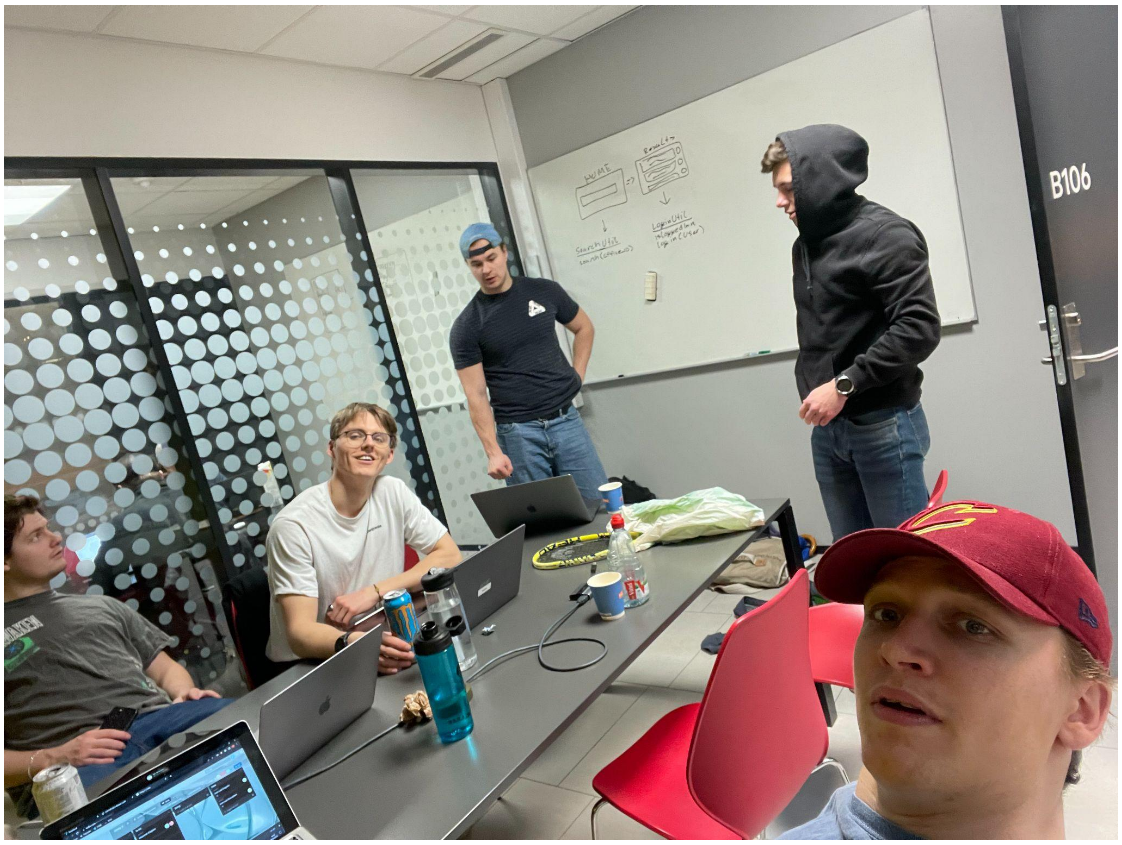
DAT109

Obligatorisk Øvelse 2

Gruppe 100



- Amund Fremming
- Adrian Berget
- Dennis Rizah Osmani
- Harald Nilsen
- Sindre Kjelsrud

Table of Contents

1. Introduction

In this assignment we were assigned to model and program an web-application for a car-rental company that wants to establish in Norway with a rental office in all the big cities, as well as at all the big airports.
The rental offices will rent out and receive cars in return after users have rented it.

This assignment was made to help us practice the following things:

- Practice developing a program from idea to product.
- Practice understanding a requirements specification and creating a model that matches the requirements.
- Practice using object-oriented principles and design patterns to create a solution.

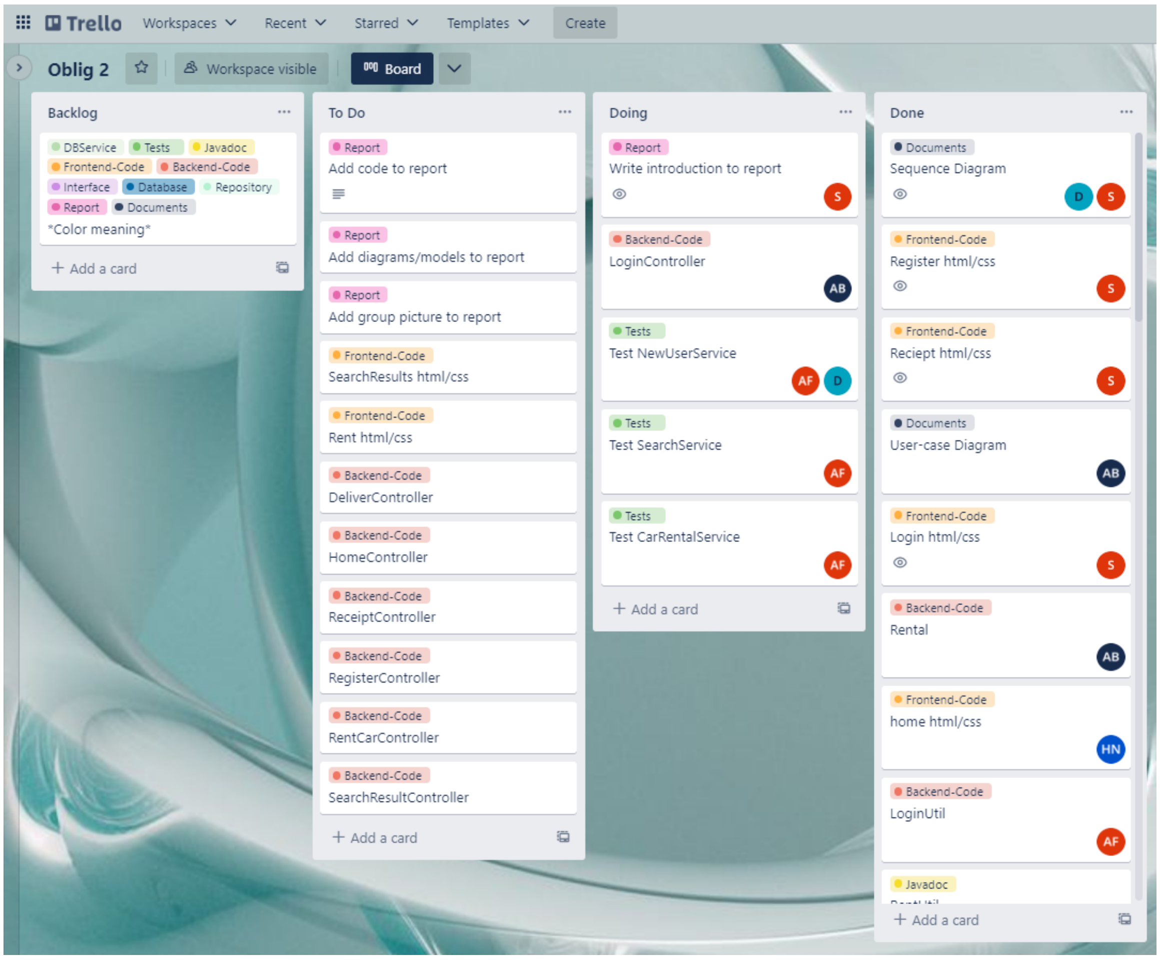
This report will contain our workflow, the different diagrams and models made, and some screenshots of our code and project structure.

2. Workflow

Like the last assignment, we're 5 people working on the same project. Since the usage of an online kanban board on trello.com worked well, we continued with it this time.

This helped us avoid the issue of several people working on the same task, as well as it made it easier for everyone to set up tasks that needed to be done, assign themselves to the tasks they are currently doing, and show which tasks that are finished to the others in the group.

The kanban board also helped us sort out the tasks with different colored tags that would help us see what type of task each task was.



Backlog
- DBService · Tests · Javadoc · Frontend-Code · Backend-Code · Interface · Database · Repository · Report · Documents — *Color meaning*
- + Add a card

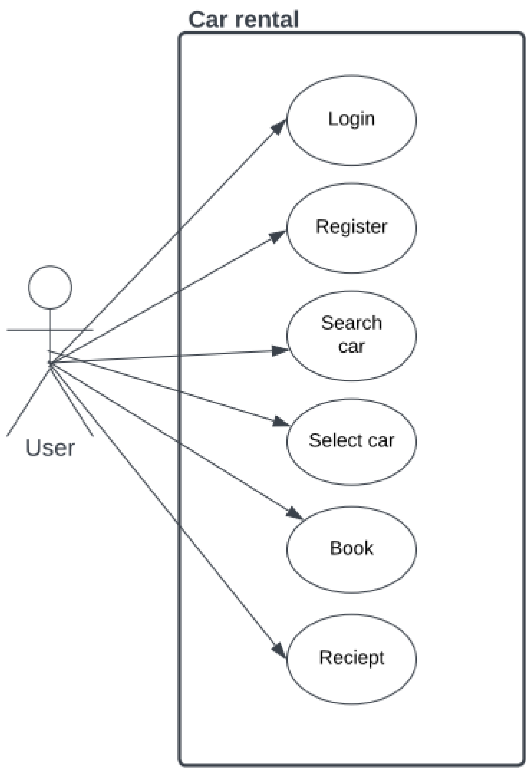
To Do
- Report — Add code to report
- Report — Add diagrams/models to report
- Report — Add group picture to report
- Frontend-Code — SearchResults html/css
- Frontend-Code — Rent html/css
- Backend-Code — DeliverController
- Backend-Code — HomeController
- Backend-Code — ReceiptController
- Backend-Code — RegisterController
- Backend-Code — RentCarController
- Backend-Code — SearchResultController
- + Add a card

