

Database project assignment 3

Lukáš Král

Obsah

Úvod	2
Popis aplikace	2
V aplikaci.....	2
Autentizace uživatele	2
Po přihlášení	3
Edit employee	4
Detailed employee view	4
Delete employee	5
Add employee	5
Požadavky na projekt	6
CMD	6
Bcrypt.....	6
Schéma/role.....	7
Více dotazů	7
SQL Injection simulace.....	7
PreparedStatement vs Statement	9
Záloha databáze každou půlnoc	9
Archivace logů	10
Github Repository.....	10

Úvod

Cílem třetího projektu v kurzu BPC-BDS bylo vytvořit databázovou aplikaci s GUI. Pro realizaci jsem zvolil prostředí IntelliJ, jazyk Java a pro vytvoření desktopové aplikace JavuFX. Projekt byl završením tohoto kurzu a vycházel z přechozích projektů, kdy byla databáze vytvořena.

Popis aplikace

Aplikace slouží ke správě zaměstnanců automobilky. Do aplikace mají přístup manažeři automobilky, přihlašující se pomocí emailu a hesla, které je šifrováno pomocí BCryptu.

V aplikaci

Autentizace uživatele

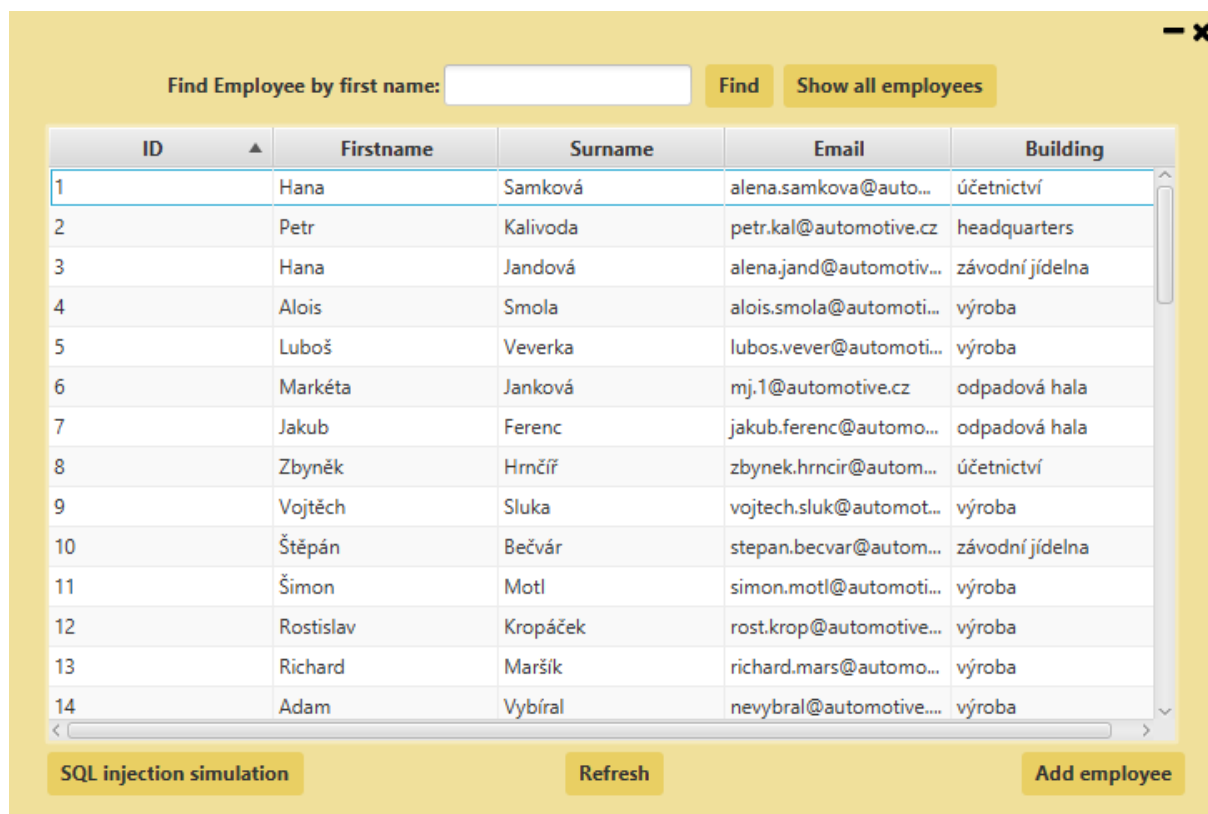
Po spuštění aplikace se objeví okno přihlášení, kam uživatel zadá email a heslo příslušného manažera. Pokud jsou údaje správné, vyskočí informační okno o úspěšném přihlášení, v opačném případě okno o zamítnutí přístupu.



