Software Engineering

343.309
2020W
Third Semester
Hogea Eduard-Florin
Studienkennzahl - 033 521

December 16, 2020

Group 2 members

1. Omar Dueñas K51849509
   roqueluisd@me.com
2. Michael Lengauer K11710940
   lengauer.michael@gmail.com
3. Mario Lischka K11712790
   mario.lishcka@gmx.at
4. Eduard-Florin Hogea K12019919
   eduard.hogea00@e-uvt.ro
5. Lukas Wais K11816105
   lukas.wais@outlook.de

Subsystems and members

Subsystem	Assigned Member
Traffic control and detection	Michael Lengauer
Traffic participants	Omar Dueñas
Control system	Lukas Wais; Mario Lischka
Road maintenance	Hogea Eduard-Florin

1 Features

The features that I plan to implement in this project, followed by a short explanation on how they are related to teammates projects are the following:

- Related to Events history, there are 2 features that i am going to implement - **add_event()** and **delete_event()**. These two are related to the Control System, more specifically to the **Backup** feature of it and are meant to store or delete events(orders made by the CEO, teams sent for emergency/maintenance) . These are important because it is good to keep track of the previous events and, if there was a mistake to remove that event and add a new, correct one.

- Features regarding actions of the CEO and those of the Worker. **removeWorker()**, **changeSalary()**, **changePosition()**, **passEmployeesToNewCEO()**, **addShift()**, **hasBirthday()**, **changeSalary()**, **changePosition()**, **changeAddress()** and CEO specific functions like **make_order()** and **send_team()**. These are linked to the **Signal** feature of the Control System and the **Maintenance** feature of the Traffic Control and Detection System. Meaning behind is that when a CEO makes an action of those two, the information about it is sent to Signal, so the API knows that there will be a team of workers sent and to the Maintenance so it is known that a maintenance is in progress.

- Account related features. Features that have to do with a worker's account. These are **create()**, **modify()** and **delete()** and can be used by the CEO. These are kept in the **Backup** feature of the Control System so in the case that anything happens to the Account's database, the information will not be lost entirely and can be recovered.

From those features, the ones that I am implementing in this milestone are the account ones. **create()**, **modify()** and **delete()** along with the ones about CEO and **Worker**. Those are **addWorker()**, **removeWorker()**, **changeSalary()**, **changePosition()**, **passEmployeesToNewCEO()**, **addShift()**, **hasBirthday()**, **changeSalary()**, **changePosition()**, **changeAddress()**.