Doing
- Report — Write introduction to report — S
- Backend-Code — LoginController — AB
- Tests — Test NewUserService — AF, D
- Tests — Test SearchService — AF
- Tests — Test CarRentalService — AF
- + Add a card

Done
- Documents — Sequence Diagram — D, S
- Frontend-Code — Register html/css — S
- Frontend-Code — Reciept html/css — S
- Documents — User-case Diagram — AB
- Frontend-Code — Login html/css — S
- Backend-Code — Rental — AB
- Frontend-Code — home html/css — HN
- Backend-Code — LoginUtil — AF
- Javadoc
- + Add a card

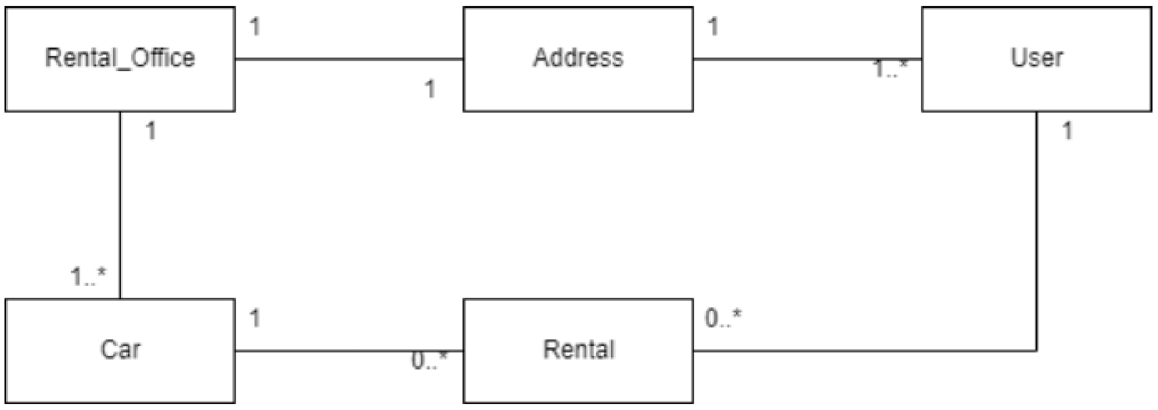
3. Models and Diagrams

3.1. Use-case Diagram + Use-case description

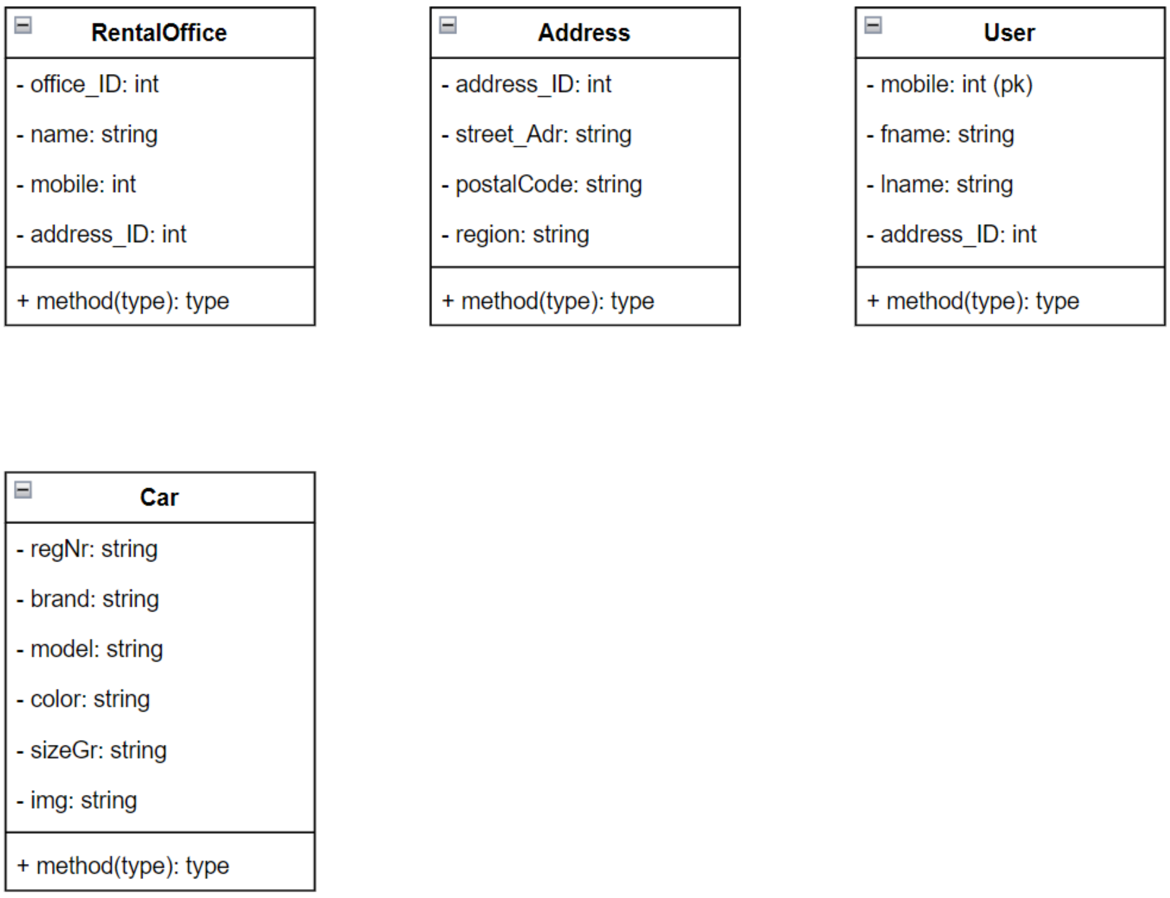


Player action	System response
Login	Check system for stored matching information with login-util, and redirect to a logged in session or the login page with error message if user not exists. Also possible to og to "register" page if user is not already registered.
Register	Provides fields with the required information to create a new user, granting this and storing the information as a user if the input validation is met. Else, redirect to "try again" for the wrongly formatted information entered.
Search car	User can enter its parameters and requirements for the rental of a car, with filters provided by the system to find a best possible car option.
Select car	A list with the available cars given the users parameters. Will show information about the cars. Possible to click on a "book" button to book the car you want from the list.
Book	Provides a booking scheme where you enter time and dates for start end for the rent, and payment information for the payment of the booking. System stores this as a "rental relation" between car, costumer and additional information.
Receipt	The system shows a receipt with the information about the booking.

