk, ami egyezik a sima csomagkövetési oldallal.

PackX

Package

User

About us

Contact us

Log out, Test

History

HUF

Track ID: VegHYdSYt

You saved 20.91 kilograms of CO2 with this package

Created At: 2023. 12. 08. 18:34:25

Sender Locker Address: Győr - Szent István út 23.

Destination Locker Address: Szombathely - Ehen Gyula tér 3.

Price: 2370 HUF

Delivery Date: 2023. 12. 09. 0:00:00

Note:

Track

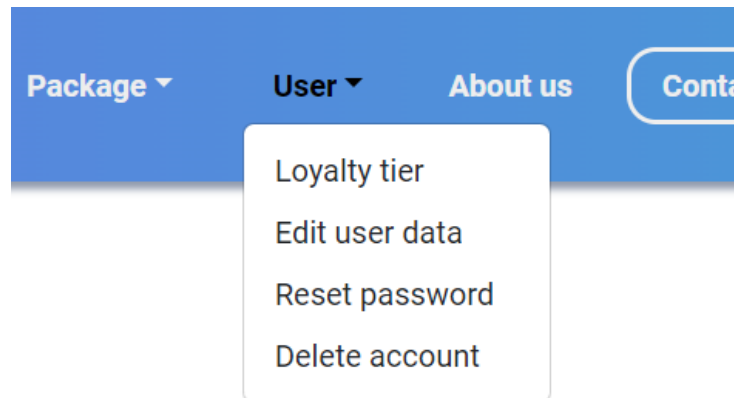
Cancel

"History" oldal

Felhasználói eszközök

A "User" legördülő menü pontra kattintás után a felhasználói fiókkal kapcsolatos funkciók tárulnak elénk. Megtekinthetjük a hűség-szintünket, változtathatunk az

adatainkon, valamint amennyiben jelszót szeretnénk változtatni, erre is van lehetőség. Ha már nem szeretnénk tovább használni a fiókot, vagy bármilyen más okból meg szeretnénk szüntetni a profilunkat, a fiók teljes mértékben, visszavonhatatlanul törölhető.



“User” navigációs panel almenü

Hűségrendszer

A PackX három hűségszintet különböztet meg az alábbiak szerint:

- 3 darab csomag után “Báró” rang, 5% kedvezmény a végösszegre
- 5 darab csomag után “Herceg” rang, 7.5% kedvezmény a végösszegre
- 7 darab csomag után “Király” rang, 10% kedvezmény a végösszegre



Loyalty Status: Regular



You are 2 order(s) away from becoming a Baron.

“Loyalty” oldal

Adatmódosítás

Az "Edit user data" menüpontra kattintva a felhasználó módosíthatja a fiókadatait. Az e-mail cím módosítására nincsen lehetőség, ezen kívül a regisztrációkor megadott adataink közül minden más adat módosítható.

The screenshot shows the 'Edit Your User Data' form in the PackX application. The form is titled 'Edit Your User Data' and includes a sub-header 'Here you can modify your account information.' The form fields are: Name (Test), Address (Test utca 1), and Phone (+36201234578). There is a checkbox for 'Nem vagyok robot' (I am not a robot) and a 'Save' button. To the right of the form are three hexagonal icons: a heart, a person, and a location pin. Further right is an illustration of a person standing next to a large, stylized letter 'S'.

"Edit user data" oldal

Jelszó visszaállítás

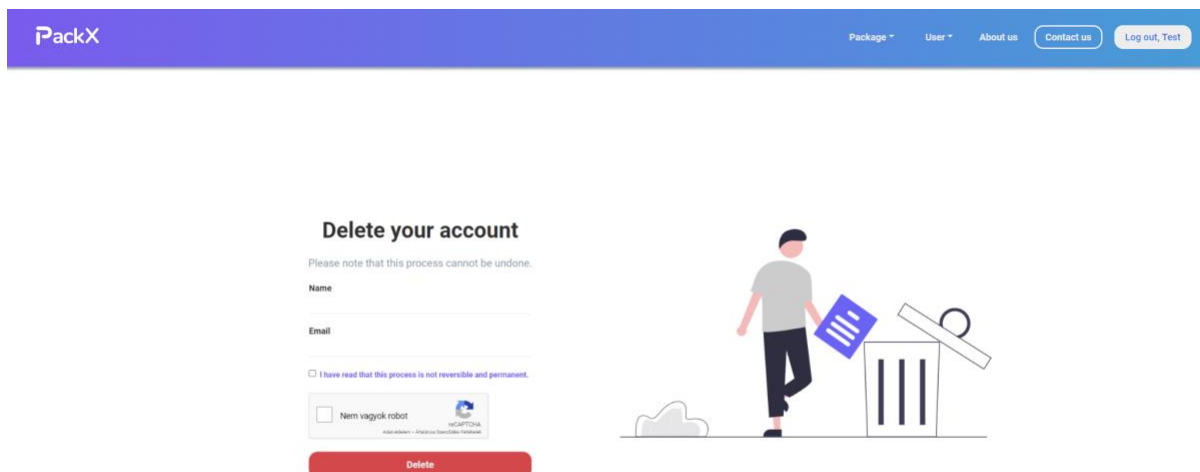
A "Reset password" gombon keresztül eljuthatunk a jelszó visszaállító felületre, ahol beállíthatunk új jelszót a fiókhoz.

