

Documentation about BDS assignment

3

Not the best attempt at creating desktop application, but not terrible.

I really tried.

This is a documentation about my dumpster fire of a program. Get ready to be properly disappointed. I hope your expectations are lower than my will to live, and trust me, it's really low after this assignment. Anyway, enough of my crying, let's get into my program.

Brief description

This program contains basic graphical user interface. First thing you will encounter log-in window. This is not a fake login or anything. It will ask you for email and password. You can only use email and password that is in database. If you use incorrect email or email of someone who doesn't have password, authentication will fail. Passwords are encrypted with PGCrypto and verified. After this you will enter a basic view of our database. Some columns of a selected table. Everything will be properly explained later. In this basic view, you can select detailed info about any person in the table. You can edit and delete any person you choose. There is a search button, where you can filter people by their name. Another thing is create button that adds person to the database. After this you need to refresh the database with the refresh button. Last but not least, I created an SQL Injection button. That takes you to an environment where you can practice SQL injections on selected table in safe environment.

Running the program

You need to go to root folder and run command **\$ mvn clean install**

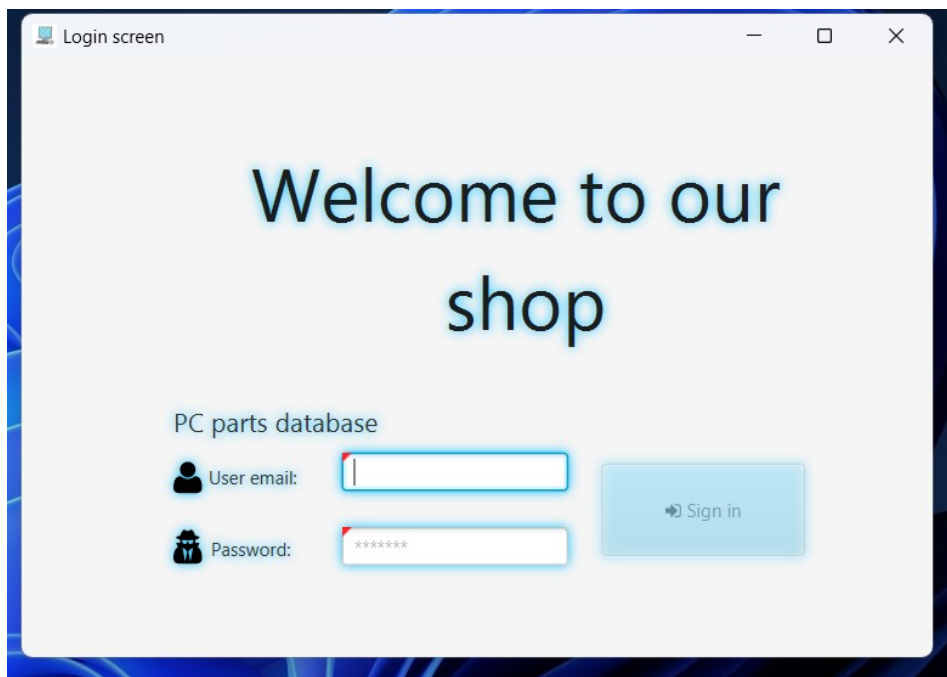
```
Terminal: Local x + v
PS C:\Users\rosik\Desktop\School\Projects\bds-project-assignment-3> mvn clean install
```

and then run command **\$ java -jar .\target\pc-shop-1.0.0.jar**

```
Terminal: Local x + v
[INFO] Installing C:\Users\rosik\Desktop\School\Projects\bds-project-assignment-3\target\pc-shop-1.0.0.jar to C:\Users\rosik\.m2\repository\org\but\pc-shop\1.0.0\pc-shop-1.0.0.jar
[INFO] Installing C:\Users\rosik\Desktop\School\Projects\bds-project-assignment-3\dependency-reduced-pom.xml to C:\Users\rosik\.m2\repository\org\but\pc-shop\1.0.0\pc-shop-1.0.0.pom
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 5.856 s
[INFO] Finished at: 2022-12-29T02:21:11+01:00
[INFO] -----
PS C:\Users\rosik\Desktop\School\Projects\bds-project-assignment-3> java -jar .\target\pc-shop-1.0.0.jar
```

Login

After doing that, you will be encountered with this window. This is my login screen.