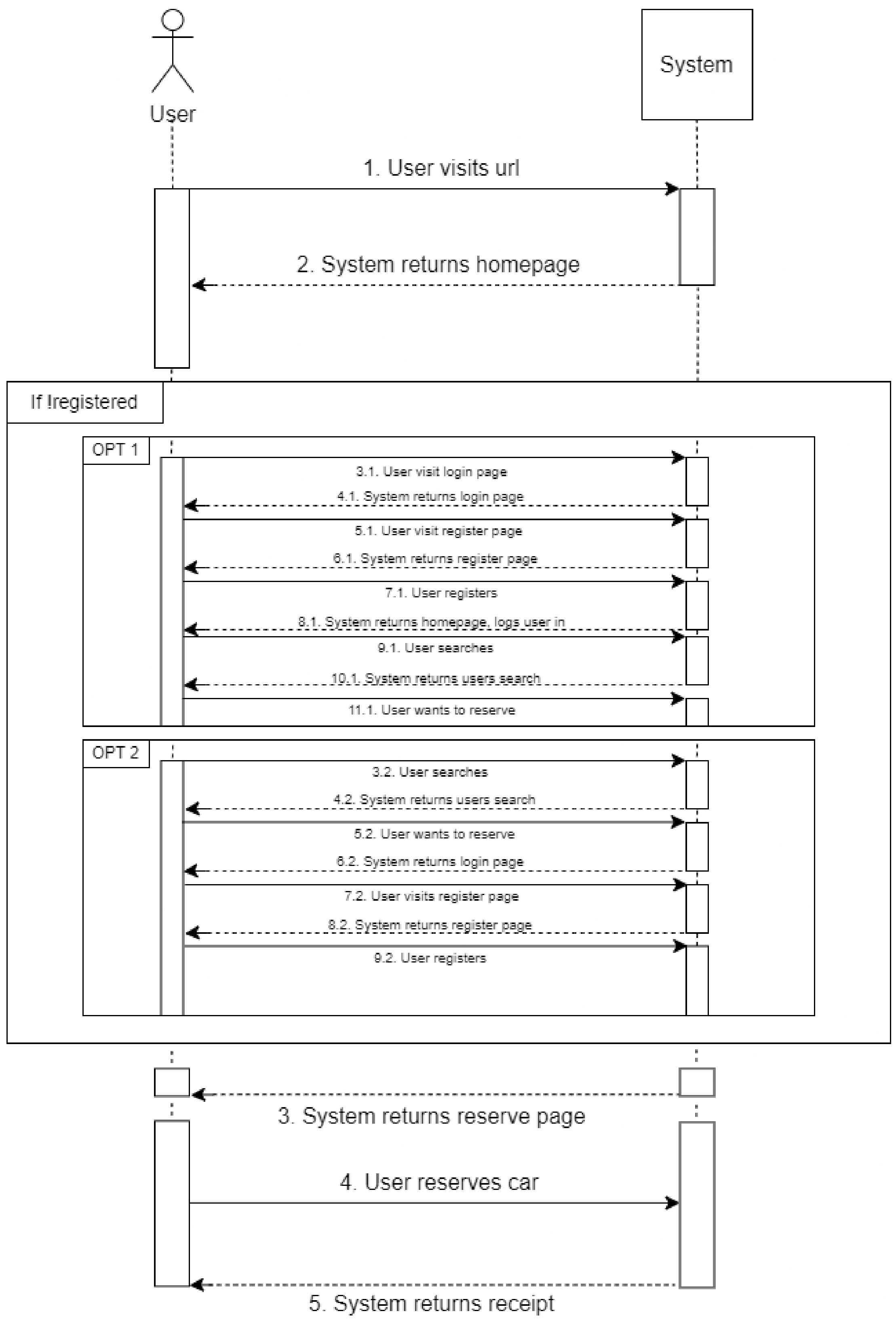
3.2. Domain Model



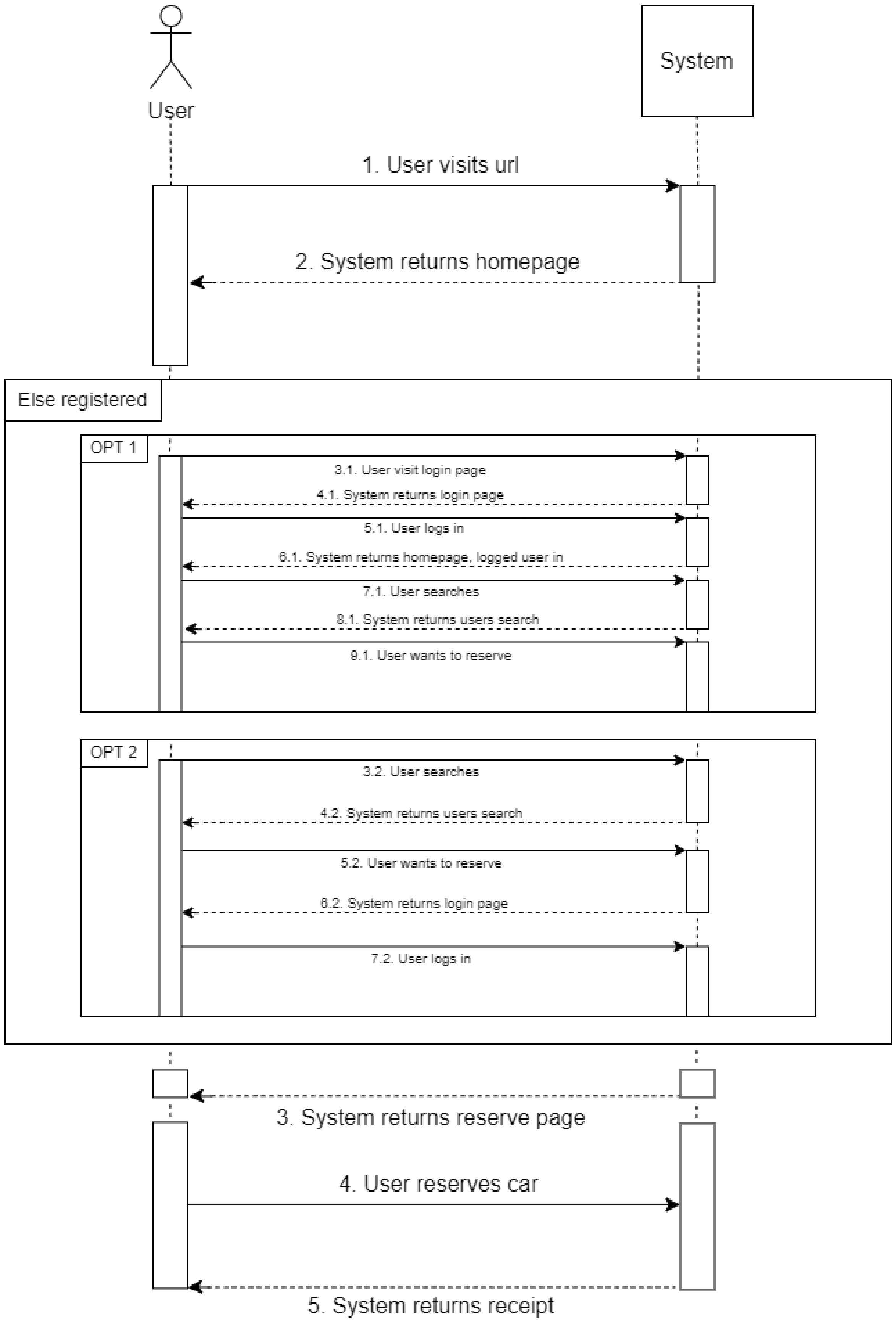
3.3. Class Diagram



3.4. Sequence Diagrams



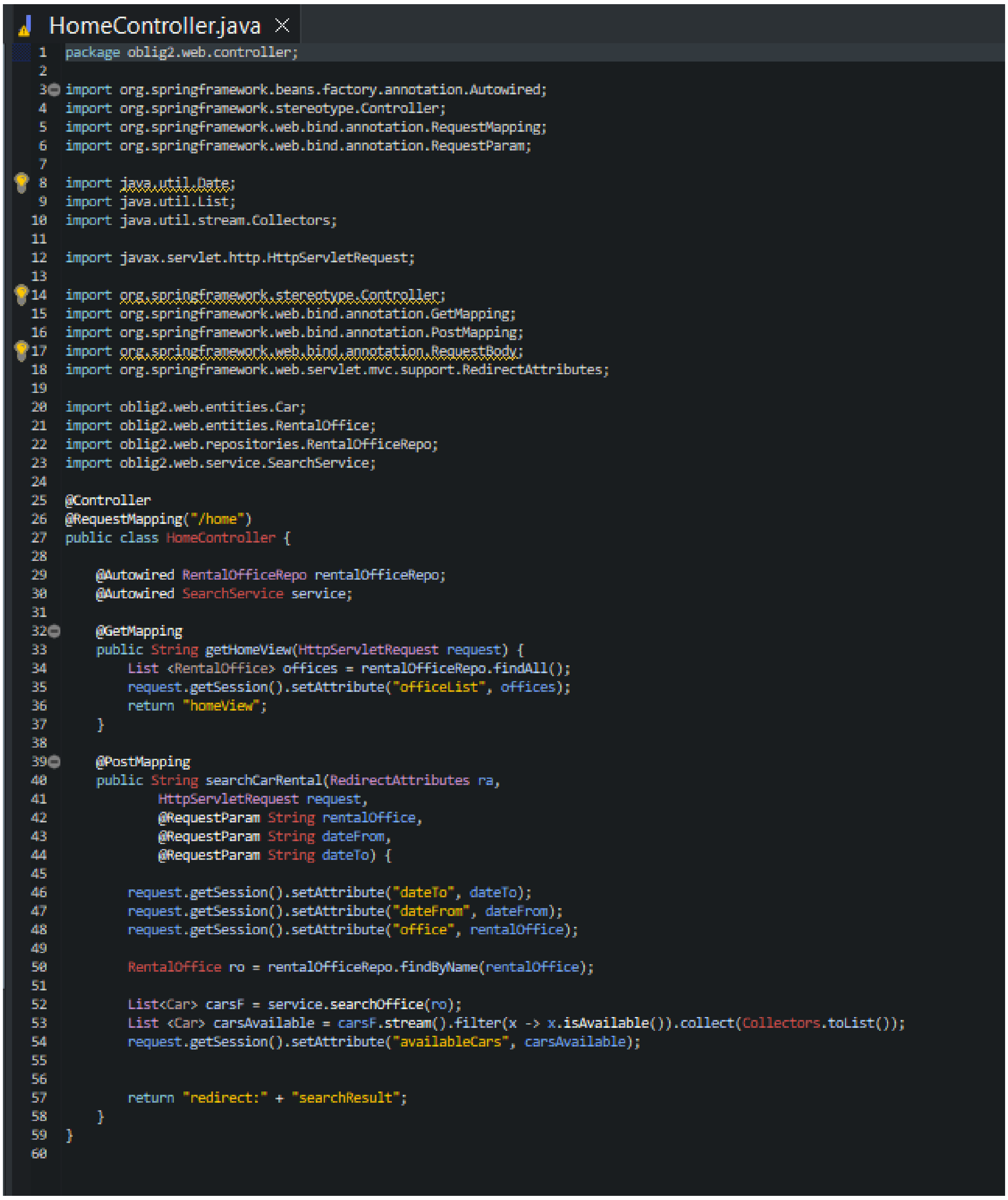
User
System
1. User visits url
2. System returns homepage
If !registered
OPT 1
3.1. User visit login page
4.1. System returns login page
5.1. User visit register page
6.1. System returns register page
7.1. User registers
8.1. System returns homepage, logs user in
9.1. User searches
10.1. System returns users search
11.1. User wants to reserve
OPT 2
3.2. User searches
4.2. System returns users search
5.2. User wants to reserve
6.2. System returns login page
7.2. User visits register page
8.2. System returns register page
9.2. User registers
3. System returns reserve page
4. User reserves car
5. System returns receipt


User
System
1. User visits url
2. System returns homepage
Else registered
OPT 1
3.1. User visit login page
4.1. System returns login page
5.1. User logs in
6.1. System returns homepage, logged user in
7.1. User searches
8.1. System returns users search
9.1. User wants to reserve
OPT 2
3.2. User searches
4.2. System returns users search
5.2. User wants to reserve
6.2. System returns login page
7.2. User logs in
3. System returns reserve page
4. User reserves car
5. System returns receipt