The screenshot shows the 'Please enter your new login credentials' form in the PackX application. The form is titled 'Please enter your new login credentials' and includes a sub-header 'Make sure to always use a different password for security reasons.' The form fields are: Email, New password, and Confirm new password. There is a checkbox for 'Nem vagyok robot' (I am not a robot) and a 'Save' button. To the right of the form is an illustration of a person standing next to a large, stylized letter 'S'.

"Reset password" oldal

Fiók törlése

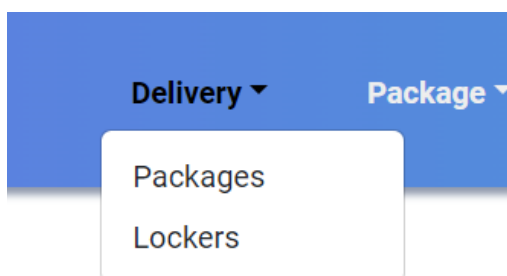
A “Delete account” oldal segítségével véglegesen törölhetjük a felhasználói fiókunkat. Ez az oldal csak sima felhasználók számára elérhető, a futár fiókok törlését az adminok kezelik.



“Delete account” oldal

Futár felhasználói nézet

Amennyiben futárok vagyunk, a navigációs panelen az eddig említett menüpontokon kívül megjelenik számunkra a “Delivery” legördülő menü. Itt két opció közül választhatunk, a “Packages” nézetben az adott futárhoz rendelt csomagokat láthatjuk. A “Lockers” gombra kattintva pedig az egyes csomag pontokkal kapcsolatban láthatunk releváns információkat. A PackX logó mellett a “Courier” kiegészítő szöveg megjelenik.



“Delivery” navigációs panel almenü

Csomagok kezelése

A "Packages" oldalon a futár tudja léptetni az adott csomag státuszát, hogy éppen a kiszállítás melyik fázisában van. Valamint a "Track" gombra kattintva megtekintheti a csomaggal kapcsolatos további információkat.

The screenshot shows the "Courier Packages" page with a header bar containing the "PackX Courier" logo and navigation links: Delivery, Package, User, About us, Contact us, and Log out, Test. The main content area displays a grid of package tracking cards. Each card includes the package ID, status, creation date, sender/destination locker address, price, delivery date, and a note. A "Track" button is present on each card, and a "Jump to next status" link is at the bottom. The currency is set to HUF.

Track ID	Status	Created At	Sender Locker Address	Destination Locker Address	Price	Delivery Date	Note
KTk2E5SDpR	Delivered	2023. 12. 05. 22:50:06	N/A	N/A	2070 HUF	2023. 12. 06. 0:00:00	Lajos
DZe56QHxOt	In Warehouse	2023. 12. 06. 11:19:33	N/A	N/A	2570 HUF	2023. 12. 06. 0:00:00	
bGy0BXRZZ5	Transit	2023. 12. 06. 11:22:45	N/A	N/A	2570 HUF	2023. 12. 06. 0:00:00	
Z0oKkk6B1H	Dispatch	2023. 12. 06. 19:46:50	N/A	N/A	2162 HUF	2023. 07. 12. 0:00:00	
VegHYdSYyT	Dispatch	2023. 12. 08. 18:34:25	N/A	N/A	2370 HUF	2023. 12. 09. 0:00:00	

"Courier Packages" oldal

Automaták nézete

A "Lockers" oldalon láthatjuk az adott csomagpontok telítettségét, megtekinthetjük, hogy hol helyezkedik el a térképen, valamint a "Show packages" gombbal ki tudjuk listázni a csomagokat amik a csomagpontban vannak éppen. A helyszíneket tudjuk rendezni név, illetve telítettség alapján is, növekvő-csökkenő sorrendbe.

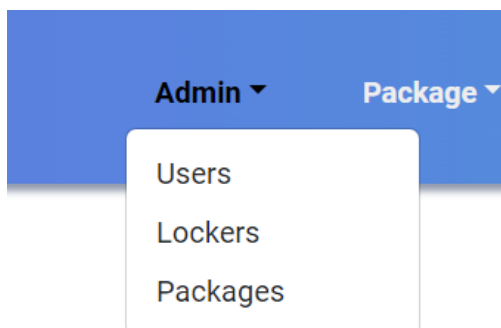
The screenshot shows the "Lockers" page with a header bar containing the "PackX Courier" logo and navigation links: Delivery, Package, User, About us, Contact us, and Log out, Test. The main content area displays a grid of locker status cards. Each card includes the locker ID, city, address, capacity, and a progress bar indicating the current status. A "Show Packages" button and a "Map" button are present on each card. The page also features sorting options: "Sort by Name" and "Sort by Fullness".

Locker ID	City	Locker Address	Capacity	Current Status
1	Győr	Szent István út 23.	4 / 7	57%
2	Győr	Kiss Ernő utca 5.	4 / 5	80%
3	Győr	Lomnic utca 30.	1 / 5	20%
4	Szombathely	Paragvári utca 74.	2 / 5	40%
5	Szombathely	Gömör utca 3.	1 / 5	20%
6	Szombathely	Éhen Gyula tér 3.	7 / 10	70%

"Lockers" oldal

Admin nézet

Amennyiben adminisztrátori fiókunk van, a navigációs panelen az eddig említett összes funkció kívül megjelenik az “Admin” legördülő menü. Itt három kategóriát átfogó teljes jogosultságú rendszerhozzáférést kapunk. A PackX logó mellett az “Admin” kiegészítő szöveg megjelenik.