2 Mock-up GUI

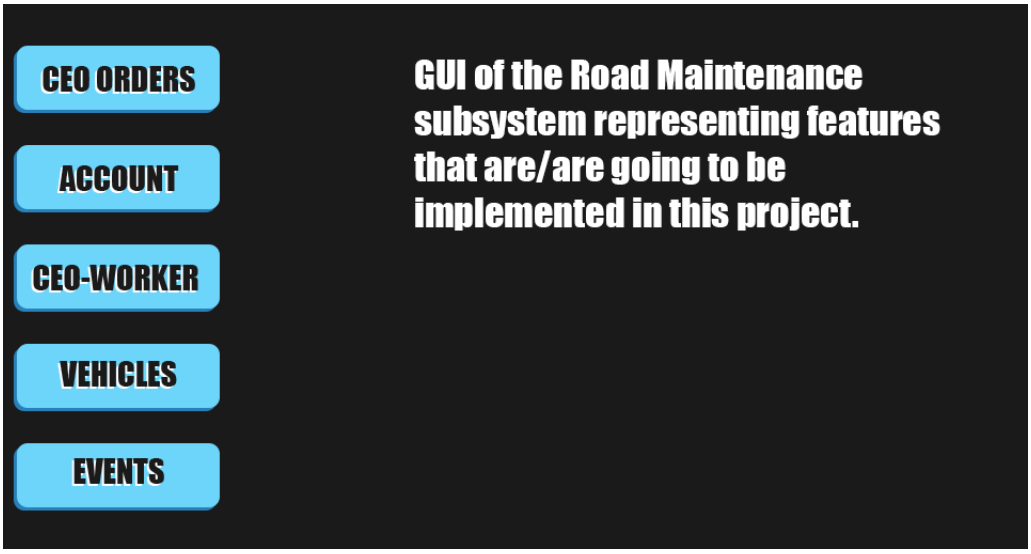
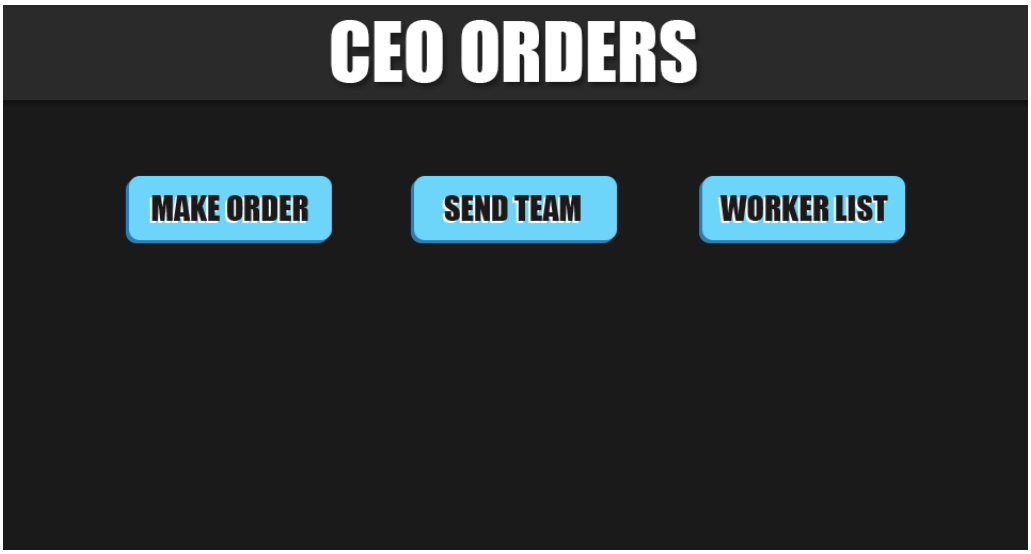
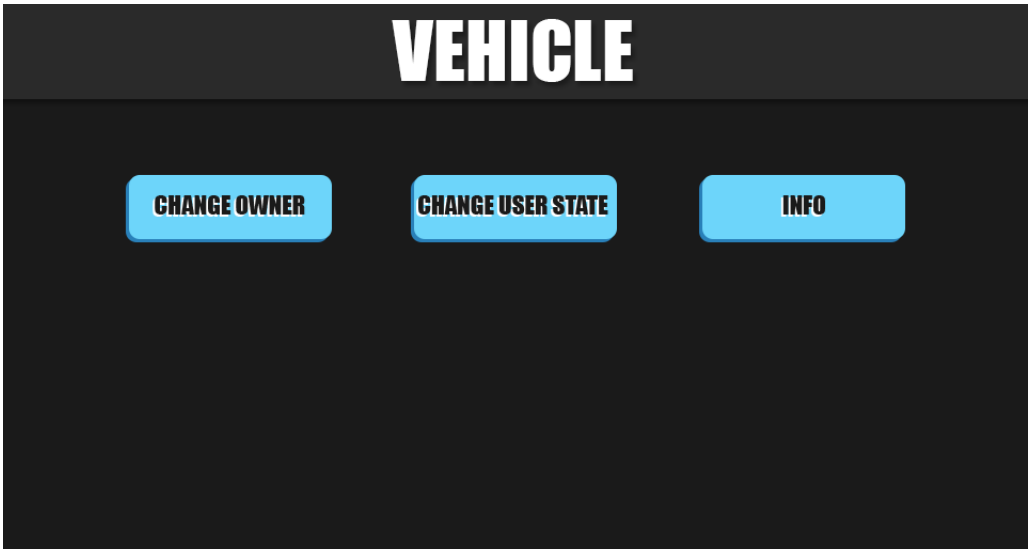
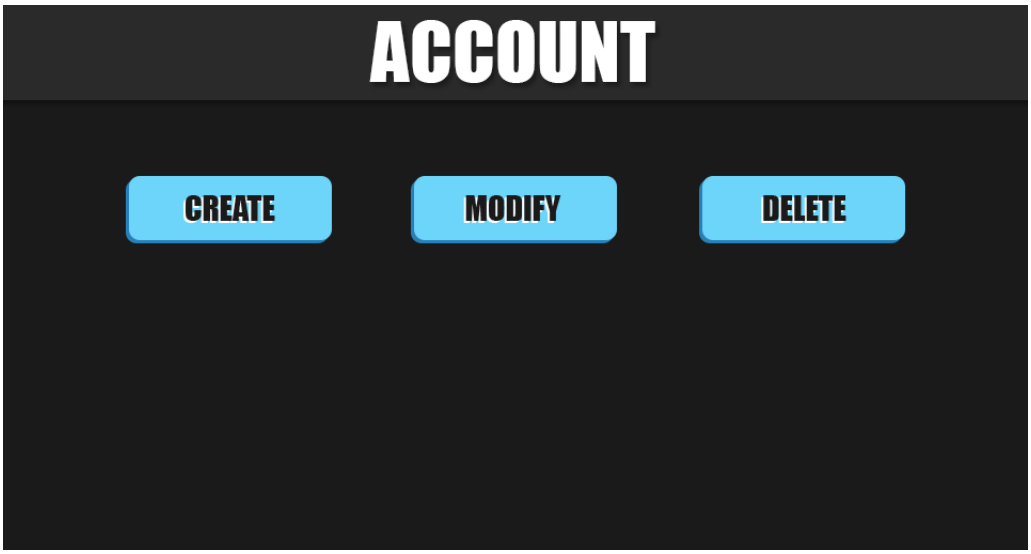
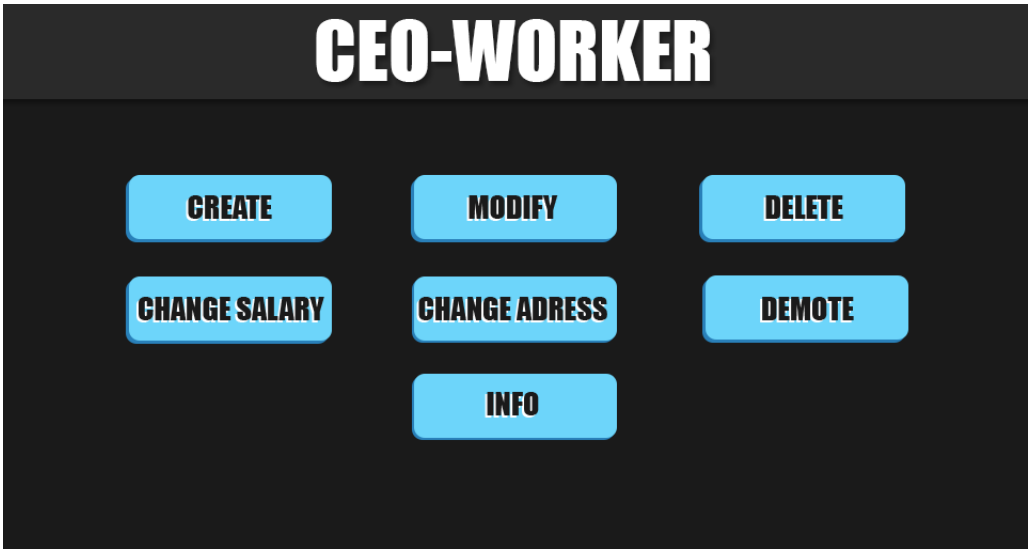