4. Code

4.1. Controllers

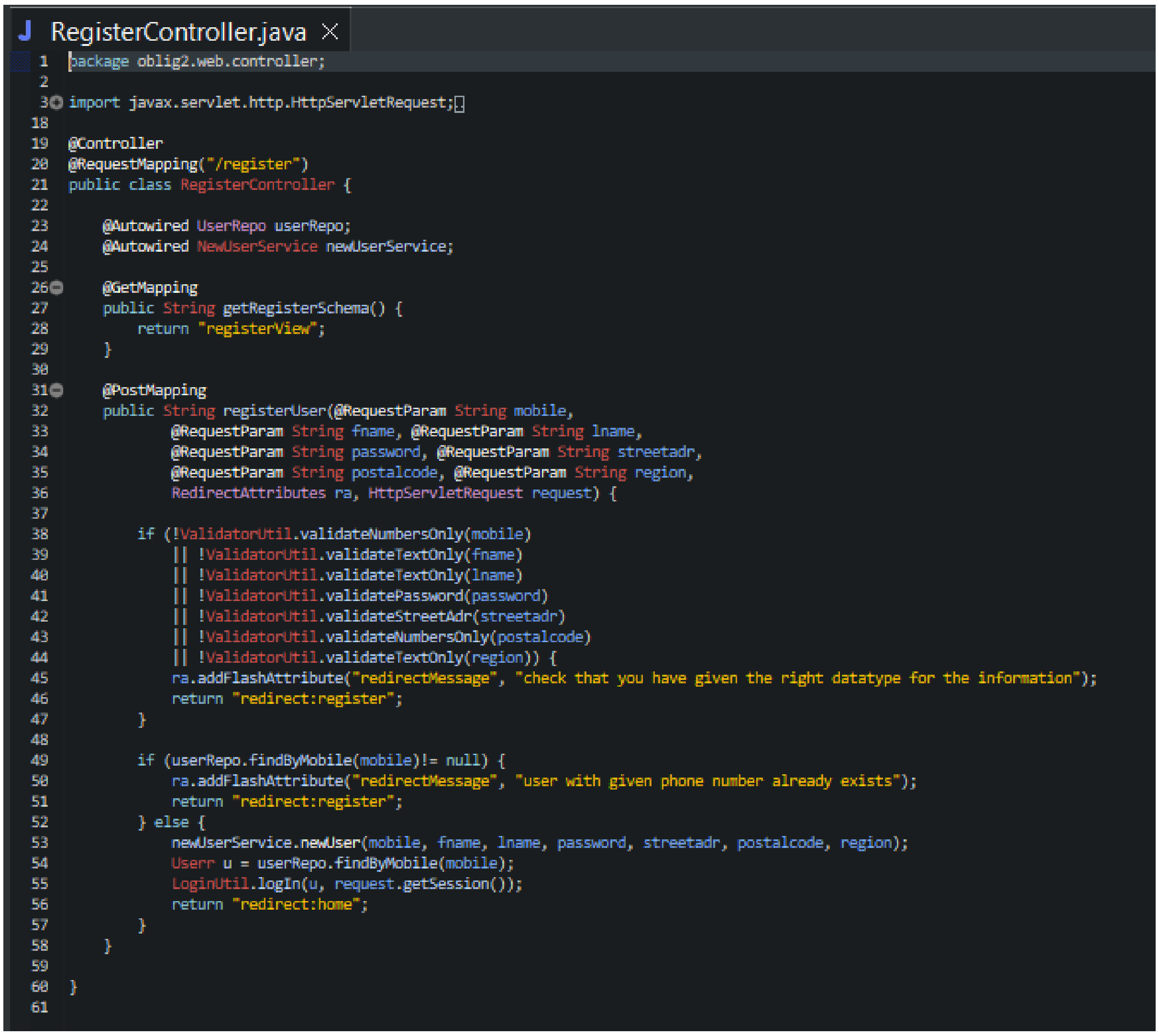
4.1.1. HomeController



HomeController.java

```java
package oblig2.web.controller;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;

import java.util.Date;
import java.util.List;
import java.util.stream.Collectors;

import javax.servlet.http.HttpServletRequest;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import oblig2.web.entities.Car;
import oblig2.web.entities.RentalOffice;
import oblig2.web.repositories.RentalOfficeRepo;
import oblig2.web.service.SearchService;

@Controller
@RequestMapping("/home")
public class HomeController {

    @Autowired RentalOfficeRepo rentalOfficeRepo;
    @Autowired SearchService service;

    @GetMapping
    public String getHomeView(HttpServletRequest request) {
        List <RentalOffice> offices = rentalOfficeRepo.findAll();
        request.getSession().setAttribute("officeList", offices);
        return "homeView";
    }

    @PostMapping
    public String searchCarRental(RedirectAttributes ra,
            HttpServletRequest request,
            @RequestParam String rentalOffice,
            @RequestParam String dateFrom,
            @RequestParam String dateTo) {

        request.getSession().setAttribute("dateTo", dateTo);
        request.getSession().setAttribute("dateFrom", dateFrom);
        request.getSession().setAttribute("office", rentalOffice);

        RentalOffice ro = rentalOfficeRepo.findByName(rentalOffice);

        List<Car> carsF = service.searchOffice(ro);
        List <Car> carsAvailable = carsF.stream().filter(x -> x.isAvailable()).collect(Collectors.toList());
        request.getSession().setAttribute("availableCars", carsAvailable);


        return "redirect:" + "searchResult";
    }
}

```

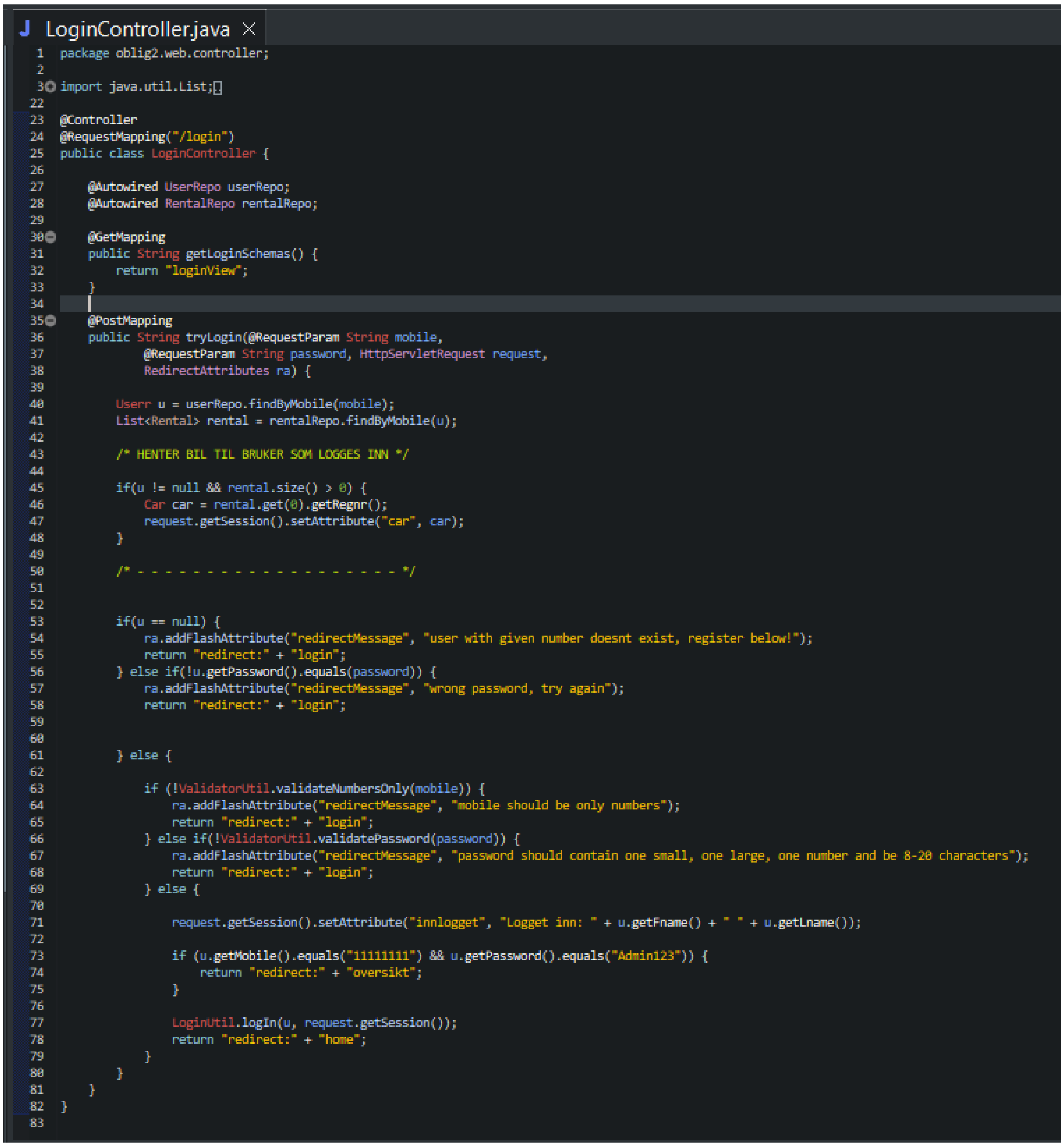
4.1.2. RegisterController



RegisterController.java

```java
package oblig2.web.controller;

import javax.servlet.http.HttpServletRequest;[]

@Controller
@RequestMapping("/register")
public class RegisterController {

    @Autowired UserRepo userRepo;
    @Autowired NewUserService newUserService;

    @GetMapping
    public String getRegisterSchema() {
        return "registerView";
    }

    @PostMapping
    public String registerUser(@RequestParam String mobile,
            @RequestParam String fname, @RequestParam String lname,
            @RequestParam String password, @RequestParam String streetadr,
            @RequestParam String postalcode, @RequestParam String region,
            RedirectAttributes ra, HttpServletRequest request) {

        if (!ValidatorUtil.validateNumbersOnly(mobile)
            || !ValidatorUtil.validateTextOnly(fname)
            || !ValidatorUtil.validateTextOnly(lname)
            || !ValidatorUtil.validatePassword(password)
            || !ValidatorUtil.validateStreetAdr(streetadr)
            || !ValidatorUtil.validateNumbersOnly(postalcode)
            || !ValidatorUtil.validateTextOnly(region)) {
            ra.addFlashAttribute("redirectMessage", "check that you have given the right datatype for the information");
            return "redirect:register";
        }

        if (userRepo.findByMobile(mobile)!= null) {
            ra.addFlashAttribute("redirectMessage", "user with given phone number already exists");
            return "redirect:register";
        } else {
            newUserService.newUser(mobile, fname, lname, password, streetadr, postalcode, region);
            Userr u = userRepo.findByMobile(mobile);
            LoginUtil.logIn(u, request.getSession());
            return "redirect:home";
        }
    }

}
```