The screenshot shows a JavaFX window titled "Database of car factory employees". The window has a yellow header bar with a title bar (minimize, maximize, close buttons) in the top right corner. The main content area is white. It contains two labels with icons: "User Name:" with a person icon and "Password:" with a key icon. Each label is followed by a text input field. The "User Name" field is empty, and the "Password" field contains the placeholder text "type your password". Below the input fields is a yellow button with a right-pointing arrow and the text "Sign in".

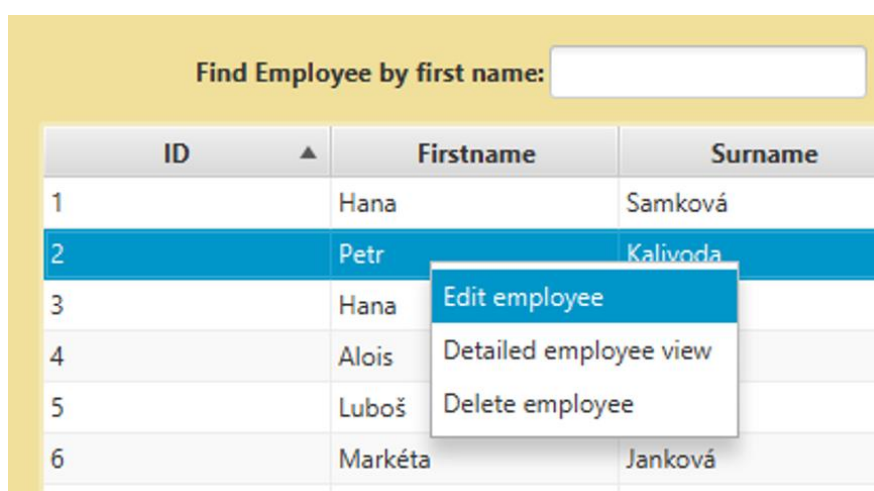
Po přihlášení

Jakmile se uživatel úspěšně přihlásí, dostane se do výchozí části programu, zde vidí všechny zaměstnance automobilky, může je vyhledat pomocí křestního jména, nebo kliknutím na příslušný nadpis sloupce zobrazit zaměstnance např. abecedně podle budovy ve které pracují.



ID	Firstname	Surname	Email	Building
1	Hana	Samková	alena.samkova@auto...	účetnictví
2	Petr	Kalivoda	petr.kal@automotive.cz	headquarters
3	Hana	Jandová	alena.jand@automotiv...	závodní jídelna
4	Alois	Smola	alois.smola@automoti...	výroba
5	Luboš	Veverka	lubos.vever@automoti...	výroba
6	Markéta	Janková	mj.1@automotive.cz	odpadová hala
7	Jakub	Ferenc	jakub.ferenc@automo...	odpadová hala
8	Zbyněk	Hrnčíř	zbynek.hrncir@autom...	účetnictví
9	Vojtěch	Sluka	vojtech.sluk@automot...	výroba
10	Štěpán	Bečvár	stepan.becvar@autom...	závodní jídelna
11	Šimon	Motl	simon.motl@automoti...	výroba
12	Rostislav	Kropáček	rost.krop@automotive...	výroba
13	Richard	Maršík	richard.mars@automo...	výroba
14	Adam	Vybíral	nevybral@automotive....	výroba

Pravým kliknutím na zaměstnance se zobrazí možnosti: Edit employee, Detailed employee view a delete employee.



