Assignment11

Report:

Description of how your code works:

Let’s start with the main function.
At line 112, I have written the functionality to check if a log file exists and if not then create one.

Line 124, I started with SQL jdbc connection. I have set the SQLite DB Url and opened the connection.

Then I have created an ArrayList to store the Input Modes so I can check if proper input mode is provided when running the program.
On line 140, I have calculated the No of Inputs so I can make sure each Input Mode gets a valid number of Input parameters.

After it I have written regular expressions for Phone Number and Name
.
For Phone number regex, first I have used + character for zero or one occurrence, international code in digits for 0 or more occurrences, then for remaining number I have used digits, hyphen character (0 or more occurences), open and closing bracket (zero or more occurences), dot (0 or more occurences) and group of all these character and digits between 5 to 20 length, dollar sign to make sure regex ends with this.

For Name regex, first I have used AtoZ zero or one occurrence for First capital letter and first Letter before single quote(zero or one occurrence)(eg. O'Reilly). Then I have used capital letter one occurrence and lower case letters both in groups of 2 to 20. For comma character(zero or one occurrence), space(one or more occurences), then again same as First line for Last name occurrence and other character’s occurrences. And then entire group after command to zero or one occurrence to prevent extra words than 3.

Then on line 145, I am checking if the first input is equal to which mode does it have the appropriate number of inputs to run the Mode operation. Inside it I am checking the phone number and/or Name regex and exiting with code 1 if the regex does not match. Then I am calling the respective function to the Mode and performing DB operations. After it I have used catch block to catch any expectations and Finally block to make sure SQL connection is closed and sys exit zero for successful execution.

In each function which performs operation on DB, passing connections and parameters from Main function, I am first forming a query with a Prepared statement, and performing DB operation.Try catch block to get the exception and print the error message.
And For logs, I am writing the present date and what operations performed by which user in the audit logs file created in the main function.

Compilation/build instructions:

For compilation requirements are Java, Jdbc jar file(download from https://github.com/xerial/sqlite-jdbc/releases)3.36.0 version.
Download the java assignment11 code and unzip it.

Step 1: Traverse to the `/home/seed/Assignment11/FinalProject/FinalProject/src` directory and copy the jdbc jar there.
Step 2: In the src folder, there is a App.java file and sqlite-jdbc-3.36.0.jar file. Now compile it using the javac command.

```
javac App.java
```

Installation, setup, and execution instructions:
Java must be already installed so let's move on to the next steps.

Step 1: DB setup - SQLite
Install DB
Make sure you have mydb.db file in the `/home/seed/Assignment11/db` directory. (There is a create command in the Create.sql file. There is Drop command in the DROP.sql file and Insert commands in Insert.sql file.)

1.1 Download sqlite3 and install

```
sudo apt install sqlite3
```

1.2 To check the DB is working you can use below commands

```
sqlite3 /home/seed/Assignment11/db/mydb.db
```

Now in sqlite terminal run .tables command you will see **personinfo** table.

```
.tables
```

Make sure jdbc url is set to this location.

```
String jdbcURL =
"jdbc:sqlite:/home/seed/Assignment11/db/mydb.db";
```

Step 2: Run the program

Now go to the `/home/seed/Assignment11/FinalProject/FinalProject/src` directory, where you have compiled the code before. So it has an App.class file.

2.1 If not, compile it.(try *sudo* for below command)

```
javac App.java
```

2.2 Make it setuid program

```
sudo chown root App.class
sudo chmod 4755 App.class
```

2.3 To run it use below commands as per each operation.

```
java -cp ".:sqlite-jdbc-3.36.0.jar" App <Mode> <Name> <Number>
```

Eg.

```
java -cp ".:sqlite-jdbc-3.36.0.jar" App ADD 'Jskljds' '36345'
java -cp ".:sqlite-jdbc-3.36.0.jar" App DEL 'Jskljds'
java -cp ".:sqlite-jdbc-3.36.0.jar" App LIST
```

Assumptions you have made:

I have made assumptions that for one execution only one Mode operation can be performed. I have made assumptions about the number of inputs required for each mode.

For LIST Mode, I am making sure there are no input data parameters.

For DEL Mode, I am making sure there are 1 input data parameters.(either Phone Number or Name).

For ADD Mode, I am making sure there are 2 input data parameters.(Phone Number and Name)

Pros/Cons of your approach:

Pros:

1. I have used separate functions to perform DB operations.

2. I have used regex for input validation.
3. I am making sure each operation gets the required number of inputs and not more than that.
4. I have used DB to store the data.I am creating and closing the DB connections after execution. Prepared statement for stopping injection attacks.
5. I am writing logs for each operation.I am creating and closing the file writer connections after execution.
6. Handling the exceptions and errors properly so the program does not fail.

Cons:

1. Need to optimize logic
2. Need to break functions into more functions as per requirements to maintain the code
3. Maybe few more error handlings are required