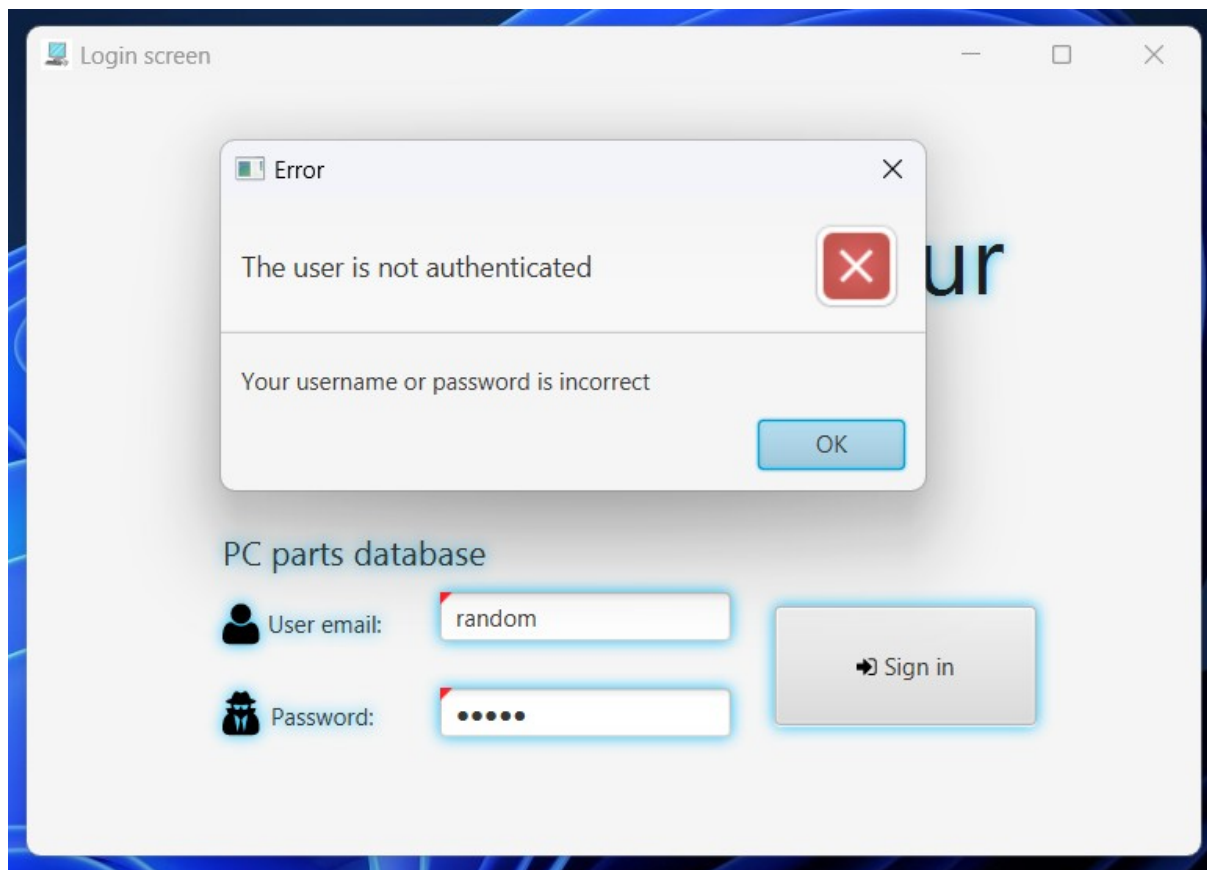


You will need to enter valid email address and password. This is the person table in our database that has passwords.