ID	Firstname	Surname
1	Hana	Samková
2	Petr	Kalivoda
3	Hana	
4	Alois	
5	Luboš	
6	Markéta	Janková

Edit employee

V Edit employee je možnost(jak už název napovídá) upravit zaměstnance, zde uživatel může změnit základní informace o zaměstnanci.

id 2

email petr.kal@automotive.cz

first name Petr

last name Kalivoda

building headquarters

edit employee

Adam	Vybiral	nevybral@automotive....	vyroba
------	---------	-------------------------	--------

Detailed employee view

Zde uživatel vidí detailní pohled na zaměstnance.

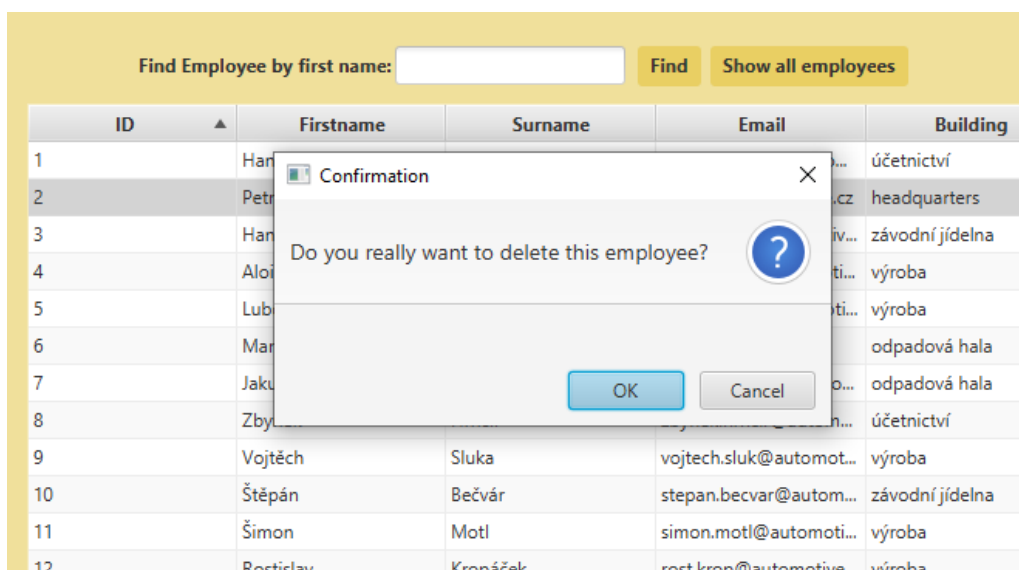
Employee's details

ID 2

first name	Petr	surname	Kalivoda
email	petr.kal@automotive.cz	building	headquarters
job type	účetní	salary	29500
contract expiration	2023-01-01	city	Zajecov
street	K Lukárně	street number	5188
zip code	26736	address type	Korespondenční adresa

Delete employee

Po zvolení nejprve vyskočí ujišťovací okno a pokud uživatel klikne na OK, zaměstnanec se smaže.



Add employee

V pravém dolním rohu výchozí části programu se nachází možnost add employee.

The screenshot shows a form for adding a new employee. It has a yellow background and a title bar with a close button. The form contains five input fields: "email", "first name", "surname", "password", and "building". The "first name", "surname", and "password" fields have placeholder text: "type new employee first name", "type new employee surname", and "type new employee password" respectively. The "building" field is a dropdown menu. To the right of the input fields is an illustration of a man and a woman in business attire. At the bottom right of the form is a "create" button.

