Brno University of Technology

Faculty of Electrical Engineering & Communication

Flight System Database - Assignment 3

Author
Petr Bráblík

December 31, 2022

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Contents

1 Abstract

In this project I implemented database design into a working application. Through the application, which has a log-in feature, you can view all passengers with their emails and some other personal information. I also implemented CRUD operations on the flight entity, which means I can create, read, update and delete flights in my database.

2 Introduction

This paper will try to explain how this program works, the design decisions behind it a other things that come to mind. I drew from my first assignment, which was the database design of a flight system, which could store data such as information about passengers, information about various flights and information about pilots and airplanes. I also drew from my second assignment, which focused on working with the database through complex queries.

The program will simulate a flight system database where you can browse peoples account, with their email addresses. The next functionality is to work with flight, you can create, read, update and delete them. Another functionality is a log-in window, where you can log-in with email address and the right password.

Passwords are stored in the database in hash form, the application then hashes the password, which the user entered. The two hashes are then compared and if they are not the same, then a pop-up window will appear, displaying that the user has entered the wrong password.

The program is written in Java programming language, with the help of Apache Maven. The database is local-hosted with PostgreSQL and is accessed through PgAdmin. In the second assignment I created a no-superuser role from which I accessed the database. Its best not to use the superuser role, because if the role gets compromised, the hacker can't destroy the database in the process.

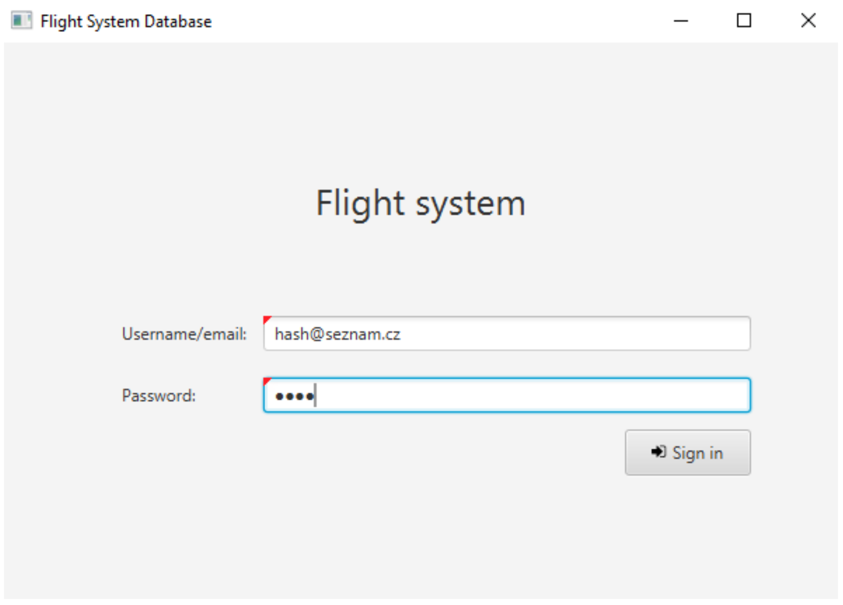
I also created a window where I can simulate a SQL-injection attack.

3 User Interface

Now I will describe each window in my program.

3.1 Log-in window

This is the log-in window. It has two text fields for log-in purposes (email and password) and a sign-in button. The password which the user enters into the text field is then hashed with the PBKDF2 algorithm (salt: salt, iterations: 10).