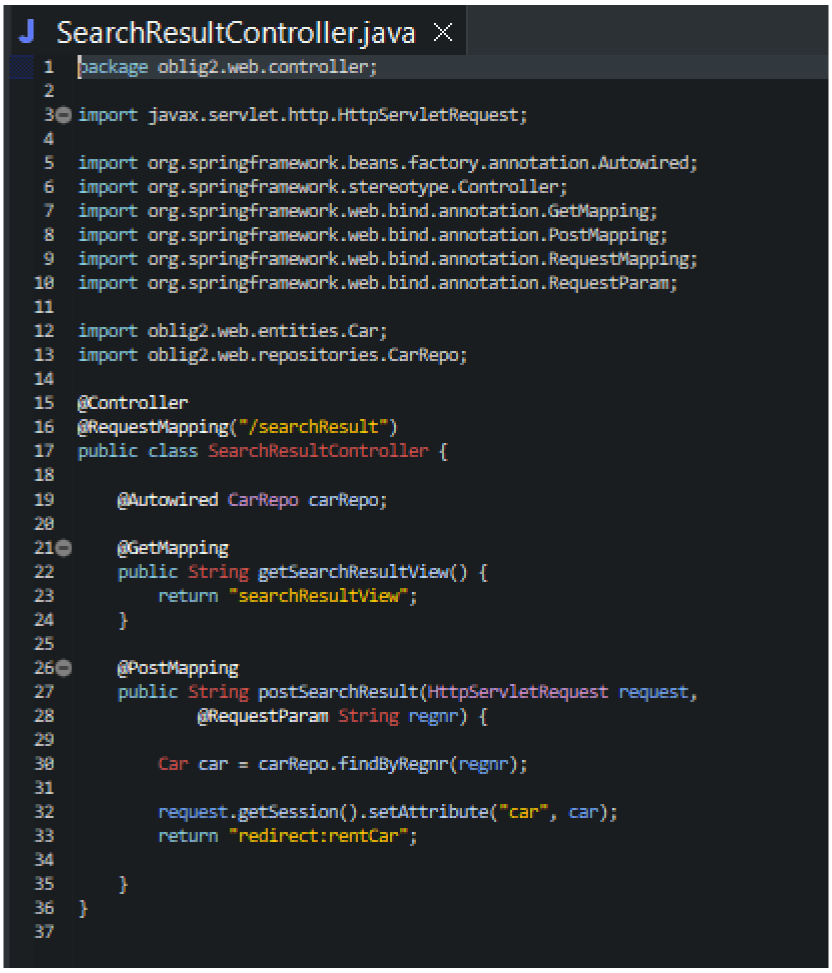
4.1.3. LoginController



LoginController.java

```java
package oblig2.web.controller;

import java.util.List;

@Controller
@RequestMapping("/login")
public class LoginController {

    @Autowired UserRepo userRepo;
    @Autowired RentalRepo rentalRepo;

    @GetMapping
    public String getLoginSchemas() {
        return "loginView";
    }

    @PostMapping
    public String tryLogin(@RequestParam String mobile,
            @RequestParam String password, HttpServletRequest request,
            RedirectAttributes ra) {

        Userr u = userRepo.findByMobile(mobile);
        List<Rental> rental = rentalRepo.findByMobile(u);

        /* HENTER BIL TIL BRUKER SOM LOGGES INN */

        if(u != null && rental.size() > 0) {
            Car car = rental.get(0).getRegnr();
            request.getSession().setAttribute("car", car);
        }

        /* - - - - - - - - - - - - - - - - - - - */


        if(u == null) {
            ra.addFlashAttribute("redirectMessage", "user with given number doesnt exist, register below!");
            return "redirect:" + "login";
        } else if(!u.getPassword().equals(password)) {
            ra.addFlashAttribute("redirectMessage", "wrong password, try again");
            return "redirect:" + "login";


        } else {

            if (!ValidatorUtil.validateNumbersOnly(mobile)) {
                ra.addFlashAttribute("redirectMessage", "mobile should be only numbers");
                return "redirect:" + "login";
            } else if(!ValidatorUtil.validatePassword(password)) {
                ra.addFlashAttribute("redirectMessage", "password should contain one small, one large, one number and be 8-20 characters");
                return "redirect:" + "login";
            } else {

                request.getSession().setAttribute("innlogget", "Logget inn: " + u.getFname() + " " + u.getLname());

                if (u.getMobile().equals("11111111") && u.getPassword().equals("Admin123")) {
                    return "redirect:" + "oversikt";
                }

                LoginUtil.logIn(u, request.getSession());
                return "redirect:" + "home";
            }
        }
    }
}
```

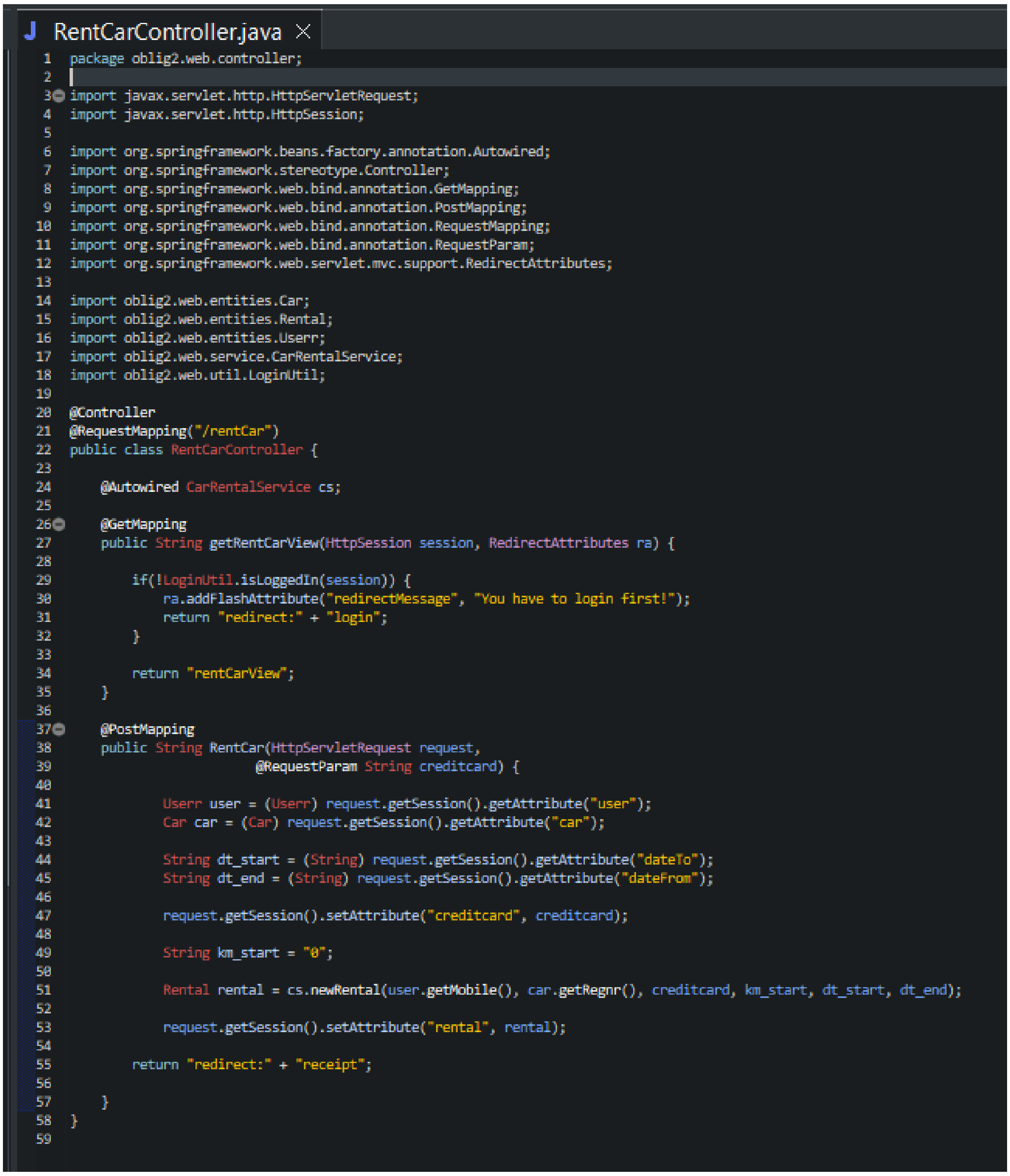
4.1.4. SearchResultController



SearchResultController.java

```java
package oblig2.web.controller;

import javax.servlet.http.HttpServletRequest;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;

import oblig2.web.entities.Car;
import oblig2.web.repositories.CarRepo;

@Controller
@RequestMapping("/searchResult")
public class SearchResultController {

    @Autowired CarRepo carRepo;

    @GetMapping
    public String getSearchResultView() {
        return "searchResultView";
    }

    @PostMapping
    public String postSearchResult(HttpServletRequest request,
            @RequestParam String regnr) {

        Car car = carRepo.findByRegnr(regnr);

        request.getSession().setAttribute("car", car);
        return "redirect:rentCar";

    }
}
```