Požadavky na projekt

CMD

Aplikace je spustitelná z příkazového řádku pomocí Mavenu.

```
C:\Windows\System32\cmd.exe - java -jar car_factory-1.0.0.jar
```

```
[WARNING] single version of the class is copied to the uber jar.
[WARNING] Usually this is not harmful and you can skip these warnings
[WARNING] otherwise try to manually exclude artifacts based on
[WARNING] mvn dependency:tree -Ddetail=true and the above output.
[WARNING] See http://maven.apache.org/plugins/maven-shade-plugin/
[INFO] Replacing original artifact with shaded artifact.
[INFO] Replacing C:\all\VUT-ukoly\Třetí semestr\BDS\projekt3\car_factory\
car_factory\target\car_factory-1.0.0-shaded.jar
[INFO] Dependency-reduced POM written at: C:\all\VUT-ukoly\Třetí seme
[INFO]
[INFO] --- maven-install-plugin:2.4:install (default-install) @ car_fa
[INFO] Installing C:\all\VUT-ukoly\Třetí semestr\BDS\projekt3\car_fac
\1.0.0\car_factory-1.0.0.jar
[INFO] Installing C:\all\VUT-ukoly\Třetí semestr\BDS\projekt3\car_fac
.0.0\car_factory-1.0.0.pom
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 6.147 s
[INFO] Finished at: 2021-12-31T11:08:02+01:00
[INFO] -----

C:\all\VUT-ukoly\Třetí semestr\BDS\projekt3\car_factory>cd target

C:\all\VUT-ukoly\Třetí semestr\BDS\projekt3\car_factory\target>java -jar car_factory-1.0.0.jar
pro 31, 2021 11:08:20 DOP. com.sun.javafx.application.PlatformImpl startup
WARNING: Unsupported JavaFX configuration: classes were loaded from 'unnamed module @a89523d'
LoginController initialized
```

Database of car factory empl

User Name:

Password:

type your password

Bcrypt

Aplikace pracuje s hesly hashované BCryptem.

	email character varying (45)	pwd character varying (60)
1	hladky.miroslav@automotive.cz	\$2a\$12\$R2cCMfzhkntzHxzQ4fdkL.F8.iUwp0ziCwS242JJhx8qalFeVCfCC
2	syrova.natalie@automotive.cz	\$2a\$12\$R2cCMfzhkntzHxzQ4fdkL.F8.iUwp0ziCwS242JJhx8qalFeVCfCC
3	bednarik.rudolf@automotive.cz	\$2a\$12\$R2cCMfzhkntzHxzQ4fdkL.F8.iUwp0ziCwS242JJhx8qalFeVCfCC
4	stupkova.alzbeta@automotive.cz	\$2a\$12\$R2cCMfzhkntzHxzQ4fdkL.F8.iUwp0ziCwS242JJhx8qalFeVCfCC
5	simcik.marcel@automotive.cz	\$2a\$12\$R2cCMfzhkntzHxzQ4fdkL.F8.iUwp0ziCwS242JJhx8qalFeVCfCC

Schéma/role

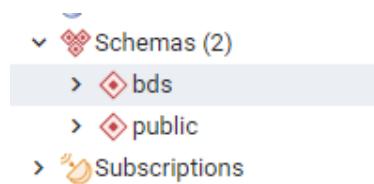
Byla vytvořena a používána role batman, která není superuserem a byli ji přiřazeny příslušné privilegia.

```
2
3 CREATE ROLE batman NOSUPERUSER LOGIN ENCRYPTED PASSWORD 'batman';
4 GRANT CONNECT ON DATABASE "automobilka" TO batman;
5 GRANT USAGE ON SCHEMA bds TO batman;
6 GRANT SELECT, INSERT, UPDATE, DELETE, REFERENCES, TRIGGER, TRUNCATE ON ALL TABLES IN SCHEMA bds TO batman;
7 GRANT ALL PRIVILEGES ON ALL SEQUENCES IN SCHEMA bds TO batman;
```

Data Output Explain Messages Notifications

GRANT

Bylo vytvořeno schéma bds ve kterém se nacházejí data se kterými aplikace pracuje.



```
datasource.url=jdbc:postgresql://localhost:5432/car_factory|
datasource.username=batman
datasource.password=batman
```

Více dotazů

Aplikace obsahuje operaci, která vyvolává více než 1 dotaz, pokud je něco špatně, použije se rollback(EmployeeRepository - editEmployee).

```
public void editEmployee(EmployeeEditView employeeEditView) {
    String insertEmployeeSQL = "UPDATE bds.employee e SET email = ?, first_name = ?, surname = ?, building_id = ? WHERE e.employee_id = ?";
    String checkIfExists = "SELECT email FROM bds.employee e WHERE e.employee_id = ?";
    try (Connection connection = DataSourceConfig.getConnection();
        PreparedStatement preparedStatement = connection.prepareStatement(insertEmployeeSQL, Statement.RETURN_GENERATED_KEYS)) {
```

SQL Injection simulace

V databázi byla vytvořena tabulka dummy_table která slouží k simulaci útoku sql injection.