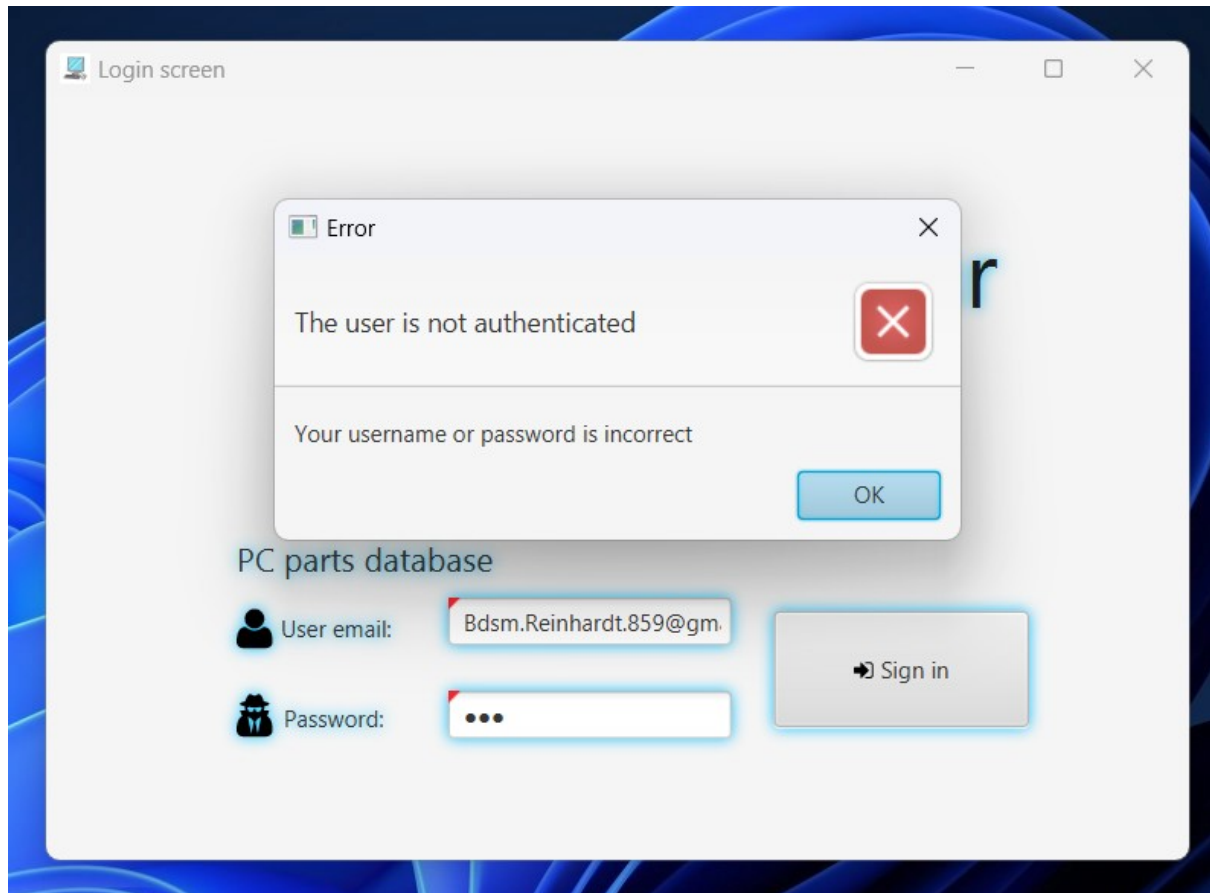
	id_person [PK] integer	name character varying (45)	surname character varying (45)	email character varying (45)	bio character varying (45)	profile_picture character varying (45)	is_student boolean	is_vip boolean	phone_number integer	password text
1	1	Marian	Flowers	Flowers.Marian.507@gmail.com	[null]	[null]	true	true	853970000	\xaGymU4XC9Xc
2	2	David	Okamura	Okamura.David.516@gmail.com	[null]	[null]	false	false	547451740	\xaGymU4XC9Xc
3	3	Raiden	Legendary	Legendary.Raiden.226@gmail.com	[null]	[null]	true	true	219787477	\xaGymU4XC9Xc
4	4	Reinhardt	Bdsm	Bdsm.Reinhardt.859@gmail.com	[null]	[null]	false	false	536292511	[null]
5	5	Adolf	Okamura	Okamura.Adolf.847@gmail.com	[null]	[null]	false	false	314934316	[null]
6	6	David	Chink	Chink.David.262@gmail.com	[null]	[null]	true	true	663569699	[null]
7	7	Bibiana	Chungus	Chungus.Bibiana.118@gmail.com	[null]	[null]	false	false	976717031	[null]
8	8	Haruka	Soyak	Soyak.Haruka.248@gmail.com	[null]	[null]	false	false	385164987	[null]
9	9	Raider	Tranny	Tranny.Raider.618@gmail.com	[null]	[null]	true	true	262730231	[null]
10	10	Marian	Kockoholka	Kockoholka.Marian.652@gmail.com	[null]	[null]	false	false	960012776	[null]
11	11	Haruka	Okamura	Okamura.Haruka.672@gmail.com	[null]	[null]	true	true	752356349	[null]
12	12	Raiden	Kotleba	Kotleba.Raiden.827@gmail.com	[null]	[null]	true	true	620514404	[null]
13	13	Raiden	Leoendary	Leoendary.Raiden.617@gmail.com	[null]	[null]	false	false	752874414	[null]

You can see that only first three people have password. That means you can only use their emails. If you use any email after ID 3, you will not be able to log in. If you use anything else than provided emails, you will also not be logged in. You can see it on the demonstration.

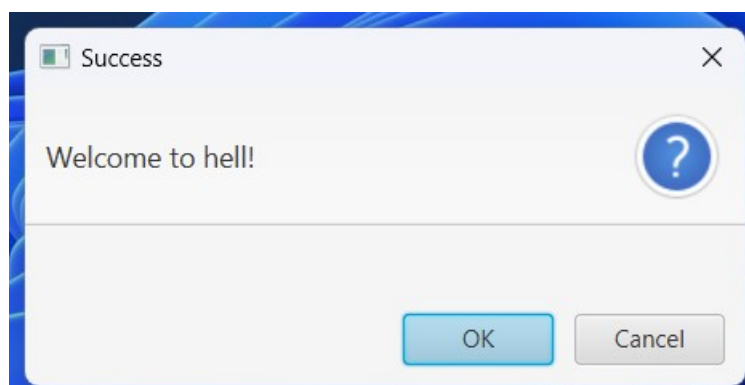
Invalid username and password.



We will use *Bdsm.Reinhardt.859@gmail.com* . (haha what a funny name, get it ?). It does not have assigned password in database.