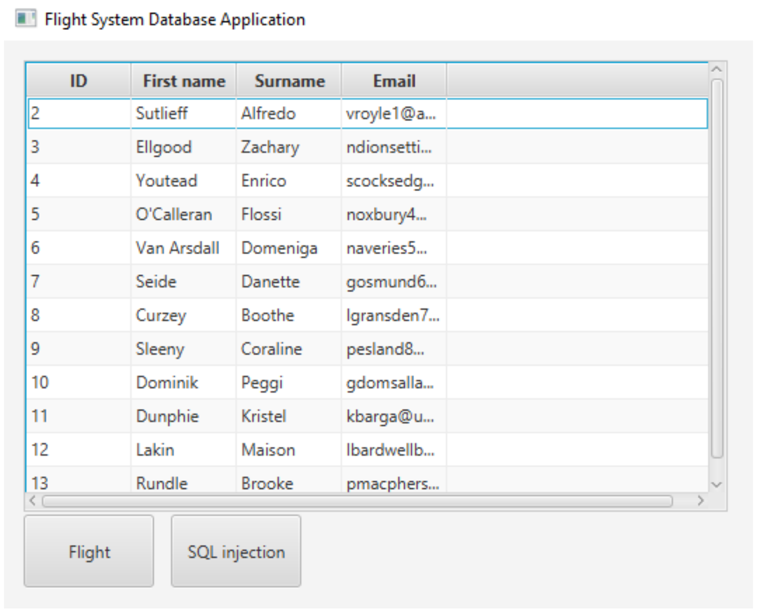


hash@seznam.cz

••••

3.2 Basic View Window

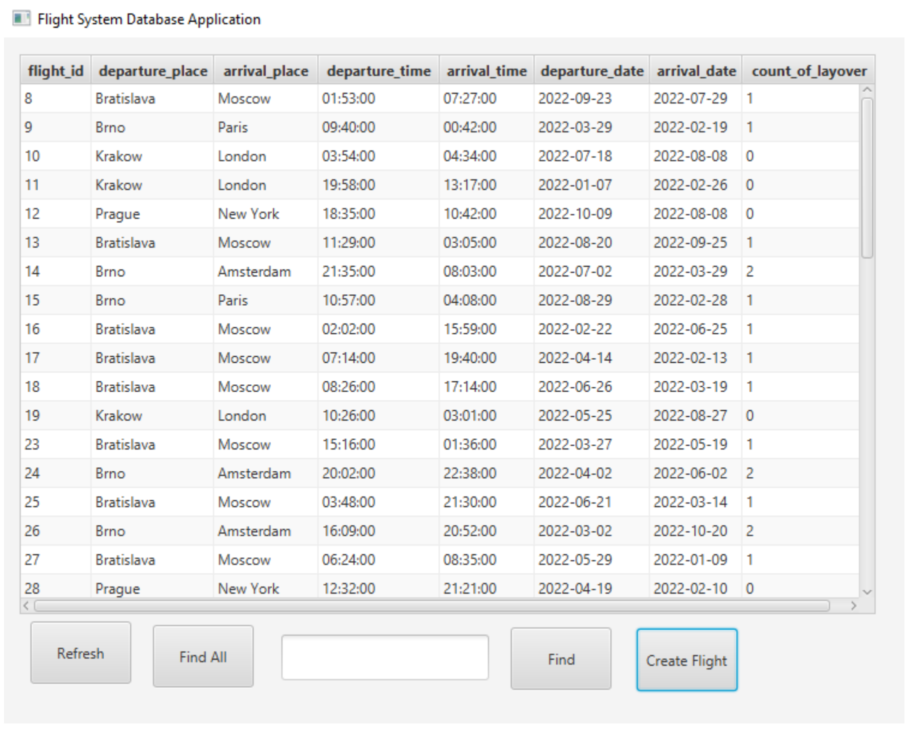
This is the first window you will see after logging in, it displays each "passenger" who can log in into the program. It also has two buttons. One leads you to a dummy window where you can do a SQL-injection attack (more on that in the prepared statement section). The second leads you to a table view with different flights.



ID	First name	Surname	Email
2	Sutlieff	Alfredo	vroyle1@a...
3	Ellgood	Zachary	ndionsetti...
4	Youtead	Enrico	scocksedg...
5	O'Calleran	Flossi	noxbury4...
6	Van Arsdall	Domeniga	naveries5...
7	Seide	Danette	gosmund6...
8	Curzey	Boothe	lgransden7...
9	Sleeny	Coraline	pesland8...
10	Dominik	Peggi	gdomsalla...
11	Dunphie	Kristel	kbarga@u...
12	Lakin	Maison	lbardwellb...
13	Rundle	Brooke	pmacphers...

3.3 Flight View Window

In this window you can see every flight in the database. This table is the "flight" and "flight_places" joined tables. With the right click on a flight you are going to see a menu, through which you can look at a detail view, edit or the the flight. In the detail view you can see information about airplanes on top the information shown in the flight view.