“Admin” navigációs panel almenü

Felhasználók kezelése

A “Users” menüpontra kattintva megkapjuk az összes regisztrált felhasználó listáját, jogunk van törölni, illetve manuálisan hozzáadni felhasználókat.

PackX Admin					
Admin Package User About us Contact us Log out, Páll Károly					
All users					
				Add User Delete User	
ID	Name	Address	Phone	Email	Permission
1	Kovács Bea	Liliom utca 4.	+36201956673	k.bea@mail.com	Normal User
2	Szalma Géza	Egresy körút 58.	+36605385438	szalmag@mail.com	Normal User
3	Veres Péter	Malom út 12.	+36504098931	vrsptr@mail.com	Normal User
4	Páll Károly	Béla utca 69	+36201234578	pista@packx.hu	Admin
5	Kiss Lajos	Lajos utca 10	+36203030565	lajos@laj.com	Normal User
6	Admin Aladár	Fő utca 10	+36207708998	admin@packx.hu	Admin
7	Test	Béla utca 69	+36201234578	bela@packx-courier.hu	Courier
8	Széchenyi István	Egyetem tér 1.	+36208766767	artur.friedrich.harka@gmail.com	Normal User
9	Test	Béla utca 69	+36201234578	test1@test.hu	Normal User

“Admin All Users” oldal

PackXAdmin

Admin

Package

User

About us

Contact us

Log out, Péli Károly

Add user

Please enter the details

Name


Address

Phone


Email

Password

☐ Nem vagyok robot


Információk – Használati Szerződés

Add user



“Admin Add User” oldal

PackXAdmin

Admin

Package

User

About us

Contact us

Log out, Péli Károly


Delete accounts

Please note that this process cannot be undone.


User ID

☐ I have read that this process is not reversible and permanent.

☐ Nem vagyok robot


Információk – Használati Szerződés

Delete



“Admin Delete User” oldal

Automaták kezelése

A “Lockers” nézetre kattintva majdnem teljesen ugyan azt a felületet kapjuk, mint a futárok, annyiival több funkcionalitás érhető el az adminok számára, hogy tudnak csomag pontokat hozzáadni a rendszerhez. A “+” gombra kattintva tudunk hozzáadni helyszíneket.

PackXAdmin
Admin *
Package *
User *
About us
Contact us
Log out, Pál Károly

Lockers

+
Sort by Name
Sort by Fullness

Locker ID: 1
City: Győr
Locker Address: Szent István út 23.
Capacity: 4 / 7
37%
Show Packages
Map

Locker ID: 2
City: Győr
Locker Address: Kiss Ernő utca 5.
Capacity: 4 / 5
80%
Show Packages
Map

Locker ID: 3
City: Győr
Locker Address: Lomnic utca 30.
Capacity: 1 / 5
20%
Show Packages
Map

Locker ID: 4
City: Szombathely
Locker Address: Paragvári utca 74.
Capacity: 2 / 5
40%
Show Packages
Map

Locker ID: 5
City: Szombathely
Locker Address: Gömör utca 3.
Capacity: 1 / 5
20%

Locker ID: 6
City: Szombathely
Locker Address: Ehen Gyula tér 3.
Capacity: 7 / 10
70%

“Admin Lockers” oldal

PackXAdmin
Admin *
Package *
User *
About us
Contact us
Log out, Pál Károly

Add Locker

Please enter the details

City
Address
Capacity
0
Add Locker

“Admin Add Locker” oldal

Csomagok kezelése

A “Packages” panelre való kattintás után a rendszer listázza az adminisztrátornak a rendszerben lévő összes csomagot, valamint a hozzá tartozó információkat. Amennyiben egy csomagot valamiért törölni kell, egy gombnyomással gyorsan meg tudja tenni.

19

Admin Packages

ID	Sender Locker ID	Destination Locker ID	Receiver Name	Receiver Email	Size	Delivery Speed	Price	Delivery Date	Co2	Note	Courier ID	Delete
7	4	3	Lajos Lajos	artur.friedrich.harka@gmail.com	Large		2070	2023. 12. 06. 0:00:00	11.08	Lajos	0	Delete
10	1	5	Bélaaaa	szekekydani5g@gmail.com	Large		2570	2023. 12. 06. 0:00:00	21.34		0	Delete
11	2	6	Bélaaaa	szekekydani5g@gmail.com	Large		2570	2023. 12. 06. 0:00:00	20.98		0	Delete
13	4	7	Béla	szekekydani5g@gmail.com	Large		2162	2023. 07. 12. 0:00:00	17.21		0	Delete
14	1	6	Test János	szekekydani5g@gmail.com	Large		2370	2023. 12. 09. 0:00:00	20.91		0	Delete

“Admin Packages” oldal

Technikai leírás

Az alábbiakban részletezzük a fejlesztés során használt megoldásokat területtől függően.

Frontend

Alapinformációk, használt technológiák

A frontendben a React nevű JavaScript könyvtárat használtuk, ez adta az alapján a felületnek. Emelett a Bootstrap CSS könyvtárat használtuk a kinézet egységesítésének érdekében.