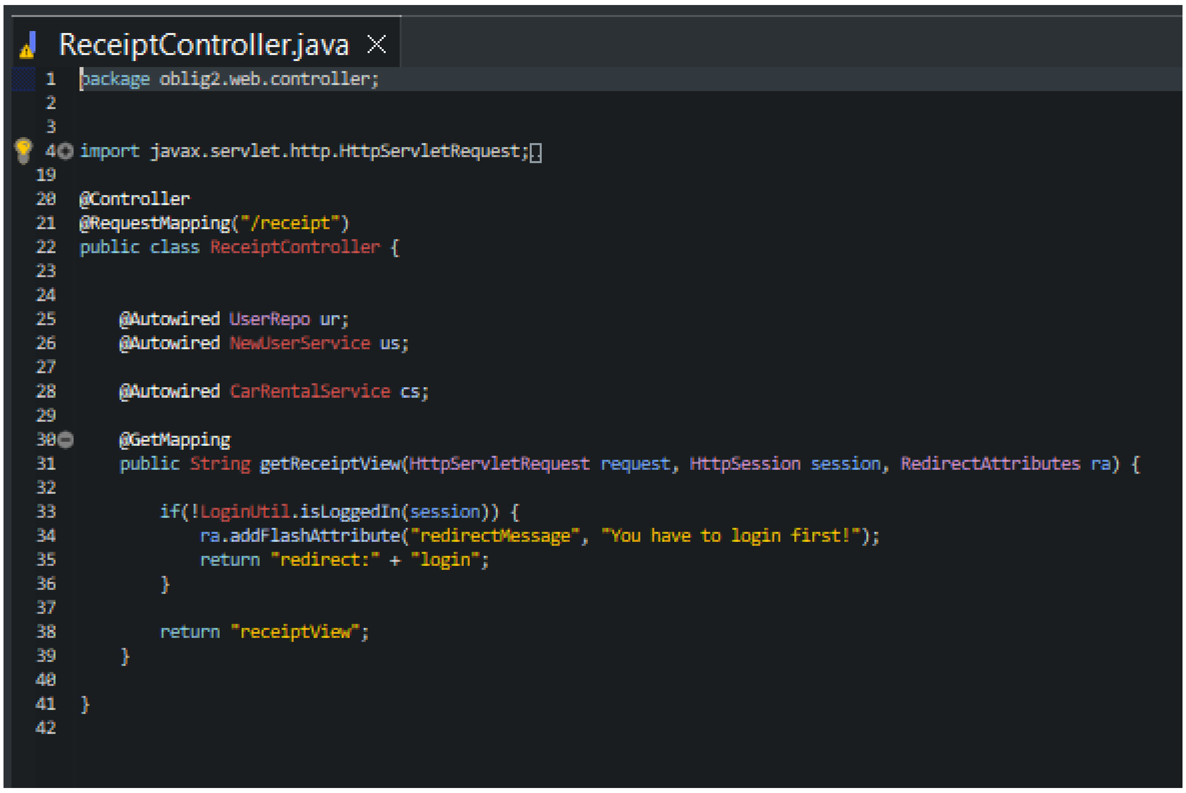
4.1.5. RentCarController



RentCarController.java

```java
package oblig2.web.controller;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpSession;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import oblig2.web.entities.Car;
import oblig2.web.entities.Rental;
import oblig2.web.entities.Userr;
import oblig2.web.service.CarRentalService;
import oblig2.web.util.LoginUtil;

@Controller
@RequestMapping("/rentCar")
public class RentCarController {

    @Autowired CarRentalService cs;

    @GetMapping
    public String getRentCarView(HttpSession session, RedirectAttributes ra) {

        if(!LoginUtil.isLoggedIn(session)) {
            ra.addFlashAttribute("redirectMessage", "You have to login first!");
            return "redirect:" + "login";
        }

        return "rentCarView";
    }

    @PostMapping
    public String RentCar(HttpServletRequest request,
                          @RequestParam String creditcard) {

            Userr user = (Userr) request.getSession().getAttribute("user");
            Car car = (Car) request.getSession().getAttribute("car");

            String dt_start = (String) request.getSession().getAttribute("dateTo");
            String dt_end = (String) request.getSession().getAttribute("dateFrom");

            request.getSession().setAttribute("creditcard", creditcard);

            String km_start = "0";

            Rental rental = cs.newRental(user.getMobile(), car.getRegnr(), creditcard, km_start, dt_start, dt_end);

            request.getSession().setAttribute("rental", rental);

        return "redirect:" + "receipt";

    }
}
```

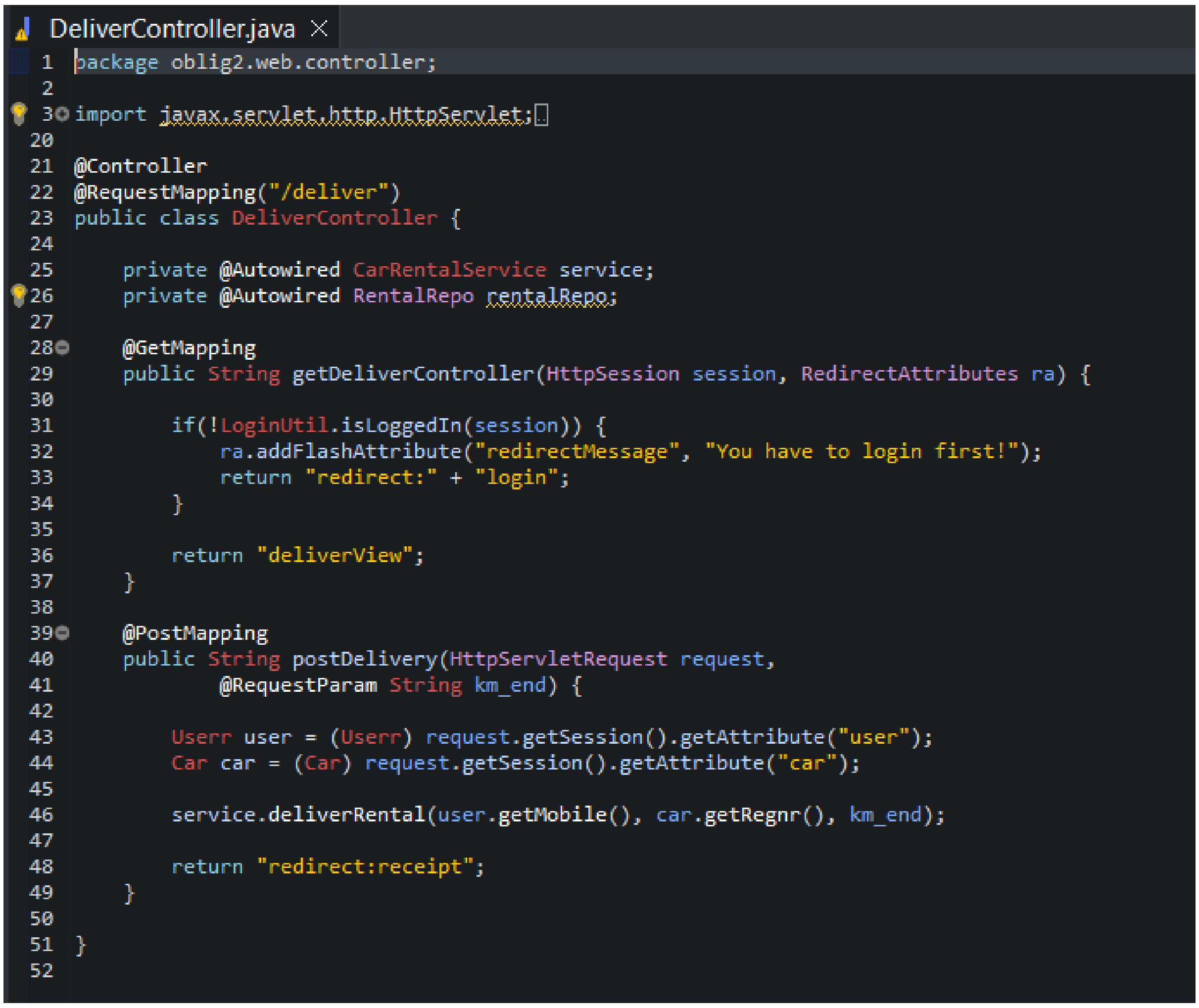
4.1.6. ReceiptController



ReceiptController.java

```java
package oblig2.web.controller;


import javax.servlet.http.HttpServletRequest;

@Controller
@RequestMapping("/receipt")
public class ReceiptController {


    @Autowired UserRepo ur;
    @Autowired NewUserService us;

    @Autowired CarRentalService cs;

    @GetMapping
    public String getReceiptView(HttpServletRequest request, HttpSession session, RedirectAttributes ra) {

        if(!LoginUtil.isLoggedIn(session)) {
            ra.addFlashAttribute("redirectMessage", "You have to login first!");
            return "redirect:" + "login";
        }

        return "receiptView";
    }

}
```

