NEU-DPSA-2018-2019-LABREPORT-08-2

Group member:

20165164 Nora Yang

20165187 Emily Chen

20165243 Rown

Contents

[1. What we’ve done 2](#_Toc533705617)

[2. Design 3](#_Toc533705618)

[2.1 Project Structure 3](#_Toc533705619)

[2.2 Description 3](#_Toc533705620)

[2.4 Class Diagram 6](#_Toc533705621)

[2.5 Sequence Diagram 7](#_Toc533705622)

[3. Unit Testing 8](#_Toc533705623)

[4. Deployment 8](#_Toc533705624)

[5. Screenshots of GUI 9](#_Toc533705625)

# 1. What we’ve done

- 1 patterns used: State

- State machine diagram, class diagram and sequence diagram and (in report and as .jpg files)

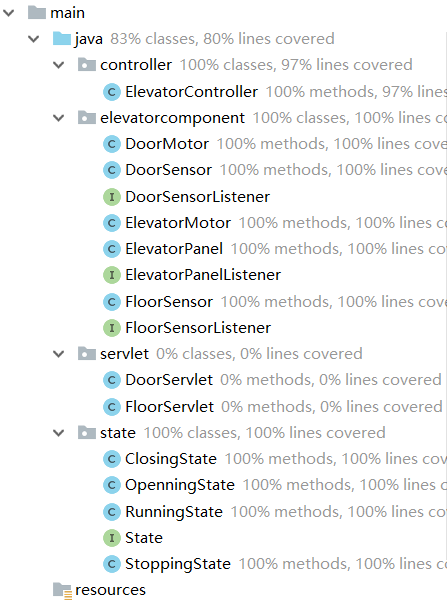
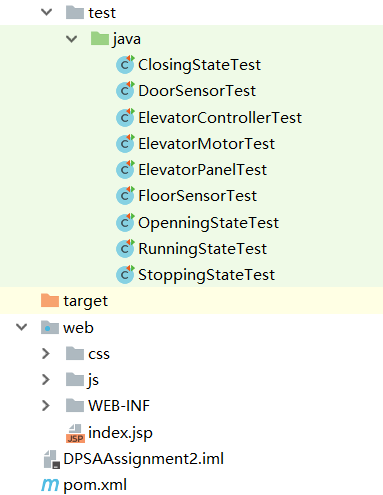
- JUnit testing

- GUI design using JSP

- Build using Maven, pom.xml and the .war file are provided

# 2. Design

## 2.1 Project Structure

## 2.2 Description

We used the state pattern, with 4 states: ClosingState, OpeningState, RunningState (elevator going up and down) and StoppingState They all implement the interface State, which has open(), close(), stop() and run(). Methods behave differently in different states. For example, the run() in OpenningState does nothing because the door must be closed first. When an elevator controller is initialized, its state is set to ClosingState.

For example, when the door open button is pressed, in the controller, openButtonPressed() is called, which will call this.state.open(). The open() of ClosingState will be called, which calls controller’s doorOpen() and then sets the controller to OpeningState.

// ElevatorController.java

**public void** openButtonPressed() {

**state**.open();

}

// ClosingState

**public void** open() {

**elevatorController**.doorOpen();

**elevatorController**.setState(**elevatorController**.**openningState**);  
}

Whenever the program initializes the controller, it also binds other related motors and sensors in the constructor. Then set the default floor to the 1st floor

**public** ElevatorController(**int** i){  
 **this**.**doorMotor**=**new** DoorMotor();  
 **this**.**elevatorMotor**=**new** ElevatorMotor();  
 **this**.**floorSensor** = **new** FloorSensor();  
 **floorSensor**.setCurrentFloor(i);  
 **this**.**elevatorPanel**=**new** ElevatorPanel(**floorSensor**);  
 **this**.setState(**closingState**);  
}