A fejlesztés során kiemelt figyelmet fordítottunk a könyvtárak által nyújtott tervezési sémák alkalmazására és sztandardizálására. A komponens alapú fejlesztés megkönnyítette és felgyorsította a folyamatokat, mivel már előre elkészített kódrészleteket tudunk felhasználni az alkalmazás bármely területén.

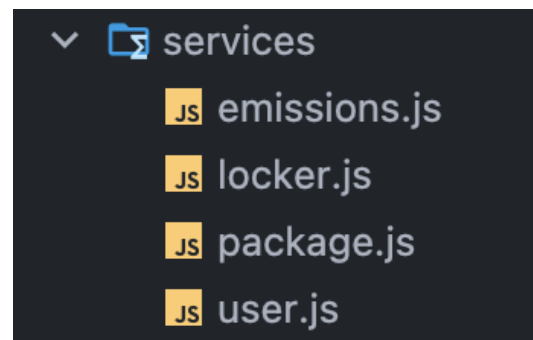
Az API

A backend által szolgáltatott Rest API műveleteket egy service alapú infrastruktúrával kezeltük, így a különböző funkciók elkülöníthetőek voltak és egymástól függetlenül könnyen lehetett őket használni az alkalmazás bármely részén.

A képen is láthatóak a jó elkülönült szolgáltatások: az emissziós, az automata, a csomag illetve a felhasználók is külön fájlokban kerültek megvalósításra.

A BaseURL egy külön fileban került definiálásra, hogy könnyebben elérhető, és változás esetén könnyen frissíthető legyen.

Az egyes szolgáltatások megvalósítját az összes szükséges eljárást amit a backend API kínál, többek között csomagok létrehozása, felhasználók regisztrálása vagy törlése, automaták hozzáadása.



```
1 usage  🧑 Artúr Friedrich
new(data, token) : Promise<AxiosResponse<...>> {
  return http.post( url: "/packages/new", data, config: {
    headers: {
      Authorization: `${token}`
    }
  });
}
```

Új csomag hozzáadása

Új csomag létrehozásánál természetesen szükségesek az új csomag adatai, valamint minden felhasználó rendelkezik egy egyedi tokennel, amivel ellenőrizzük, hogy valós és bejelentkezett felhasználó próbálja-e a csomagot létrehozni, amennyiben ez a feltétel nem teljesül, a csomag létrehozása sikertelen. Ha minden megfelelő akkor egy új csomag jön létre az adatbázisban a megadott adatokkal.

```
1 usage  🧑 Artúr Friedrich +1
const getPackageStatus = (id) : void => {
  PackageDataService.get(id) Promise<AxiosResponse<...>>
    .then((response : AxiosResponse<any> ) : void => setPackageData(response.data)) Promise<void>
    .catch((e) => console.error(e));
};
```

Csomag státusz lekérése

Egy adott service-t az alábbi módon lehet meghívni: az adott DataService tartalmazza a definiált metódusokat, amit API típustól függően vár egy data json objektumot vagy egy tokenet. Ezek után egy .then, .catch metódussal lehet az adatokat megkapni, vagy sikertelen hívás esetén az error ágat lefuttatni.

Az adatok valamint a felület kirajzolása a React JSX segítségével valósul meg, amivel az alapvetően HTML alapú környezetbe könnyen beilleszthetően JavaScript metódusok, amivel a reszponzív oldalak elkészítése egészen egyszerűvé válik.

Miután egy csomag státuszát a szerver sikeresen visszaadja, a packagesData nevű state-ben tároljuk, ami a React useState hook-jának köszönhetően egy nagyszerű lehetőséget kínál arra, hogy nagyobb mennyiségű adatot is könnyen tárolhassunk, módosíthassunk és elérhessünk.

```
2 usages  🧑 Artúr Friedrich +1 *
export const PackageStatus = () => {
  const [packageData, setPackageData] = useState( initialState: null);
```

useState segítségével egyszerűen létrehozhatók reaktív változók

Az API hívás populálja a változót, jelen esetben egy JSON objektummal, amit a JSX-en belül könnyen elérhetővé tehetünk.

```

<tr>
  <td className="package-table-title">Tracking number:</td>
  <td className="package-table-info">{packageData.Data.TrackID}</td>
</tr>
<tr>
  <td className="package-table-title">Size:</td>
  <td className="package-table-info">{packageData.Data.Size}</td>
</tr>

```

A csomag adatai a packageData változóból

A fejlesztés során sokszor használtuk a Bootstrap adta lehetőségeket, mivel meggyorsítják a fejlesztést, valamint egyedi CSS osztályok létrehozása sokszor időigényes és feleslegesen nehéz.

```

<ul className="dropdown-menu dropdown-menu-light">
  <li>
    <Link className="dropdown-item" to="/track">
      <p className="button-text mb-0">Track</p>
    </Link>
  </li>
  <li>
    <Link className="dropdown-item" to="/dispatch">
      <p className="button-text mb-0">Dispatch</p>
    </Link>
  </li>
  <li>
    <Link className="dropdown-item" to="/history">
      <p className="button-text mb-0">History</p>
    </Link>
  </li>
</ul>

```

A navigációs sáv egyik részlete

A navigációs sáv elkészítésénél, illetve az oldalak elrendezésénél volt a leghasznosabb a bootstrap könyvtár. Előre definiált oldalelrendezési lehetőségeket ad, amivel könnyen be lehet osztani egy oldalon a szövegek, képen, különböző tartalmi elemek helyzetét, valamint azok eltartását és méretét.