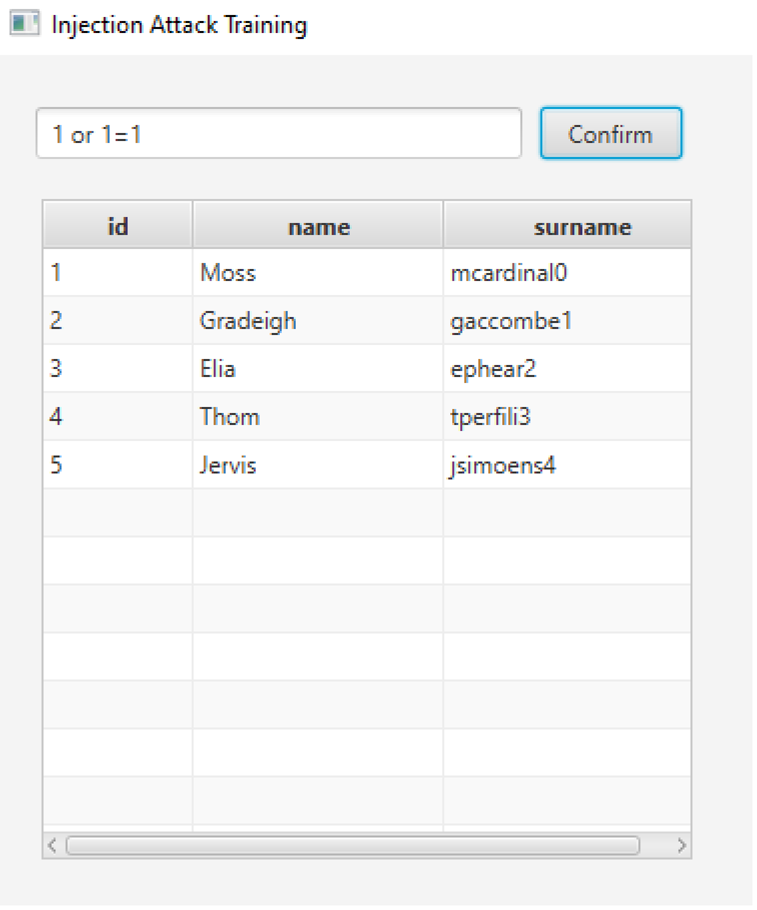
flight_id	departure_place	arrival_place	departure_time	arrival_time	departure_date	arrival_date	count_of_layover
8	Bratislava	Moscow	01:53:00	07:27:00	2022-09-23	2022-07-29	1
9	Brno	Paris	09:40:00	00:42:00	2022-03-29	2022-02-19	1
10	Krakow	London	03:54:00	04:34:00	2022-07-18	2022-08-08	0
11	Krakow	London	19:58:00	13:17:00	2022-01-07	2022-02-26	0
12	Prague	New York	18:35:00	10:42:00	2022-10-09	2022-08-08	0
13	Bratislava	Moscow	11:29:00	03:05:00	2022-08-20	2022-09-25	1
14	Brno	Amsterdam	21:35:00	08:03:00	2022-07-02	2022-03-29	2
15	Brno	Paris	10:57:00	04:08:00	2022-08-29	2022-02-28	1
16	Bratislava	Moscow	02:02:00	15:59:00	2022-02-22	2022-06-25	1
17	Bratislava	Moscow	07:14:00	19:40:00	2022-04-14	2022-02-13	1
18	Bratislava	Moscow	08:26:00	17:14:00	2022-06-26	2022-03-19	1
19	Krakow	London	10:26:00	03:01:00	2022-05-25	2022-08-27	0
23	Bratislava	Moscow	15:16:00	01:36:00	2022-03-27	2022-05-19	1
24	Brno	Amsterdam	20:02:00	22:38:00	2022-04-02	2022-06-02	2
25	Bratislava	Moscow	03:48:00	21:30:00	2022-06-21	2022-03-14	1
26	Brno	Amsterdam	16:09:00	20:52:00	2022-03-02	2022-10-20	2
27	Bratislava	Moscow	06:24:00	08:35:00	2022-05-29	2022-01-09	1
28	Prague	New York	12:32:00	21:21:00	2022-04-19	2022-02-10	0

4 Prepared Statements

I access the database through prepared statements, which are pre-done queries. The application just sends the variables. Prepared statements are the industry standard when it comes to security, but there are a lot of systems which still do not use them. Systems not using prepared statements are prone to an attack called SQL-injection.

SQL-injection is where you end a SQL query with a semicolon (usually through a text field) and then you type whatever command you want. If the application just takes whatever you wrote in the text field and sends it to a database, it is a big security flaw.

I made one window which filters (by ID) rows in this dummy table. As you can see, I am able to access more data than I should if I write "1 OR 1=1". I also could end the query with a semicolon and then for example write: "; DROP TABLE injection" which would destroy the database altogether.



id	name	surname
1	Moss	mcardinal0
2	Gradeigh	gaccombe1
3	Elia	ephear2
4	Thom	tperfili3
5	Jervis	jsimoens4

5 Conclusion

Overall, my goal of making an application that can show you and allow you to work with flight information was successful. However, during the development I encountered many problems with my initial design, which then prevented me from for example deleting some rows because of foreign key constraints. I also could have implemented some quality of life features, such as a working window which warns you before deleting a user, although that would be really time consuming.