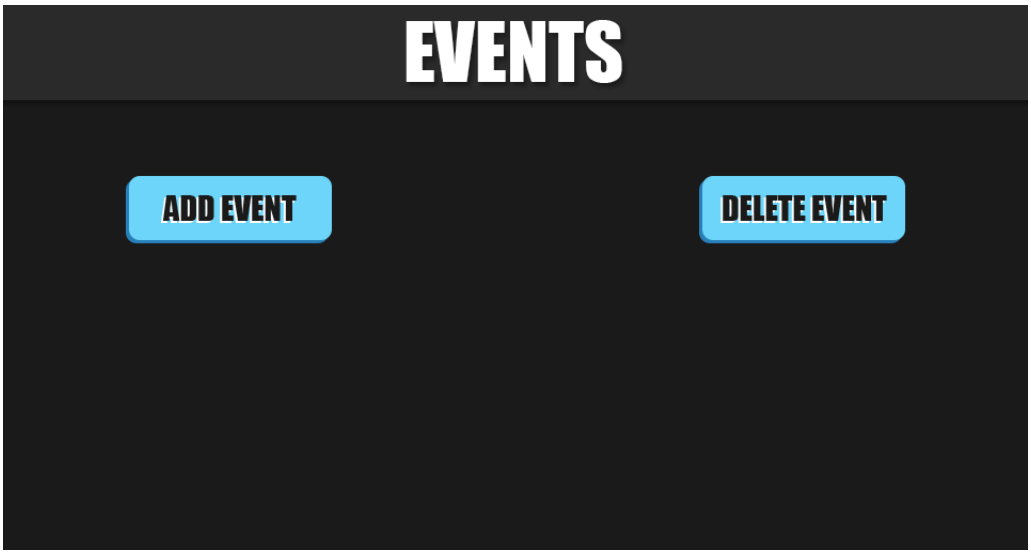
Figure 1: Initial view of the GUI




VEHICLE
CHANGE OWNER
CHANGE USER STATE
INFO


ACCOUNT
CREATE
MODIFY
DELETE


CEO-WORKER
CREATE
MODIFY
DELETE
CHANGE SALARY
CHANGE ADRESS
DEMOTE
INFO


EVENTS
ADD EVENT
DELETE EVENT

3 Selected technologies

List of selected technologies:

