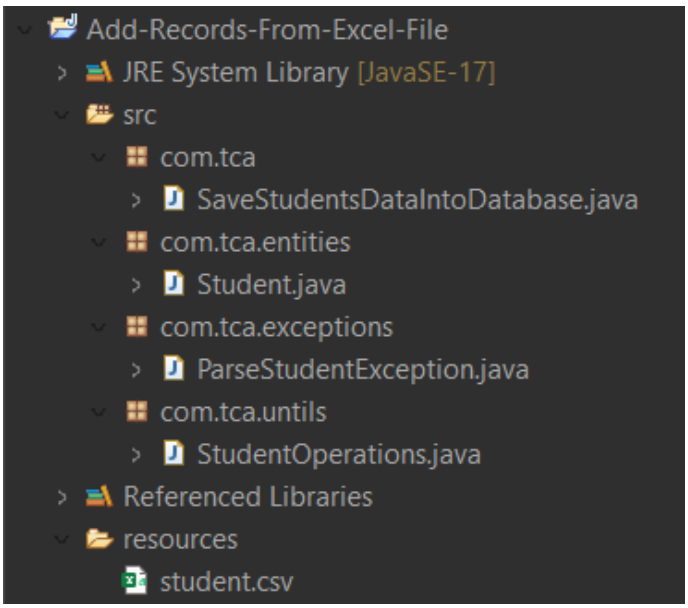
Add Student Records From CSV file into database

Project Structure:



Output:-

```
{Student [rollNo=101, name=AAA, per=63.0]} is inserted successfully.
{Student [rollNo=102, name=BBB, per=49.0]} is inserted successfully.
{Student [rollNo=103, name=CCC, per=59.0]} is inserted successfully.
{Student [rollNo=104, name=DDD, per=99.0]} is inserted successfully.
{Student [rollNo=105, name=EEE, per=98.0]} is inserted successfully.
{Student [rollNo=106, name=FFF, per=89.0]} is inserted successfully.
{Student [rollNo=107, name=GGG, per=77.0]} is inserted successfully.
{Student [rollNo=108, name=HHH, per=95.0]} is inserted successfully.
{Student [rollNo=109, name=III, per=85.0]} is inserted successfully.
{Student [rollNo=110, name=JJJ, per=75.0]} is inserted successfully.
{Student [rollNo=111, name=KKK, per=60.0]} is inserted successfully.
{Student [rollNo=112, name=LLL, per=50.0]} is inserted successfully.
{Student [rollNo=113, name=MMM, per=70.0]} is inserted successfully.
{Student [rollNo=114, name=NNN, per=80.0]} is inserted successfully.
{Student [rollNo=115, name=OOO, per=90.0]} is inserted successfully.
```

Student.java

```java
package com.tca.entities;

public class Student {

	private short rollNo;
	private String name;
	private float per;

	public Student() {}

	public Student(String name,short rollNo, float per) {
		super();
		this.name = name;
		this.rollNo = rollNo;
		this.per = per;
	}

	public short getRollNo() {
		return rollNo;
	}

	public void setRollNo(short rollNo) {
		this.rollNo = rollNo;
	}

	public String getName() {
		return name;
	}

	public void setName(String name) {
		this.name = name;
	}

	public float getPer() {
		return per;
	}

	public void setPer(float per) {
		this.per = per;
	}

	@Override
	public String toString() {
		return "Student [rollNo=" + rollNo + ", name=" + name + ", per=" + per + "]";
	}

}
```