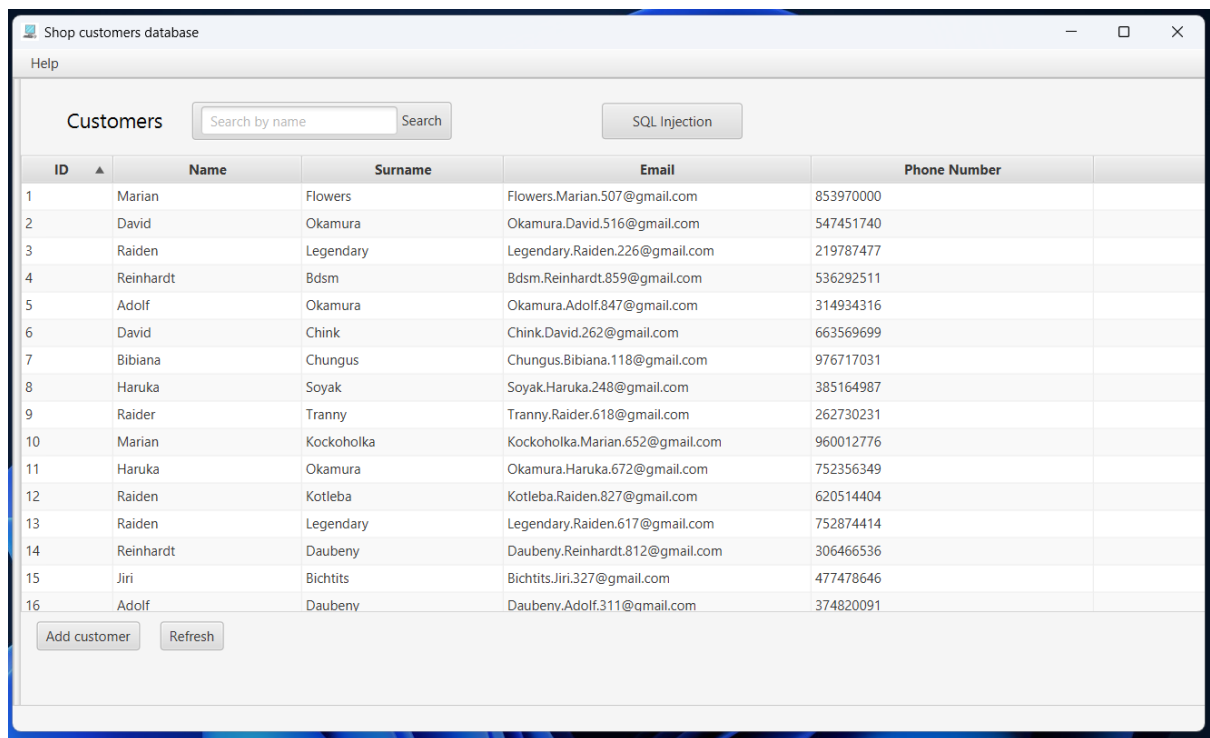


This is how successful login looks like. Email *Flowers.Marian.507@gmail.com* and password is *user*



Application

After succesful login, you will see window like this.



ID	Name	Surname	Email	Phone Number
1	Marian	Flowers	Flowers.Marian.507@gmail.com	853970000
2	David	Okamura	Okamura.David.516@gmail.com	547451740
3	Raiden	Legendary	Legendary.Raiden.226@gmail.com	219787477
4	Reinhardt	Bdsm	Bdsm.Reinhardt.859@gmail.com	536292511
5	Adolf	Okamura	Okamura.Adolf.847@gmail.com	314934316
6	David	Chink	Chink.David.262@gmail.com	663569699
7	Bibiana	Chungus	Chungus.Bibiana.118@gmail.com	976717031
8	Haruka	Soyak	Soyak.Haruka.248@gmail.com	385164987
9	Raider	Tranny	Tranny.Raider.618@gmail.com	262730231
10	Marian	Kockoholka	Kockoholka.Marian.652@gmail.com	960012776
11	Haruka	Okamura	Okamura.Haruka.672@gmail.com	752356349
12	Raiden	Kotleba	Kotleba.Raiden.827@gmail.com	620514404
13	Raiden	Legendary	Legendary.Raiden.617@gmail.com	752874414
14	Reinhardt	Daubeny	Daubeny.Reinhardt.812@gmail.com	306466536
15	Jiri	Bichtits	Bichtits.Jiri.327@gmail.com	477478646
16	Adolf	Daubeny	Daubeny.Adolf.311@gmail.com	374820091

First off, we will start with create button in the bottom right corner. After you press it, you will see something like this.

Help

Customer

ID	
1	Maria
2	David
3	Raide
4	Reinh
5	Adolf
6	David
7	Bibian
8	Haruk
9	Raide
10	Maria
11	Haruk
12	Raide
13	Raide
14	Reinh
15	Jiri
16	Adolf


Add customer

Name

Surname

Email

Phone number



Add person

Add customer

After you fill in the table, it will look like this.


Add customer

Name

Surname

Email

Phone number



Add person

As you can see, we successfully added Andrew Tate.