	Data Output	Explain	Messages	Notifications
	employee_id [PK] integer	name character varying (40)	email character varying (40)	
1	1	Pepa Novák	pepa.novak@seznam.cz	
2	2	Aneta Žaneta	zaneta.aneta@seznam.cz	
3	3	Jaromír Rukavica	rukavicka.jara@seznam.cz	
4	4	Lucie Oranžová	oranzova.lucie@seznam.cz	

Dále bylo vytvořeno GUI do kterého se dostaneme po kliknutí na SQL Injection simulation, nacházející se v levém dolním rohu výchozího programu.

Find Employee by first name:

ID	Firstname	Surname	Email	
1	Hana	Samková	alena.samkova@auto...	účet
2	Petr	Kalivoda	petr.kal@automotive.cz	head
3	Hana	Jandová	alena.jand@automotiv...	závc
4	Alois	Smola	alois.smola@automoti...	výro
5	Luboš	Veverka	lubos.vever@automoti...	výro
6	Markéta	Janková	mj.1@automotive.cz	odp
7	Jakub	Ferenc	jakub.ferenc@automo...	odp
8	Zbyněk	Hmčír	zbynek.hmci@autom...	účet
9	Vojtěch	Štika	vojtech.sluk@automot...	výro
10	Štěpán	Bečvář	stepan.becvar@autom...	závc
11	Šimon	Motl	simon.motl@automoti...	výro
12	Rostislav	Kropáček	rost.krop@automotive...	výro
13	Richard	Maršík	richard.mars@automo...	výro
14	Adam	Vybrál	nevybral@automotive...	výro

-x

ID:

Hint for injection: instead of just number(id) type e.g. 1 or 1=1 for seeing all rows of table or 1; DROP TABLE bds.dummy_table for deleting whole table

ID	Full name	Email
No content in table		

Zde můžeme simulovat sql injection útok. SQL Injection je druh útoku na databázi kdy při nedostatečné ochraně vstupu do databáze může útočník získat více informací, než tvůrce aplikace zamýšlel nebo dokonce vymazat celou tabulku ze které se vyhledávaná data vytahují.

Pro potřebu simulace je tudíž vstup pro zadávání hledaného id neošetřen.

```
public List<SQLInjectionView> findByIdStatement(String id) {
    String sqlResult = "SELECT employee_id, email, name FROM bds.dummy_table WHERE employee_id=" + id ;
    try (Connection connection = DataSourceConfig.getConnection();
        Statement statement = connection.createStatement();
        ResultSet resultSet = statement.executeQuery(sqlResult)) {
        List<SQLInjectionView> SQLInjectionViews = new ArrayList<>();
        while (resultSet.next()) {
            SQLInjectionViews.add(mapToSQLInjectionView(resultSet));
        }
    }
}
```

Díky tomu že id vstupuje do dotazu jako prostý string a je použit pouze statement, tak uživatel může zadat do id prakticky cokoli a díky znalostem může buď zobrazit všechny uživatele, nebo dokonce tabulku vymazat. Jak napovídá nápověda v GUI stačí zadat např 1 or 1=1 a aplikace nám místo jednoho konkrétního zaměstnance vyhodí všechny. Při předpokladu, že útočník uhodne jméno tabulky(v našem případě je to těžší právě z důvodu použití jiného schématu než public), může zadáním 1; DROP TABLE bds.dummy_table celou tabulku vymazat. V takovém případě již vyhledávání nebude funkční – útok byl úspěšný. Pro obnovení tabulky stačí kliknout na Create table.

PreparedStatement vs Statement

Pouhý statement je, jak je vidět výše, zranitelný na útok sql injection, je tedy vhodné ho používat pouze tam, kde neočekáváme vstup od uživatele. V opačném případě je vhodné použít PreparedStatement – je vhodnější na opakované dotazy a string se vkládá přímo do query.