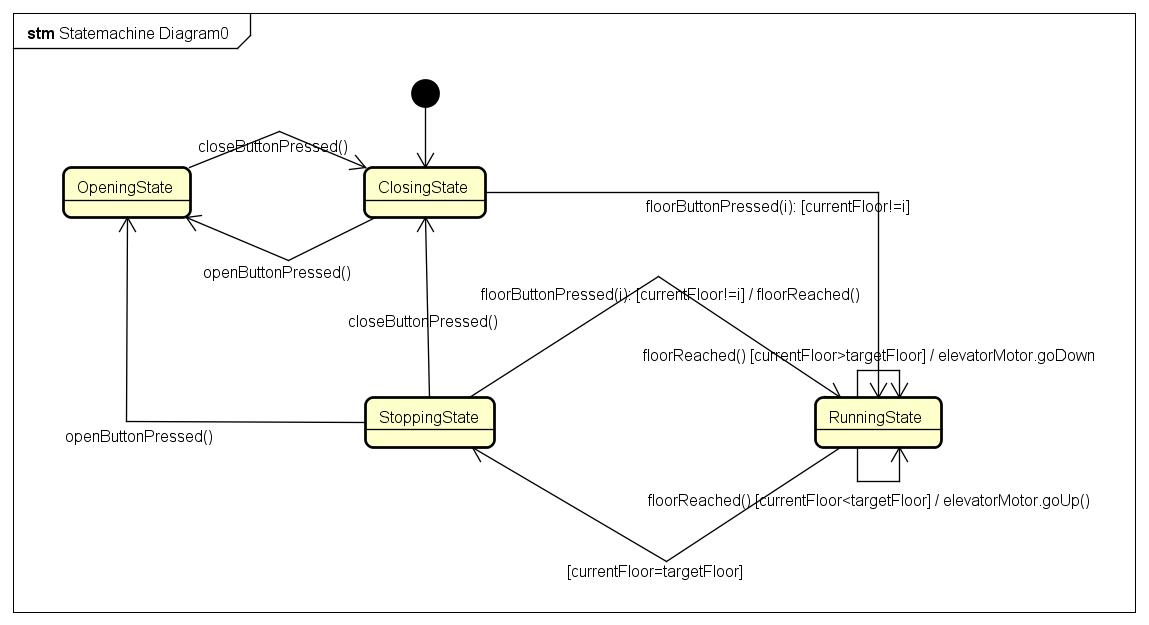
floorSensor saved the current floor and the target floor, floorSensor will also send the floor information to the elevator panel, then the panel can show the current floor.

**public** ElevatorPanel(FloorSensor floorSensor) {  
 **this**.**floorSensor** = floorSensor;  
}

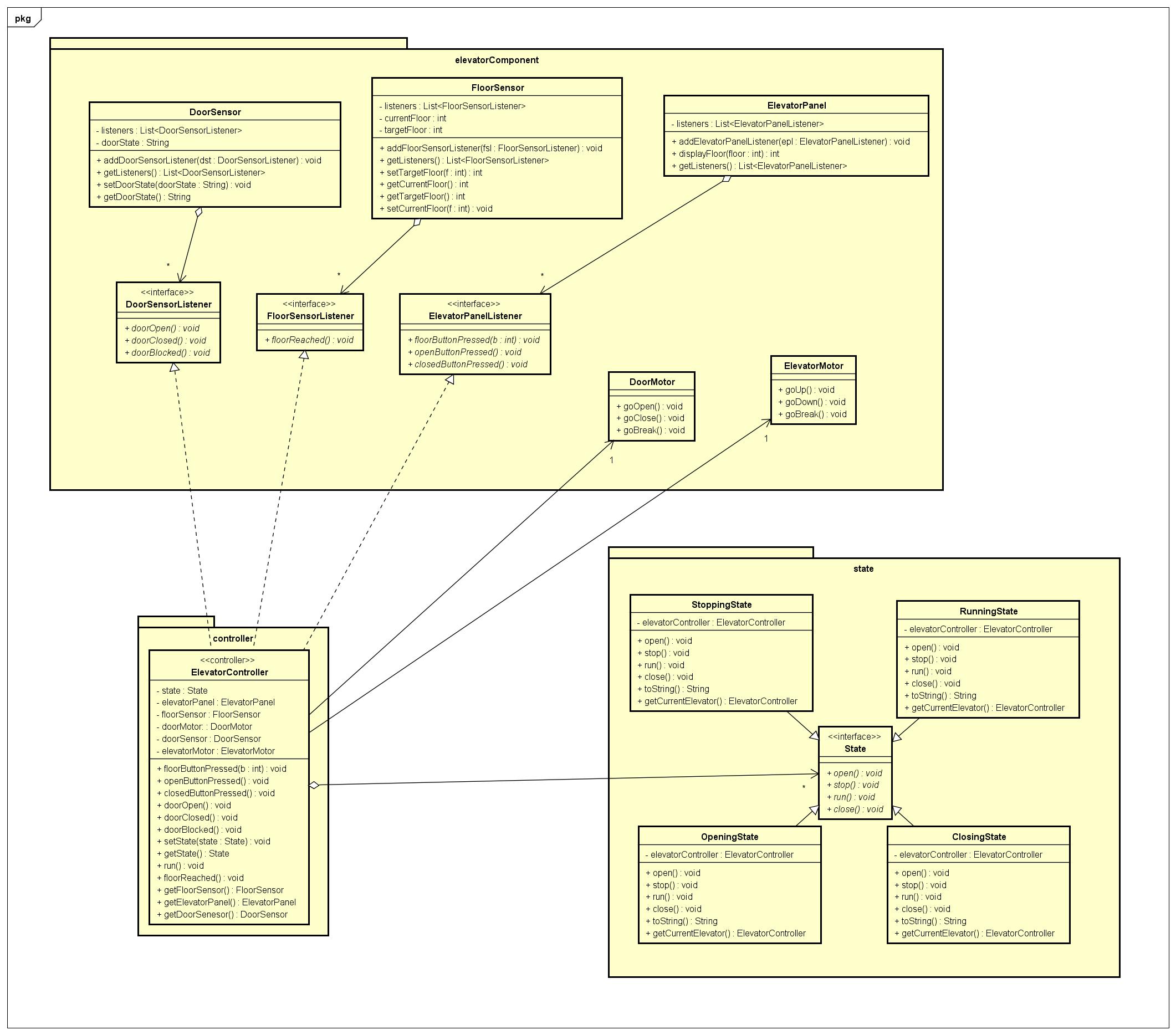
and we use floorReached(); method to let elevator go to target floor.