SaveStudentsDataIntoDatabase.java

```java
package com.tca;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.SQLException;
import java.util.ArrayList;

import com.tca.entities.Student;
import com.tca.untils.StudentOperations;

public class SaveStudentsDataIntoDatabase {

	public static final String DB_URL = "jdbc:postgresql://localhost/mydb";
	public static final String DB_USER = "dnyaneshwar";
	public static final String DB_PASSWORD = "root@3121";

	public static void main(String[] args) {

		Connection con = null;
		PreparedStatement pst = null;

		try {

			// Loading Driver
			Class.forName("org.postgresql.Driver");

			// Forming Connection with the database
			con = DriverManager.getConnection(DB_URL, DB_USER, DB_PASSWORD);

			// Preparing SQL statement
			pst = con.prepareStatement("insert into student values(?, ?, ?)");

			double per = 0;
			int rno = 0;
			String name = null;


			ArrayList<Student> studentList =
			StudentOperations.getAllStudentsFromFile("resources/student.csv");

			if(studentList == null) {
				System.out.println("Student records not found !");
				return;
			}

			for(Student ob : studentList) {
				try {
					per = ob.getPer();
					rno = ob.getRollNo();
					name = ob.getName();

					//Update values in the prepared query
					pst.setInt(1, rno);
					pst.setString(2, name);
					pst.setDouble(3, per);

					if(pst.executeUpdate() < 1) {
						System.out.println("Unable to insert:" + ob.toString());
					}
					else {
						System.out.println("{" + ob.toString() + "} is inserted
						successfully.");
					}
				}
				catch(SQLException e){
					 //If one record is invalid don't insert it but insert all valid records.
					System.out.println("Unable to insert:" + ob.toString());
					System.out.println(e.getMessage());
				}
			}

		} catch (ClassNotFoundException e) {
			System.out.println(
					"Please make sure, you have added postgresql.jar file in classpath or
					modulepath of your project.");
			System.out.println("Please check Driver class's location !");
			e.printStackTrace();
		} catch (SQLException e) {
```

```
                        e.printStackTrace();
                } finally {
                        try {
                                con.close();
                        } catch (SQLException e) {
                                System.out.println("Unable to close the connection with database !");
                                e.printStackTrace();
                        }

                }

        }
}
```

ParseStudentException.java

```
package com.tca.exceptions;

@SuppressWarnings("serial")
public class ParseStudentException extends Exception {

        public ParseStudentException() {
        }

        public ParseStudentException(String message) {
                super(message);
        }

        public ParseStudentException(Throwable cause) {
                super(cause);
        }

        public ParseStudentException(String message, Throwable cause) {
                super(message, cause);
        }

}
```

StudentOperations.java

```java
package com.tca.untils;

import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.util.ArrayList;

import com.tca.entities.Student;
import com.tca.exceptions.ParseStudentException;

public class StudentOperations {

	public static ArrayList<Student> getAllStudentsFromFile(String filePath) {
		try (FileReader fr = new FileReader(filePath); BufferedReader br = new BufferedReader(fr)) {

			ArrayList<Student> studentList = new ArrayList<Student>();
			String line = null;

			while ((line = br.readLine()) != null) {
				try {
					studentList.add(parseStudent(line));
				}
				catch(ParseStudentException e) {
					System.out.println(e.getMessage());
				}
			}
			return studentList;

		} catch (FileNotFoundException e) {
			System.out.println("Please check the file path !");
			System.out.println(e.getMessage());
			return null;
		} catch (IOException e) {
			System.out.println("Something went wrong while doing I/O operation !");
			System.out.println(e.getMessage());
			return null;
		} catch (Exception e) {
			System.out.println(e.getMessage());
			return null;
		}
	}


	public static Student parseStudent(String studentData) throws ParseStudentException {
		if (studentData == null || studentData.equals("")) {
			throw new ParseStudentException("Student data is missing or empty.");
		}

		String tokens[] = studentData.split(",");
		if (tokens.length != 3) {
			throw new ParseStudentException("Invalid student data format: " + studentData);
		}

		try {
			Student ob = new Student(tokens[0], Short.parseShort(tokens[1]),
			Float.parseFloat(tokens[2]));
			if(ob.getPer() > 100) {
				throw new ParseStudentException("Invalid student percentage: " + ob.getPer());
			}

			return ob;

		} catch (NumberFormatException e) {
			throw new ParseStudentException("Invalid student data: " + e.getMessage(), e);
		}
	}

}
```