Záloha databáze každou půlnoc

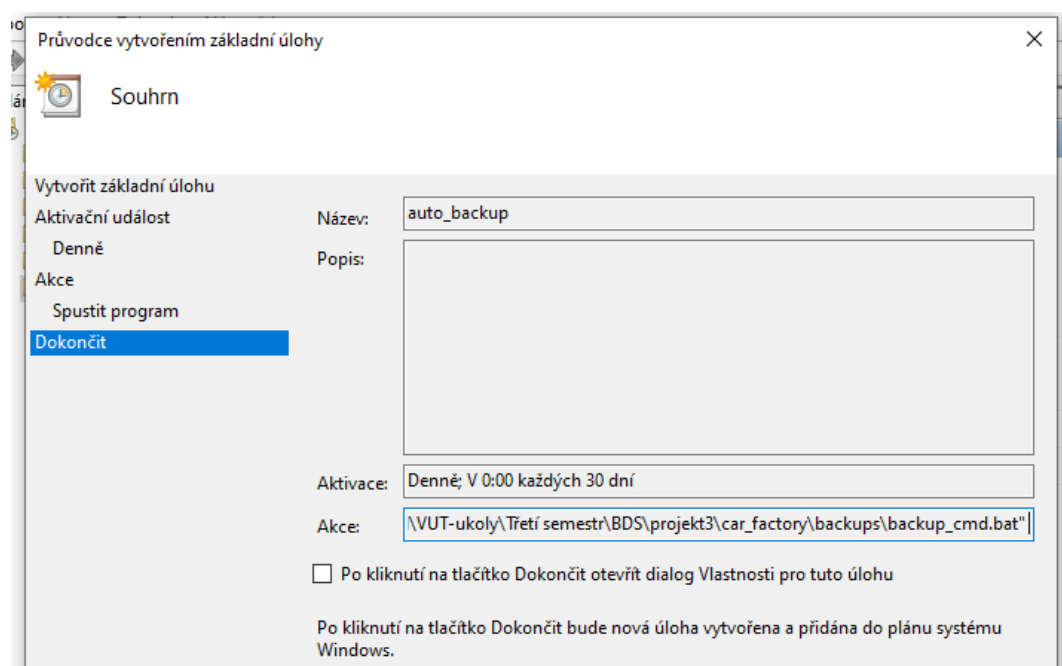
Pro nastavení zálohování databáze každou půlnoc v systému Windows jsem použil Task Scheduler.

Nejprve jsem vytvořil .bat soubor do kterého jsem napsal příslušný command pro zálohování databáze a nastavil heslo pro postgres.

```
*backup_cmd.bat – Poznámkový blok
Soubor Úpravy Formát Zobrazení Nápověda
set PGPASSWORD=postgres

pg_dump -h localhost -U postgres -f car_factory_backup car_factory|
```

Poté už stačilo v Task scheduleru vytvořit nový Basic Task, nastavit aktivaci na každou půlnoc a jako akci zvolit vytvořený bat. Soubor.



Archivace logů

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
  <!-- check the logback documentation -> http://logback.qos.ch/manual/appenders.html -->
  <appender name="FILE" class="ch.qos.logback.core.rolling.RollingFileAppender">
    <file>logs/bds-java-logs.log</file>
    <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
      <!-- daily rollover -->
      <fileNamePattern>logFile.%d{yyyy-MM-dd}.log</fileNamePattern>
      <!-- keep 30 days' worth of history capped at 3GB total size -->
      <maxHistory>1</maxHistory>
      <totalSizeCap>500MB</totalSizeCap>
    </rollingPolicy>
    <encoder>
      <pattern>%-4relative [%thread] %-5level %logger{35} - %msg%n</pattern>
    </encoder>
  </appender>

  <appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender">
    <encoder>
      <pattern>%msg%n</pattern>
    </encoder>
  </appender>

  <logger name="bds.javafx">
    <appender-ref ref="FILE"/>
  </logger>

  <root level="debug">
    <appender-ref ref="STDOUT"/>
  </root>
</configuration>
```

Github Repository

https://github.com/stroyec/bds_project_3