Sokszor azonban az egyedi stílusjegyek miatt szükség volt CSS osztályok létrehozására is.

```
.logout-button {  
  color: var(--white);  
  font-size: 16px;  
  font-style: normal;  
  font-weight: 700;  
  line-height: normal;  
  text-decoration: none;  
}
```

A kijelentkezés gomb egyedi stílusa

Összefoglalva az általunk választott technológiák remekül kiegészítették a backend nyújtotta lehetőségeket, valamint az agilis fejlesztési módszertan nagyszerűen kiegészíthető volt a modern és gyors fejlesztési tempót lehetővé tevő keretrendszerek használatával.

Backend

Alapinformációk, használt technológiák

A backend Golang nyelven íródott.

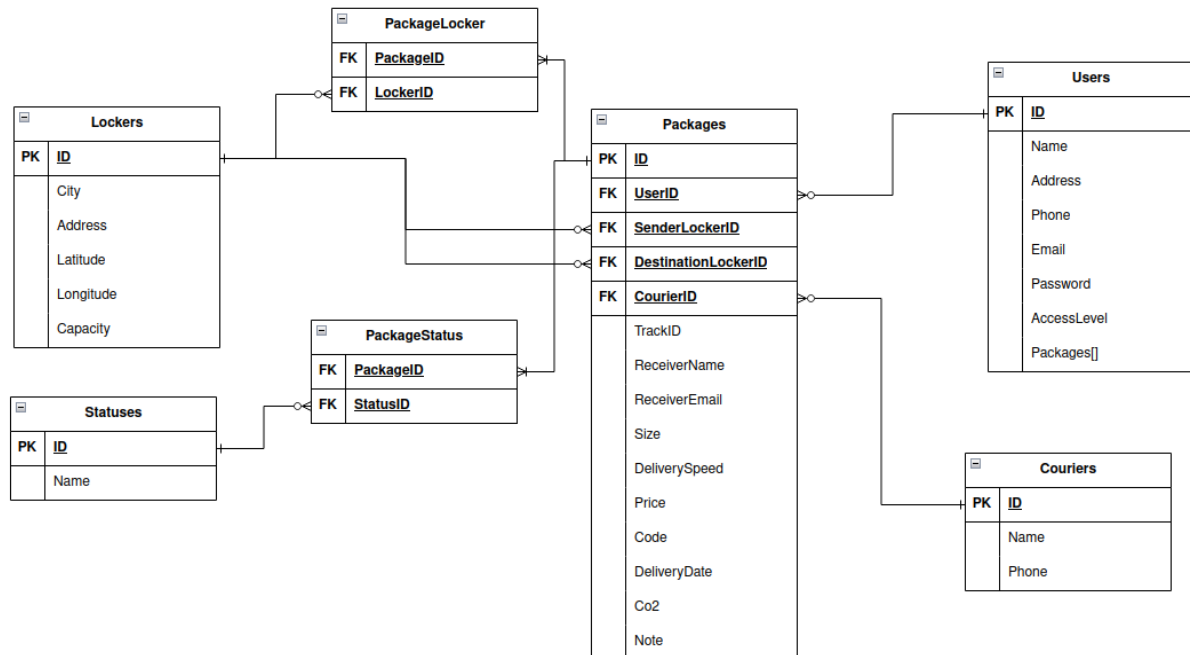
A választott adatbázis megoldásunk a PostgreSQL - nyílt forrású, relációs - adatbázisra esett.

Rest API-val valósítottuk meg az interface kialakítását a frontend felé, mellyel a relációs adatbázisban kezelt adatokat könnyen lehet kezelni.

A fejlesztési folyamat során kiemelt figyelmet fordítottunk a Go specifikus konvenciók betartására.

Adatbázis

Adatbázis modell:



A képen látható az adatbázis sematikus ábrája. A főbb táblák:

- Package
- User
- Courier
- Locker

A több-a-többhöz kapcsolatokat kapcsoló táblákkal oldottuk meg a csomag és automata, illetve a csomag és státusza között.

Ez utóbbit azért szerveztük ki a "statikus tartalmú" Package táblából, hogy amikor egy futár frissíti a csomagok státuszát, akkor a szervernek csak ebben a 2 mezőt tartalmazó táblában kelljen frissítenie a rekordokat. Ezzel téve hatékonyabbá a működést.

API

Az API kialakításához a gofiber package-t használtuk.