- Java
- Tomcat
- MySQL
- HTML
- CSS
- JavaScript

One of the first questions that I had while planning the implementation of the features was what programming language to choose. After some thought, I ended up doing it in **Java**. The reason behind it is that Java is one of the most popular open-source programming languages, widely known and used by software developers and engineers for creating mobile and web applications. A quick summary of it would be that Java is an object-oriented language known for its fast performance, high security and reliability. Present everywhere, it is renowned for its exhaustive testing and consistent delivery that it offers. For me, the main reason that i have chosen Java is because of its basic principle that is applicable "Write Once and Run Anywhere". This means that Java allows developers to write the Java code on one system and use it on other systems/-platforms/devices without much change. Looking at out milestone, Java seemed to suit our interests the best, offering us a solution for implementing features from one subsystem that are linked and related to features from other subsystems. Another technology used in this milestone was the usage of **Tomcat** in the implementation part.

The one that we used was Apache-Tomcat-9.0.41 the latest version available as of writing of this report. Tomcat acted as a web server or servlet container and it made our Java implementation much easier with its open-source and highly flexible built-in customization options. In the near future, the project's technology stack will be composed of more components, the ones that plan on using would be **MySQL**, because of its relational database management system and something for the front end part of this project. I'm thinking about **HTML**, **CSS** and **JavaScript** so the whole part that the user sees can be nicely implemented and user-friendly. Those mentioned for the front-end are technologies that are great ones, providing configuration, optimization and operation of all interface elements, so that's why I think that choosing them will be a great addition to the project. Reasons behind choosing them may be a subjective one also, because I am already familiar with them so working with those technologies would be easier than starting from the beginning the other ones.

4 Code

Note: Code is also in the zipped folder of the submission.
The code I used for implementing the features is the following:
For the Worker class:

```
package subsystem.security.firstwebproject;

public class Worker {
    public String name;
    private String address;
    private int age;
    private int salary;
    private String position;
    private int hours;

    Worker(){

    }

    Worker(String _name, String _address, int _age, int _salary, String
        _position ){
        name = _name;
        address = _address;
        age = _age;
        salary = _salary;
        position = _position;
        hours = 0;
    }

    public void addShift(int workedHours){
        hours = hours + workedHours;
    }

    public void hasBirthday(){
        age++;
    }

        public void changeSalary(int newSalary){
        salary = newSalary;
    }

    public void changePosition(String newPosition){
        position = newPosition;
    }

    public void changeAddress(String newAddress){
        address = newAddress;
    }
}
```

For the CEO:

```
package subsystem.security.firstwebproject;

import com.sun.corba.se.spi.orbutil.threadpool.Work;

import java.util.HashMap;

public class CEO extends Worker {
    public HashMap<String, Worker> employees = new HashMap<String,
        Worker>();

    CEO(){
        super();
    }

    CEO(String _name, String _address, int _age, int _salary, String
        _position ){
        super(_name, _address,_age, _salary, _position );
    }

    public void addWorker(Worker newWorker){
        employees.put(newWorker.name, newWorker);
    }

    public void removeWorker(Worker worker){
        employees.remove(worker.name);
    }

    public void changeSalary(String name, int newSalary){
        Worker tmp = employees.get(name);
        tmp.changeSalary(newSalary);
        employees.remove(name);
        employees.put(tmp.name, tmp);
    }

    public void changePosition(String name, String newPosition){
        Worker tmp = employees.get(name);
        tmp.changePosition(newPosition);
        employees.remove(name);
        employees.put(tmp.name, tmp);
    }

    public void passEmployeesToNewCEO(CEO newCEO){
        newCEO.employees = employees;
    }
}
```

For the Account class:

```
package subsystem.security.firstwebproject;

import java.util.Date;

public class Account {
    public int id;
    public String history;
    public Date opened;

    public void create(int _id, String _history, Date _opened){
        id = _id;
        history = _history;
        opened = _opened;
    }

    public void modify(int _id, String _history, Date _opened){
        id = _id;
        history = _history;
        opened = _opened;
    }


    public void delete(){
        id = -1;
        history = "$deleted";
    }
}
```