Shop customers database

Help

Customers Search

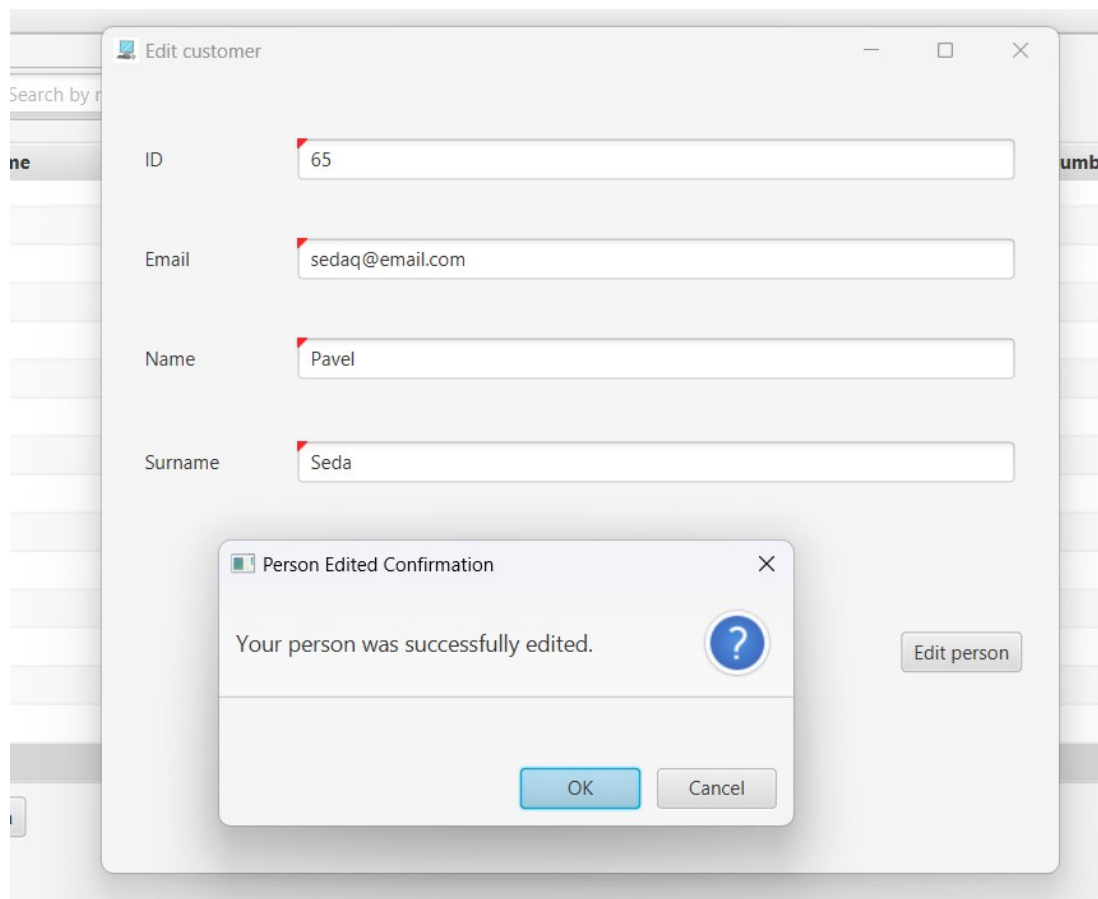
ID ▲	Name	Surname	Email	Phone Num
41	Reinhardt	Kockoholka	Kockoholka.Reinhardt.562@gmail.com	537739606
42	Bibiana	Marchand	Marchand.Bibiana.856@gmail.com	558089652
43	Janko	Soyak	Soyak.Janko.772@gmail.com	775368125
44	Ramona	Chad	Chad.Ramona.551@gmail.com	679471907
45	Raider	Cejka	Cejka.Raider.113@gmail.com	627787286
46	Dominik	Cejka	Cejka.Dominik.206@gmail.com	992749862
47	David	Chungus	Chungus.David.445@gmail.com	746721870
48	Jiri	Cejka	Cejka.Jiri.360@gmail.com	129406578
49	Ramona	Laska	Laska.Ramona.478@gmail.com	613144721
50	Laszlo	Sus	Sus.Laszlo.479@gmail.com	799964513
51	Raiden	Flowers	Flowers.Raiden.684@gmail.com	438842562
52	Laszlo	Bichtits	Bichtits.Laszlo.664@gmail.com	289591830
53	Laszlo	Laska	Laska.Laszlo.183@gmail.com	442817274
54	Marian	Chink	Chink.Marian.617@gmail.com	500092316
55	Marian	Kockoholka	Kockoholka.Marian.736@gmail.com	270401363
65	Andrew	Tate	topg@gmail.com	123456

We can now edit Andrew to someone more pleasant. Simply right click on him and pick option Edit.

If Shop customers database press
 Ec Help

Customers Search

ID ▲	Name	Surname	Email	Phone Num
41	Reinhardt	Kockoholka	Kockoholka.Reinhardt.562@gmail.com	537739606
42	Bibiana	Marchand	Marchand.Bibiana.856@gmail.com	558089652
43	Janko	Soyak	Soyak.Janko.772@gmail.com	775368125
44	Ramona	Chad	Chad.Ramona.551@gmail.com	679471907
45	Raider	Cejka	Cejka.Raider.113@gmail.com	627787286
46	Dominik	Cejka	Cejka.Dominik.206@gmail.com	992749862
47	David	Chungus	Chungus.David.445@gmail.com	746721870
48	Jiri	Cejka	Cejka.Jiri.360@gmail.com	129406578
49	Ramona	Laska	Laska.Ramona.478@gmail.com	613144721
50	Laszlo	Sus	Sus.Laszlo.479@gmail.com	799964513
51	Raiden	Flowers	Flowers.Raiden.684@gmail.com	438842562
52	Laszlo	Bichtits	Bichtits.Laszlo.664@gmail.com	289591830
53	Laszlo	Laska	Laska.Laszlo.183@gmail.com	442817274
54	Marian	Chink	Chink.Marian.617@gmail.com	500092316
55	Marian	Kockoholka	Kockoholka.Marian.736@gmail.com	270401363
65	Andrew		topg@gmail.com	123456



As you can see, you are now entity with ID number 65.