Ennek segítségével a megfelelő elérési utakon a megfelelő backend function-höz lehet irányítani a kérést és védeni lehet a végpontokat.

```
10 // Setting up endpoints + handle functions
11 func Routes(app *fiber.App) {
12     api := app.Group("/api")
13     packages := api.Group("/packages")
14
15     packages.Get("/:trackid", controllers.ListPackageByID) // /api/packages/get/{id} : Getting the {id}. package details
16     packages.Get("/code/:code", controllers.ListPackageCode) // /api/packages/code/{code} : Getting the package details by code
17     // From this point, all package endpoints are being authenticated
18     packages.Use(middleware.RequireJwtTokenAuth)
19
20     packages.Get("/all", controllers.ListPackages) // /api/packages/all : Listing all packages
21     packages.Post("/new", controllers.AddNewPackage) // /api/packages/new : Inserting new package via input json
22     packages.Delete("/:id", controllers.DeletePackageByID) // /api/packages/{id} : Delete package based on pathvariable 'id'
23     packages.Get("/getstatus/:id", controllers.GetPackageStatus) // /api/packages/getstatus/{id} : Getting the {id}. package status
24     packages.Post("/statusup/:id", controllers.ChangeStatusUp) // /api/packages/statusup/{id} : Increment {id}. package status
25     packages.Get("/courierpackages/:id", controllers.GetPackagesUnderCourier) // /api/packages/courierpackages/:id : Get packages under desired courier
26     packages.Post("/change-status", controllers.ChangeStatus) // /api/packages/change-status : Change a package status via input JSON (ID, NewStatusID)
27     packages.Post("/cancel/:id", controllers.MakeCanceled) // /api/packages/cancel/{id} : Make canceled a package based on pathvariable 'id'
28
29     users := api.Group("/users")
```

Authentication

Bizonyos végpontokat el lehet érni authentication nélkül.

Ugyanakkor bizonyos végpontokat csak a megfelelő jogosultsággal, vagy csak a bejelentkezett felhasználóknak szeretnénk megjeleníteni, elérhetővé tenni.

Ehhez mi a JWT token authentication megoldást használtuk.

A JWT token egy tömör megoldást biztosít a frontend-backend kommunikáción belül az autentikációra.

```

func RequireJwtTokenAuth(c *fiber.Ctx) error {
    tokenString := c.Get("Authorization")

    if tokenString == "" {
        return c.Status(fiber.StatusUnauthorized).JSON(fiber.Map{"message": "Unauthorized"})
    }

    token, err := jwt.Parse(tokenString, func(token *jwt.Token) (interface{}, error) {
        // Verify the token using the secret key
        claims := token.Claims.(jwt.MapClaims)

        // If expired
        exp := claims["exp"].(float64)
        if exp <= float64(time.Now().Unix()) {
            return nil, c.Status(fiber.StatusUnauthorized).JSON(fiber.Map{"message": "Unauthorized"})
        }

        // If user does not exist
        userId := claims["user_id"]
        var foundUser models.User
        initializers.DB.First(&foundUser, "id = ?", userId)
        if foundUser.ID == 0 {
            return nil, c.Status(fiber.StatusUnauthorized).JSON(fiber.Map{"message": "Unauthorized"})
        }

        return SecretKey, nil
    })

    if err != nil || !token.Valid {
        return c.Status(fiber.StatusUnauthorized).JSON(fiber.Map{"message": "Unauthorized"})
    }

    // Token is valid; proceed with the request
    return c.Next()
}

```

Csomagok kezelése

A csomagok létrehozásánál több dologra is kellett figyelni.

- Telítettsége alapján megfelelő automata választás küldő és fogadó oldalon
- TrackID generálás nyomonkövetéshez
- CO₂ kibocsátás számítása a csomag paraméterei alapján
- Email tájékoztatás a felek részére

```
if senderLocker.Capacity <= uint(nPackagesSenderLocker) {
    return c.Status(fiber.StatusBadRequest).JSON(fiber.Map{
        "Message": "Sender locker's capacity is full",
    })
}

if destinationLocker.Capacity <= uint(nPackagesDestinationLocker) {
    return c.Status(fiber.StatusBadRequest).JSON(fiber.Map{
        "Message": "Destination locker's capacity is full",
    })
}
```

```
// Generate a random 6 digit number for the package code
packageCode := utils.RandomPackageCode(6)
candidatePackage.Code = packageCode

// Generate TrackID
trackId := utils.RandomString(10)
candidatePackage.TrackID = trackId

// Calculate CO2 savings
candidatePackage.Co2 = utils.CalculateEmissionDifference(utils.CalculateDistance(senderLocker.Latitude, senderLocker.Longitude, destinationLocker.Latitude, destinationLocker.Longitude))
```

```
var body = fmt.Sprintf(BODY_ADD_PACKAGE, senderLocker.City+", "+senderLocker.Address, destinationLocker.City+", "+destinationLocker.Address, candidatePackage.TrackID, packageCode)
utils.SendEmail([]string(candidatePackage.ReceiverEmail, sender.Email), SUBJECT_ADD_PACKAGE, body)
```

Email küldés

Az email küldés-t a golang-v2-val valósítottuk meg.

```
func SendEmail(recipientEmails []string, subject string, body string) error {
    SenderEmailAddress := os.Getenv("SENDER_EMAIL_ADDRESS")
    SenderEmailPassword := os.Getenv("SENDER_EMAIL_PASSWORD")
    SMTP_PORT, _ := strconv.Atoi(os.Getenv("SMTP_PORT"))
    EMAIL_HOST := os.Getenv("EMAIL_HOST")

    for _, recipientEmailAddress := range recipientEmails {
        message := gomail.NewMessage()
        message.SetHeader("From", SenderEmailAddress)
        message.SetHeader("To", recipientEmailAddress)
        message.SetHeader("Subject", subject)
        message.SetBody("text/html", body)

        dialer := gomail.NewDialer(EMAIL_HOST, SMTP_PORT, SenderEmailAddress, SenderEmailPassword)

        if err := dialer.DialAndSend(message); err != nil {
            fmt.Println("Error when sending the email to: " + recipientEmailAddress + "\nError: ")
            fmt.Println(err)
        }

        fmt.Println("Email has been sent to: " + recipientEmailAddress + "\nfrom: " + SenderEmailAddress)
    }

    return nil
}
```