**public void** floorReached() {  
 **if** (**floorSensor**.getCurrentFloor()<**floorSensor**.getTargetFloor()){  
 **elevatorMotor**.goUp();  
 **floorSensor**.setCurrentFloor(**floorSensor**.getCurrentFloor()+1);  
 System.***out***.println(**"now you are in floor "**+**floorSensor**.getCurrentFloor());  
 }**else if** (**floorSensor**.getCurrentFloor()>**floorSensor**.getTargetFloor()){  
 **elevatorMotor**.goDown();  
 **floorSensor**.setCurrentFloor(**floorSensor**.getCurrentFloor()-1);  
 System.***out***.println(**"now you are in floor "**+**floorSensor**.getCurrentFloor());  
 }**else** {  
 **elevatorMotor**.goBreak();  
 }  
}

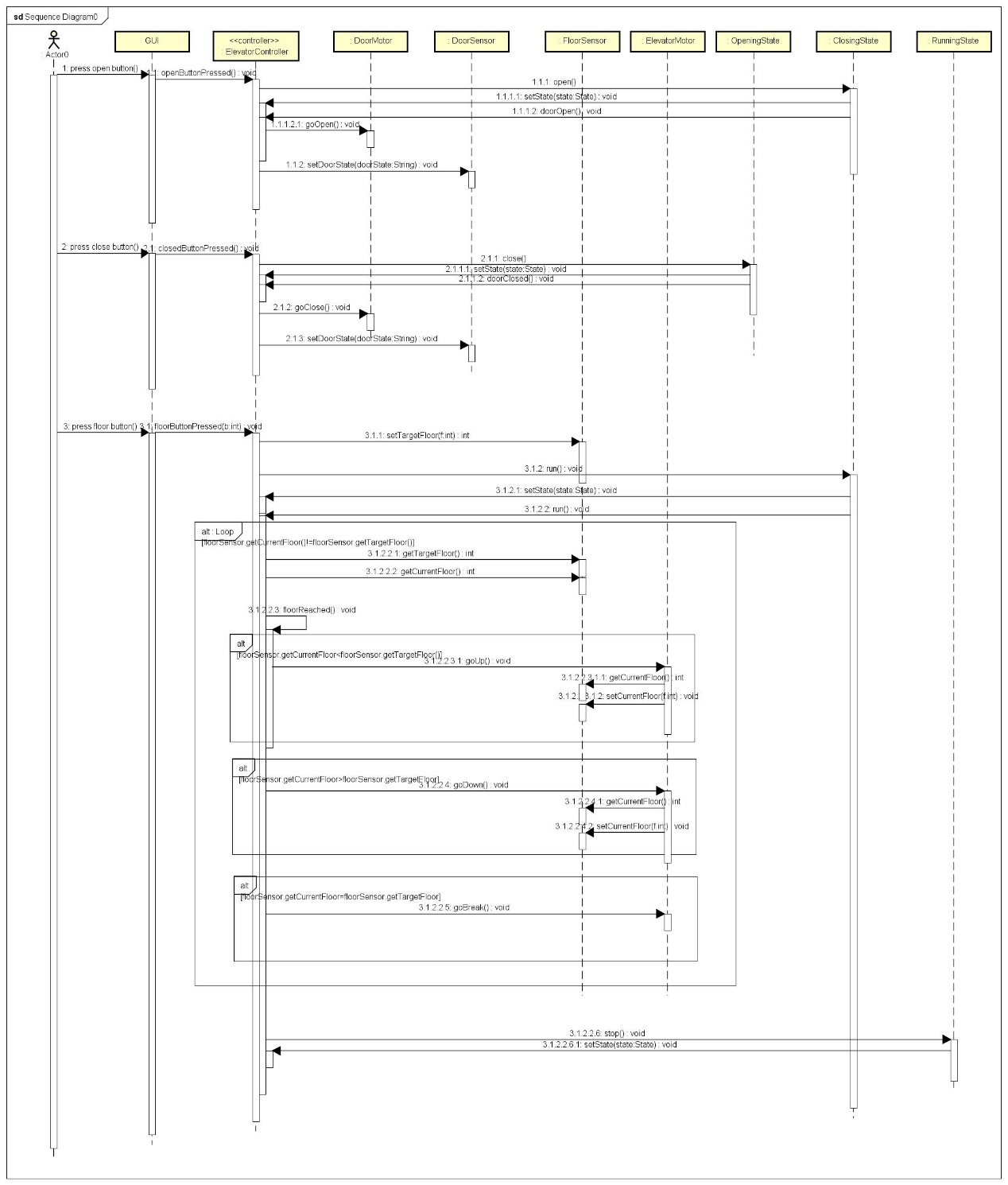
2.3 State Machine Diagram



## 2.4 Class Diagram



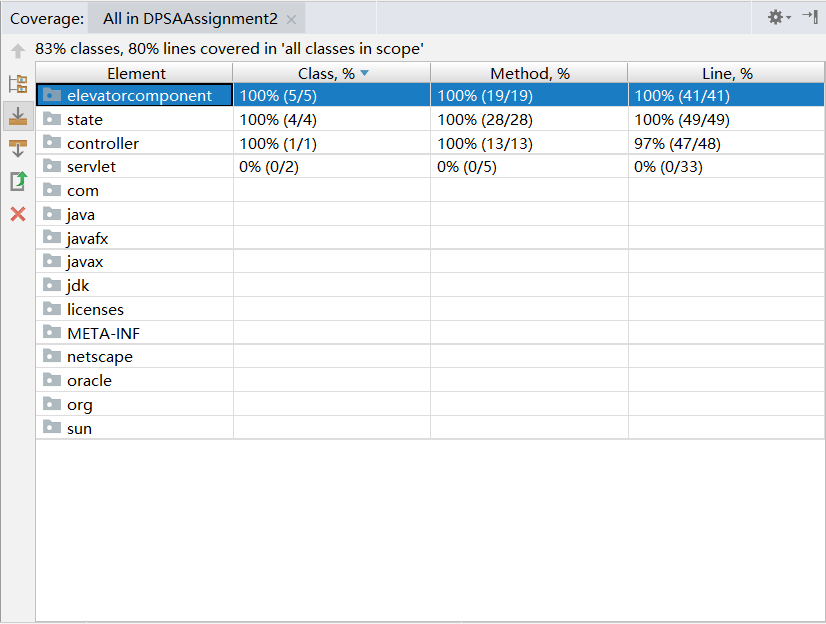
## 2.5 Sequence Diagram



# 3. Unit Testing

All tests are passed before the report is handed in.

Code coverage is shown below.



# 4. Deployment

This project is built by Maven and tested only on Chrome. It should be deployed on a Tomcat server. JDK is specified as 1.8.

A .war file is provided, and the address is:

<http://localhost:8080/DPSAAssignment2-1.0-SNAPSHOT/index.jsp>

# 5. Screenshots of GUI