Shop customers database

Help

Customers Search

ID ▲	Name	Surname	Email	Phone
41	Reinhardt	Kockoholka	Kockoholka.Reinhardt.562@gmail.com	537739606
42	Bibiana	Marchand	Marchand.Bibiana.856@gmail.com	558089652
43	Janko	Soyak	Soyak.Janko.772@gmail.com	775368125
44	Ramona	Chad	Chad.Ramona.551@gmail.com	679471907
45	Raider	Cejka	Cejka.Raider.113@gmail.com	627787286
46	Dominik	Cejka	Cejka.Dominik.206@gmail.com	992749862
47	David	Chungus	Chungus.David.445@gmail.com	746721870
48	Jiri	Cejka	Cejka.Jiri.360@gmail.com	129406578
49	Ramona	Laska	Laska.Ramona.478@gmail.com	613144721
50	Laszlo	Sus	Sus.Laszlo.479@gmail.com	799964513
51	Raiden	Flowers	Flowers.Raiden.684@gmail.com	438842562
52	Laszlo	Bichtits	Bichtits.Laszlo.664@gmail.com	289591830
53	Laszlo	Laska	Laska.Laszlo.183@gmail.com	442817274
54	Marian	Chink	Chink.Marian.617@gmail.com	500092316
55	Marian	Kockoholka	Kockoholka.Marian.736@gmail.com	270401363
65	Pavel	Seda	sedaq@email.cz	123456

Next we can show address details about selected person. Just right click like before and select address details.

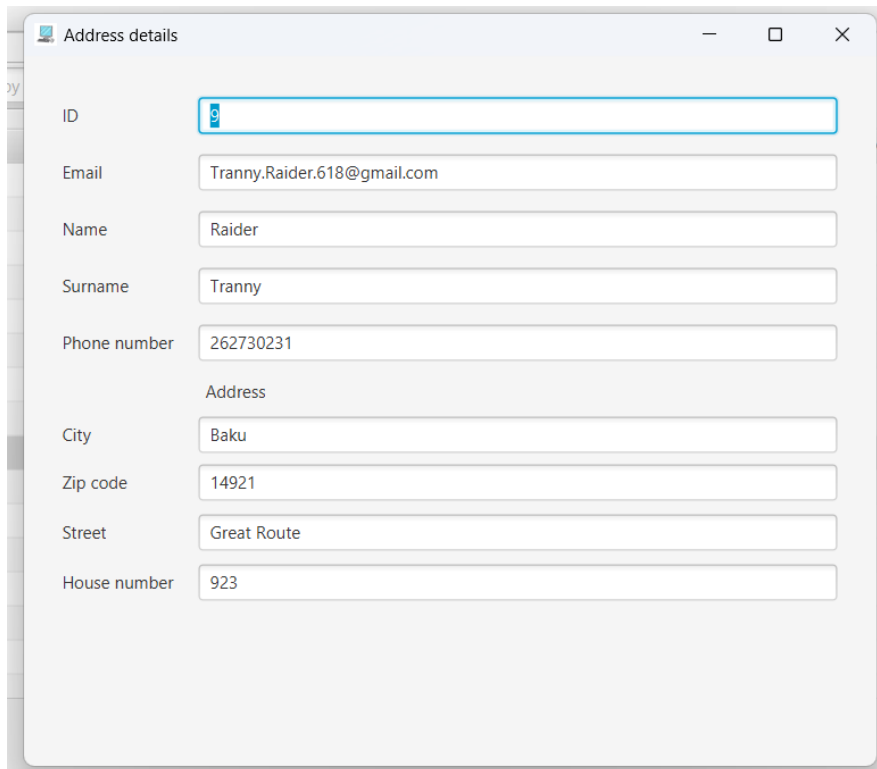
8	Haruka	Soyak
9	Raider	Tranny
10	Marian	Kockoholka
11	Haruka	Okamura
12	Raiden	Kotleba
13	Raiden	Legendary
14	Reinhardt	Reinhardt

Edit customer

Show address details

Delete customer

Then it will look like this. Address details is simply SELECT from persons and LEFT JOIN with address table.



The screenshot shows a web application window titled "Address details". It contains several input fields for user information:

- ID: 8
- Email: Tranny.Raider.618@gmail.com
- Name: Raider
- Surname: Tranny
- Phone number: 262730231
- Address section:
 - City: Baku
 - Zip code: 14921
 - Street: Great Route
 - House number: 923

We can also delete person, so I will delete you (sorry, please don't delete my points).

55	Marian	Kockoholka
65	Pavel	Seda

Add customer

Edit customer

Show address details

Delete customer

As you can see, your entry is not there after refresh.

51	Kaiden	Flowers	Flowers.Kaiden.684@gmail.com	4388425
52	Laszlo	Bichtits	Bichtits.Laszlo.664@gmail.com	2895918
53	Laszlo	Laska	Laska.Laszlo.183@gmail.com	4428172
54	Marian	Chink	Chink.Marian.617@gmail.com	5000923
55	Marian	Kockoholka	Kockoholka.Marian.736@gmail.com	2704013

Name	Surname
Flowers	
Okamura	
Legendary	
Bdsm	
Okamura	
Chink	
Chungus	
Soyak	
Tranny	
Kockoholka	
Okamura	
Kotleba	
Legendary	
Darkony	