CO₂ kibocsátás számolás

Az kibocsátás számolásnál egy átlagos benzines, és átlagos elektromos autó kibocsátásával számoljuk ki a különbséget, a 2 automata közötti távolságot pedig haversine formulával számoljuk ki (nem akartunk geo-api-okat használni).

```

func CalculateEmissionDifference(distance float64) float64 {
    electricCarEnergyConsumption := 0.2 // kWh per mile
    electricCarEmissions := 0.3 // kg CO2 per kWh

    petrolCarFuelConsumption := 30 // miles per gallon (mpg)
    petrolCarEmissions := 8.8 // kg CO2 per gallon of gasoline

    electricCarEnergy := electricCarEnergyConsumption * distance
    electricCarTotalEmissions := electricCarEnergy * electricCarEmissions

    petrolCarFuel := distance / float64(petrolCarFuelConsumption)
    petrolCarTotalEmissions := petrolCarFuel * petrolCarEmissions

    emissionSavings := petrolCarTotalEmissions - electricCarTotalEmissions

    return emissionSavings
}

func CalculateDistance(latitude1, longitude1, latitude2, longitude2 float64) float64 {
    // Convert latitude and longitude from degrees to radians
    latitude1 = latitude1 * math.Pi / 180
    longitude1 = longitude1 * math.Pi / 180
    latitude2 = latitude2 * math.Pi / 180
    longitude2 = longitude2 * math.Pi / 180

    // Haversine formula
    dlat := latitude2 - latitude1
    dlon := longitude2 - longitude1
    squaredChordLength := math.Sin(dlat/2)*math.Sin(dlat/2) + math.Cos(latitude1)*math.Cos(latitude2)*math.Sin(dlon/2)*math.Sin(dlon/2)

    centralAngle := 2 * math.Atan2(math.Sqrt(squaredChordLength), math.Sqrt(1-squaredChordLength))

    // Calculate the distance
    distance := earthRadius * centralAngle
    return distance
}

```

DevOps

Automatizált infrastruktúra

Terraform

A Terraform a Hashicorp terméke, ami lehetővé teszi, hogy az alkalmazásainkat futtató környezetet és paramétereit kódként definiáljuk.

A különböző nagy felhőszolgáltatók (pl. AWS, Azure, GCP vagy a DigitalOcean, Linode) saját providereit (modulok, melyek az adott szolgáltató REST API-ján keresztül hajtanak végre műveleteket) felhasználva kész futtatási környezetet kapunk percek alatt. Így egy verziókövetett és könnyen újrafelhasználható infrastruktúrát kapunk, ami jelentősen csökkenti az új projektek elindításához szükséges időt.

A projektünk egyik főpillére, hiszen mind az infrastruktúráért (virtuális gépek), mind pedig az Ansible playbook meghívásért a terraform/main.tf fájlban található erőforrás definíciók felelnek.

Ezt a hivatalos DigitalOcean, Cloudflare és Ansible providerek segítségével éri, melyek előkészítése a terraform/providers.tf állományban történik. Az elérhető bemeneti változók a terraform/inputs.tf, míg a futás során kigyűjtött, később fontos értékek listája a terraform/outputs.tf-ben kerülnek listázásra. A gitben nem

tárolt, `terraform/terraform.tfvars` tartalmazza a bemeneti változók tényleges értékeit.

A dropletek létrehozásakor megadjuk a kívánt image azonosítóját, a gép régióját, nevét, és a `cloud-init`-hez használt állomány elérési útját. A `cloud-config.yml` alapján felkészíti a gépet az Ansible-lel való konfigurációra (pl. `mgmt` user létrehozása, `root` SSH elérésnek letiltása). A VM létrehozása után becsatornázza az eszköz elérhetőségét `terraform.tfstate` fájlból egy dinamikus inventoryba, amit majd az Ansible playbook fog felhasználni a konfigurációra.

Ansible

Az Ansible, mint konfiguráció menedzsmentet elősegítő eszköz segítségével, akár egy távoli eszköztől is (control node) is képesek vagyunk konfigurálni az eszközeinket, idempotens módon. A különböző lépések végrehajtása úgynevezett playbookok futtatásával történnek, amik szabványos YAML leírófájlok formájában vannak definiálva.

A programcsomag Python-alapú, és nem szükséges a céleszközökre (Linux, Windows vagy akár nagyvállalati hálózati eszközök) agentet (valamilyen egyedi protokollt értelmezni képes kliensprogram) telepíteni, mindent az SSH protokollon keresztül intéz, a rendszernek megfelelő parancsok kiadásának segítségével.

A létrehozás utáni tényleges konfigurációt az `ansible/main.yml` fájlban definiált playbook látja el. Ez az állomány vonultatja fel, hogy hol és milyen feladatokat szeretnénk végrehajtani. A céleszköz csatlakozásai adatait egy dinamikus forrásból, a Terraform által, futásidőben generált adatokból kapja meg (`ansible/inventory/hosts.yml`), míg a főbb változókat az `ansible/inventory/group_vars` alatti mappa csoportokra szedve tartalmazza.