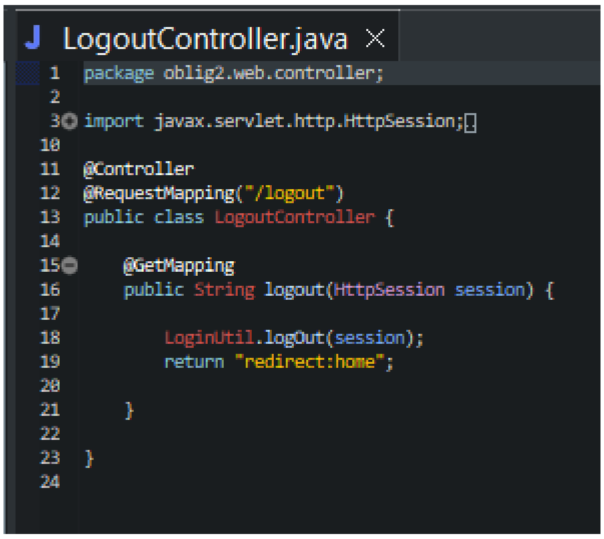
4.1.7. DeliverController



DeliverController.java

```java
package oblig2.web.controller;

import javax.servlet.http.HttpServlet;

@Controller
@RequestMapping("/deliver")
public class DeliverController {

    private @Autowired CarRentalService service;
    private @Autowired RentalRepo rentalRepo;

    @GetMapping
    public String getDeliverController(HttpSession session, RedirectAttributes ra) {

        if(!LoginUtil.isLoggedIn(session)) {
            ra.addFlashAttribute("redirectMessage", "You have to login first!");
            return "redirect:" + "login";
        }

        return "deliverView";
    }

    @PostMapping
    public String postDelivery(HttpServletRequest request,
            @RequestParam String km_end) {

        Userr user = (Userr) request.getSession().getAttribute("user");
        Car car = (Car) request.getSession().getAttribute("car");

        service.deliverRental(user.getMobile(), car.getRegnr(), km_end);

        return "redirect:receipt";
    }

}
```

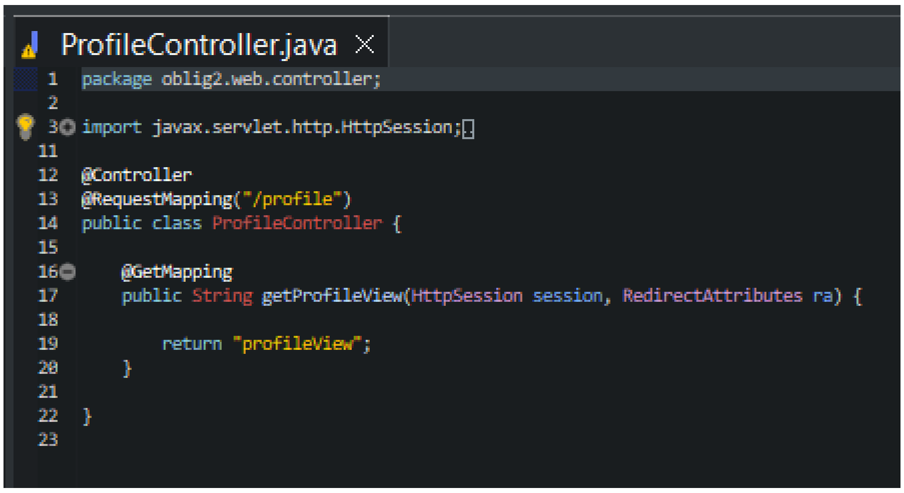
4.1.8. LogoutController



LogoutController.java

```java
package oblig2.web.controller;

import javax.servlet.http.HttpSession;

@Controller
@RequestMapping("/logout")
public class LogoutController {

    @GetMapping
    public String logout(HttpSession session) {

        LoginUtil.logOut(session);
        return "redirect:home";

    }

}
```

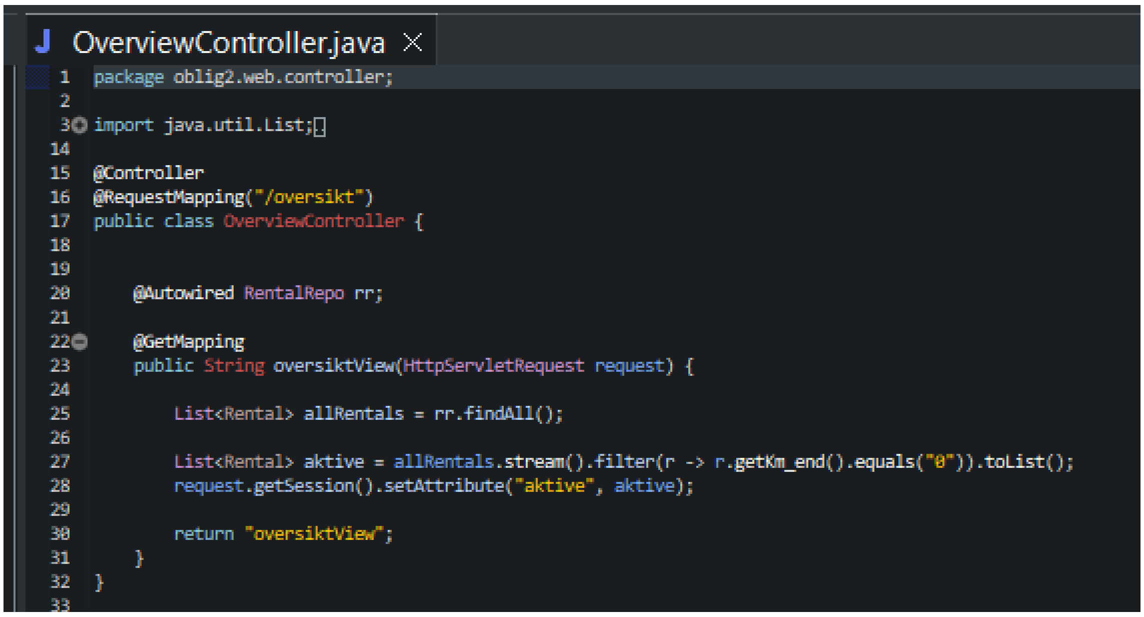
4.1.9. ProfileController



ProfileController.java

```java
package oblig2.web.controller;

import javax.servlet.http.HttpSession;[]

@Controller
@RequestMapping("/profile")
public class ProfileController {

    @GetMapping
    public String getProfileView(HttpSession session, RedirectAttributes ra) {

        return "profileView";
    }

}

```

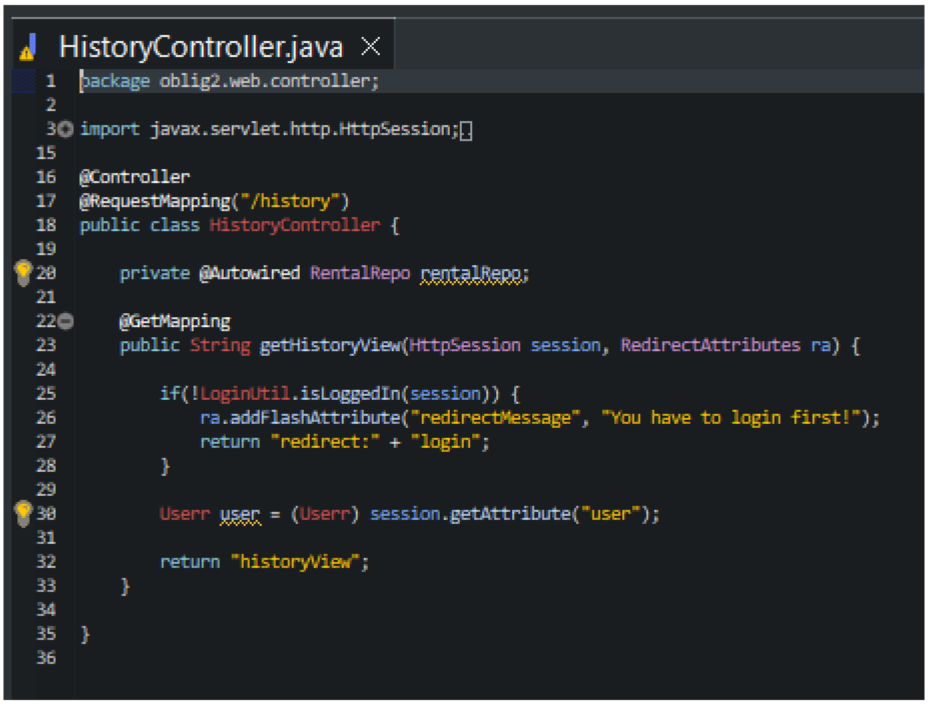
4.1.10. OverviewController



OverviewController.java

```java
package oblig2.web.controller;

import java.util.List;[]

@Controller
@RequestMapping("/oversikt")
public class OverviewController {


    @Autowired RentalRepo rr;

    @GetMapping
    public String oversiktView(HttpServletRequest request) {

        List<Rental> allRentals = rr.findAll();

        List<Rental> aktive = allRentals.stream().filter(r -> r.getKm_end().equals("0")).toList();
        request.getSession().setAttribute("aktive", aktive);

        return "oversiktView";
    }
}