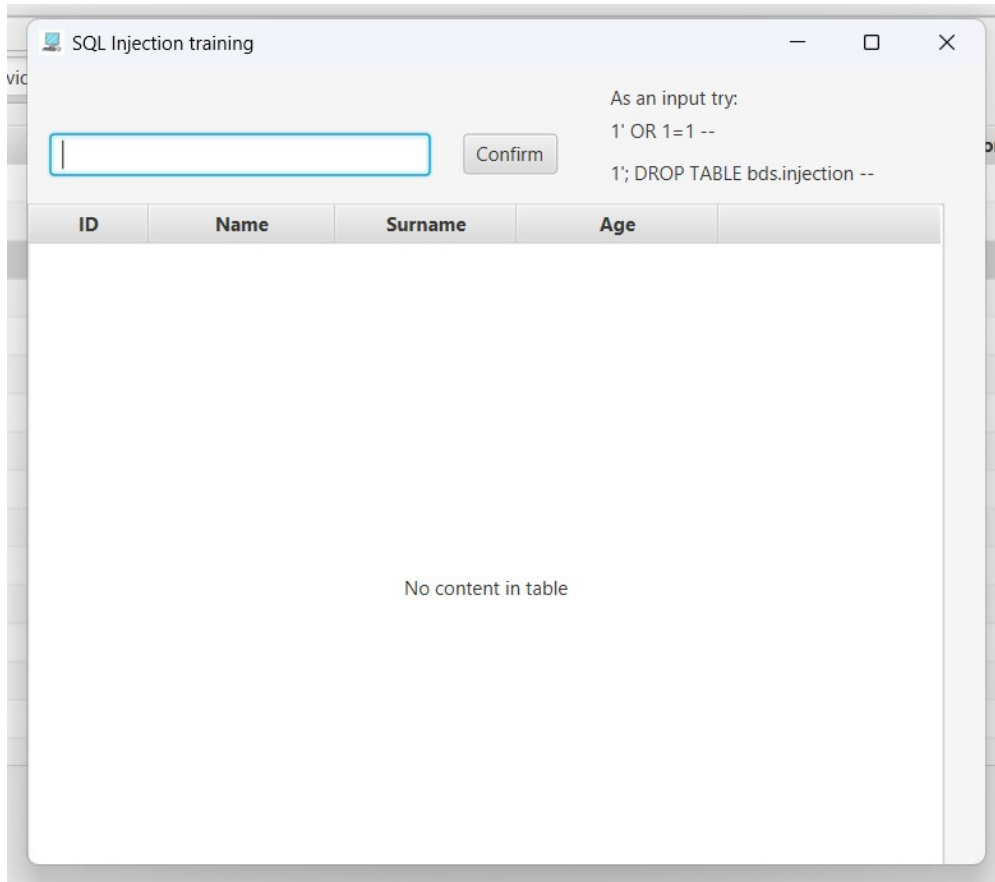
ID	Name	Surname	Email
2	David	Okamura	Okamura.David.516@...
6	David	Chink	Chink.David.262@gm...
21	David	Flowers	Flowers.David.246@g...
37	David	Revival	Revival.David.567@g...
47	David	Chungus	Chungus.David.445@...

Another thing you can you is person filter. This simple search bar filters people by their name. You don't have to write the full name, only couple of letters and it will filter all people who have them in their name. Here I filtered people by name David.

Lastly, we will focus on this button here. It will take us to SQL Injection environment where we can test different types of injections. I didn't use prepared statements here, so it is very exposed and liable to attacks in form of SQL injection. We will demonstrate later.

SQL Injection		
	Email	Ph
	Flowers.Marian.507@gmail.com	853970000

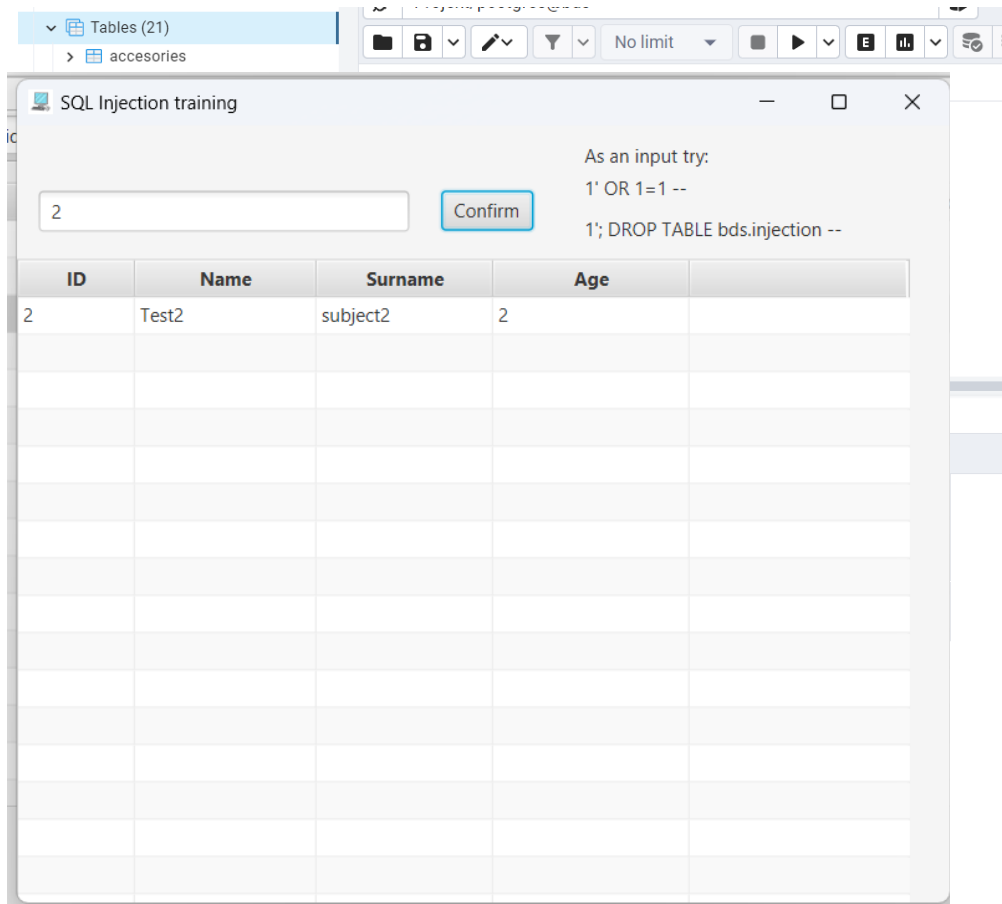
After pressing that button, window like this pops up.