A playbook moduláris egységeket, ún. role-okat hív meg, amik a különböző, jól elkülönülő feladatok ellátására készítettem. Ezek az alegységek saját változókkal és feladatdefiníciókkal rendelkeznek.

CI/CD

Célunk volt, hogy elkerüljük a manuális módon való fordítást és telepítést, ezért döntöttük egy robusztus, könnyen újrafelhasználható CI/CD pipeline kialakítása mellett. Ehhez platformnak a GitHub Actions-t választottuk, hisz így nem volt szükség egy külön szolgáltatást igénybe vennünk és a mindannyiunk számára elérhető tanulói csomag rengeteg ingyenes futási időt biztosít havonta.

Backend

Egy Go alkalmazás esetén elkerülhetetlen, hogy binárisá alakítsuk az alkalmazásunkat ahhoz, hogy éles környezetben futtassuk. Ehhez kiváló segítséget

nyújtott a goreleaser nevű szoftver, mely egy YAML fájlban tárolt definíció alapján képes egyszerre több platformra fordítani. Emellett hasznos funkciója, hogy képes előkészíteni megannyi különbözős disztribúciós csatornára való feltöltéshez a programunkat.

A pipeline a main branchre történő push eseménynél a commit üzenet alapján (feature, fix vagy egyéb), hogy milyen módon inkrementálja a szemantikus verziót (pl. egy új feature esetén: v1.2.3 -> v1.3.3). A következő fázisban először előkészíti a függőségeket (Docker és virtualizációs segédprogramok, goreleaser), majd futtatja a repóban tárolt goreleaser.yml állomány megadásával a segédprogramot.

Sikeres build és release esetén létrehoz egy új release-t a GitHub-on és feltölti az új konténer képfájlt a GitHub Container Registrybe. A képfájl ún. multistage image, azaz az első stádiumban maga a fordítás történik, míg a második, végső stádiumban már csak a kész bináris kerül átmásolásra, ami jelentősen csökkenti az image méretét. Biztonsági szempontból is előnyösebb - a konténerben már csak az adott binárist lehetséges futni, amivel elkerülhetjük a jogosultságok eskalációjából származó sérülékenységeket.

Frontend

A webes felület esetén nem értelmezhető a tradicionális, binárisra való fordítás, így kisebb mértékben eltér a backendnél kialakított workflow-tól.

A szemantikus verziózásért a semantic-release nevű, méltán népszerű package felel, ami egy releaserc állományból tölti be a konfigurációját a futásidő közben - milyen pluginokat szeretnénk használni, milyen elérési utakat szeretnénk figyelmen kívül hagyni stb.

Az image létrehozásához és feltöltéséhez a hivatalos Docker GitHub Actionöket hívjuk segítségül. A feltöltött képfájlok ez esetben is több stádiumú fordítással készülnek, de itt az NGINX webszerver szolgálja ki a kéréseket, az optimalizált, statikus HTML oldal megjelenítésével.

Folyamatos kiadás (CD, continuous deployment)

Környezet és enklávé (backend, frontend, monitoring) páronként 1-1 repóban tároljuk az adott stacket leíró Docker Compose állományokat, a példa környezeti változókkal együtt.

A célgépeken ezek kerülnek elhelyezésre és indításra (pl. infra-backend-sandbox repo -> packx-backend-sandbox gép). Induláskor letölti a fájlban található image-eket, majd létrehozza a kért konténereket.

A képfájlok (így az alkalmazásoké is) verziója a YAML-fájlban került specifikálásra, így egy központi helyről tudjuk igény szerint frissíteni a futó példányokat. A hostok 5 percenként kísérlik meg a leírófájl frissítését, mely egy cron job jóvoltából történik.

Monitoring

Exporterek

Az exporterprogramok szolgálnak arra, hogy kinyerjék egy adott rendszer működését leíró adatok és metrikákat. Ezeket az adatok egy TSDB-típusú (time series database, idősor adatbázis) adatbázisba tudjuk írni, amely lehetővé teszi a historikus adatelemzést.

Esetünkben kettő különböző exportert használunk a környezetenként 2 node-on:

- Node Exporter, mely a VM erőforrásaival (CPU, RAM és disk) kapcsolatos mérőszámokat gyűjti be
- cAdvisor, mely a Docker konténerek viselkedését reprezentáló adatok biztosítja számunkra

VMAgent és VictoriaMetrics

Az exporterek által összegyűjtött adathalmazokat be is kell gyűjtenünk, erre szolgál a VMAgent. A VMAgent egy Prometheus-szal kompatibilis konfigurációs állomány alapján begyűjti a megadott exporterek elérési útján, megadott időközönként az elérhető metrikákat.

A metrikákat a későbbi feldolgozására és tárolására a VictoriaMetrics-et választottuk, amely a Prometheus mellett a vezető TSDB a piacon. Képes HA (high availability, magas elérhetőségű) architektúra keretei között is üzemelni, mely megkönnyíti az igény szerinti skálázást. A VictoriaMetrics fog adatforrásként szolgálni a Grafana számára.

Grafana

A Grafana különböző adatforrásokat felhasználva képes adatvizualizációkat készíteni, emellett pedig képes előre definiált szabályok alapján értesítéseket küldeni.

A projektünk keretében környezetenként üzemeltetünk egy 1-1 Grafana instance-t.