The screenshot shows a window titled "SQL Injection training" with standard window controls (minimize, maximize, close). Inside the window, there is an input field with a blue border and a "Confirm" button to its right. To the right of the input field, there is text that says "As an input try:" followed by two examples of SQL injection payloads: "1' OR 1=1 --" and "1'; DROP TABLE bds.injection --". Below this input area is a table with four columns: "ID", "Name", "Surname", and "Age". The table is currently empty, and the text "No content in table" is displayed in the center of the table area.

ID	Name	Surname	Age
No content in table			

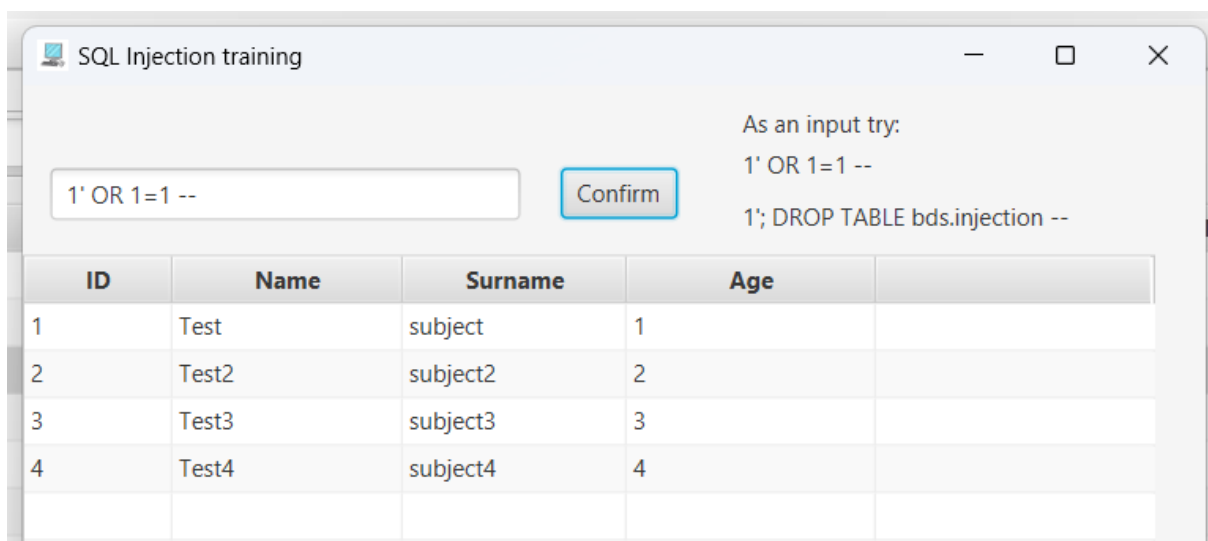
This view is linked with simple SELECT by ID from dummy table I created. There are no prepared statements, so it should be easy to exploit this table. Just for information, it looks like this .



We will try simple SELECT by ID. As you can see it works normally.

Now let's try to exploit this table (I feel like writing a TryHackMe room description). We can for example enter something like this: **1' OR 1=1** –

What this does is that after the first 1 we have ' , that means it ends the select there. After that we have 1=1 which always equals true. – after that means that everything else that might be written in that query afterwards is taken as a comment and will not execute. With this, we can retrieve more data than we are supposed to. Let's see.



Here comes the fun part. If you want to do more damage to such vulnerable database, you can try dropping the table itself. It's pretty cool, not so much for the person responsible for the security of the database.

After the 1 we have ' again, that ends our input, ; after that means end of the query itself. Next we our DROP TABLE query which will execute. – makes sure that nothing else after our query executes because it is perceived as a comment .

[illegible]

But if we look into our database again, we can see that there are only 20 tables, with injection missing.

	Sequences
▼	Tables (20)
>	accessories
>	address
>	cart_info
>	feedback
>	login
>	notebook_type
>	payment
>	pc_config
>	pc_parts
>	...

Summary

I had a love-hate relationship with this project. When everything worked perfectly it was awesome, but when things were going downhill it was pain and suffering.