```

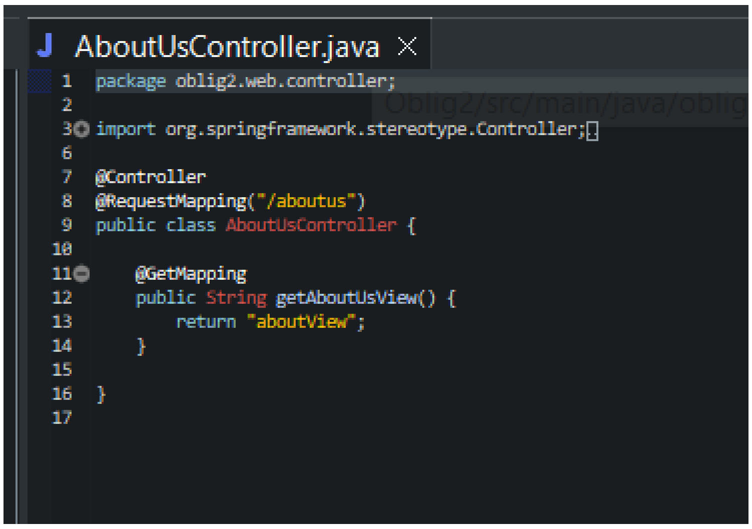
4.1.11. HistoryController



HistoryController.java

```java
package oblig2.web.controller;

import javax.servlet.http.HttpSession;[]

@Controller
@RequestMapping("/history")
public class HistoryController {

    private @Autowired RentalRepo rentalRepo;

    @GetMapping
    public String getHistoryView(HttpSession session, RedirectAttributes ra) {

        if(!LoginUtil.isLoggedIn(session)) {
            ra.addFlashAttribute("redirectMessage", "You have to login first!");
            return "redirect:" + "login";
        }

        Userr user = (Userr) session.getAttribute("user");

        return "historyView";
    }

}
```

4.1.11. AboutUsController

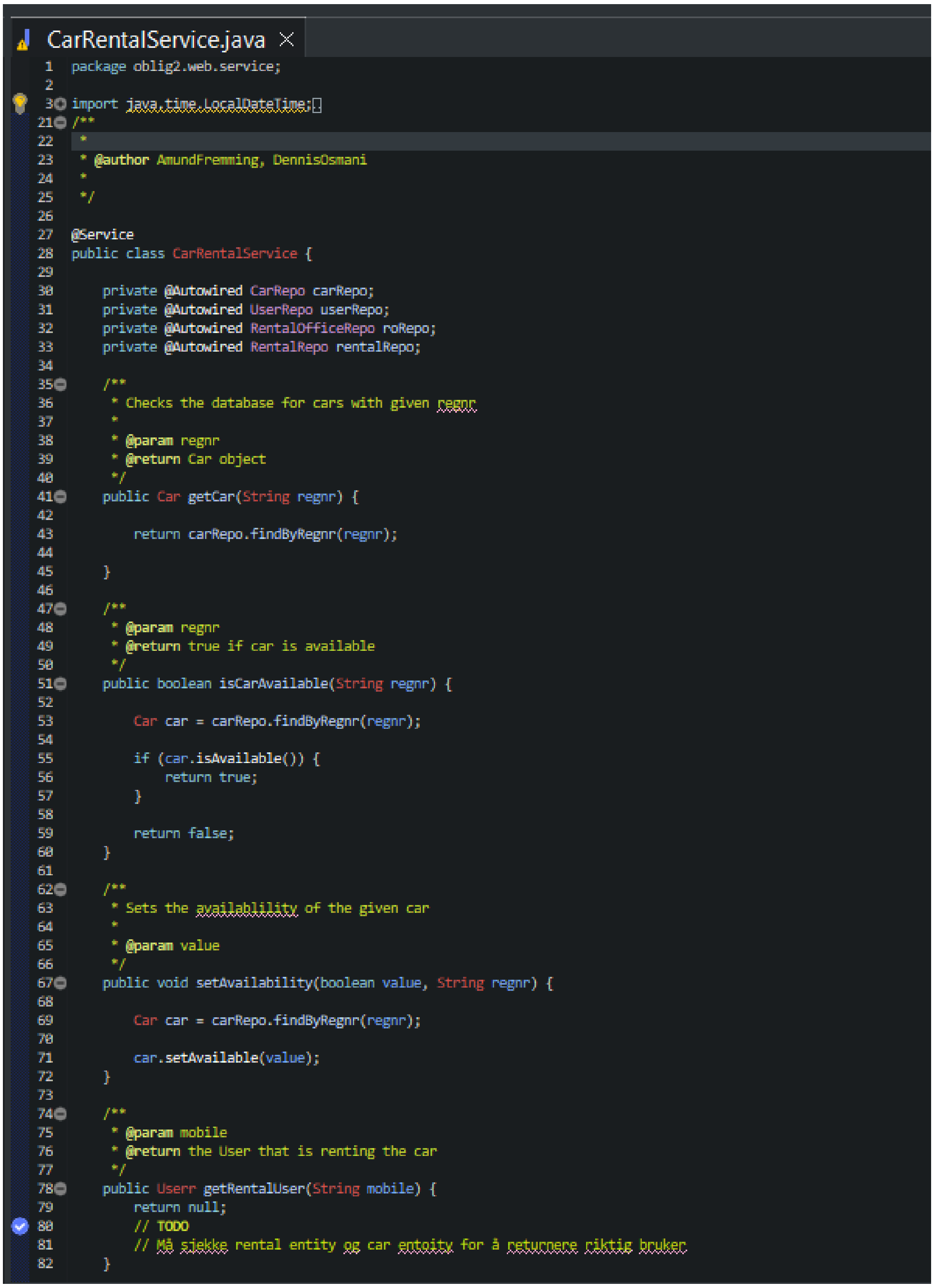


AboutUsController.java

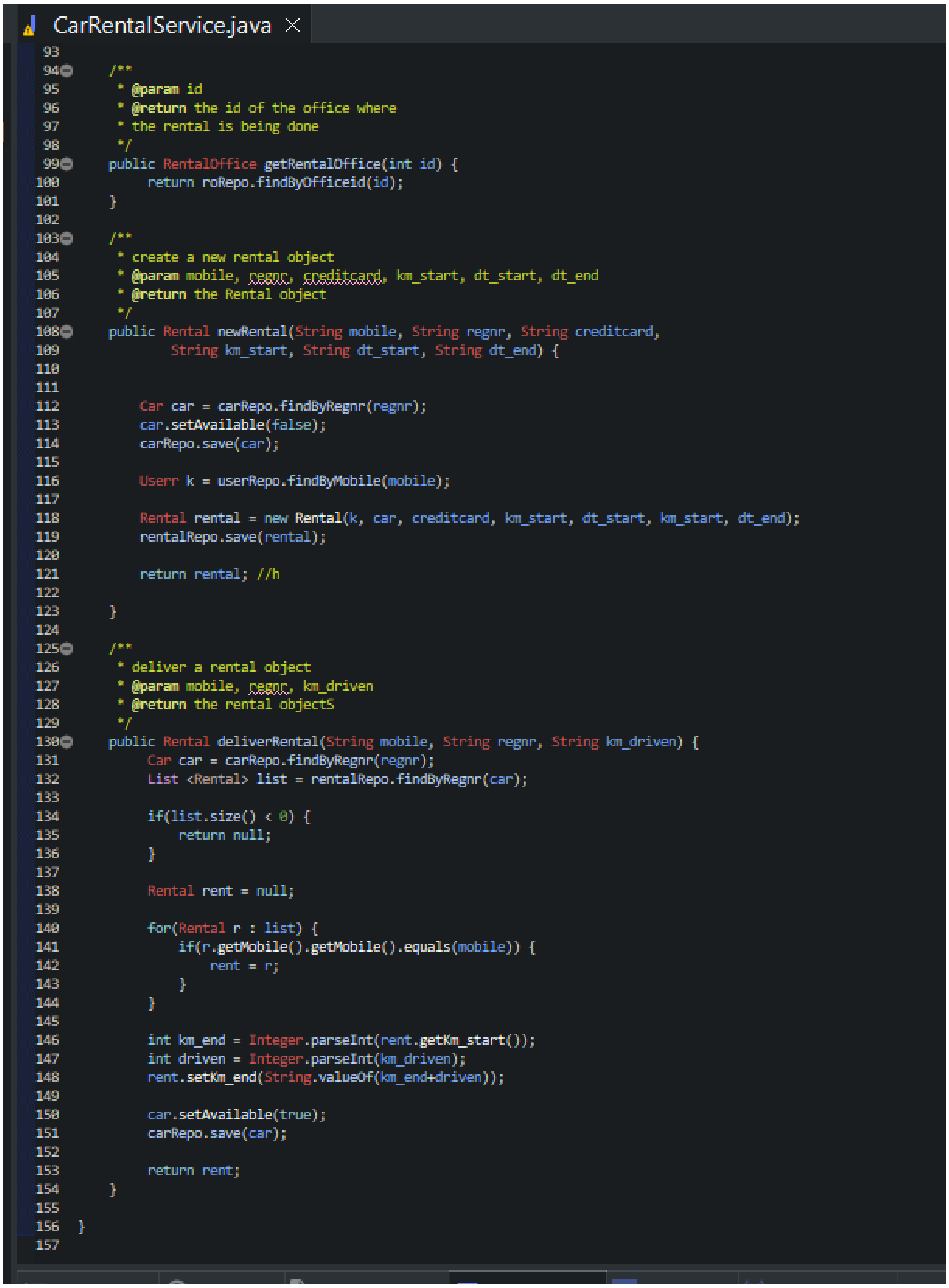
```java
package oblig2.web.controller;

import org.springframework.stereotype.Controller;[]

@Controller
@RequestMapping("/aboutus")
public class AboutUsController {

    @GetMapping
    public String getAboutUsView() {
        return "aboutView";
    }

}
```

4.2. Services

4.2.1. CarRentalService

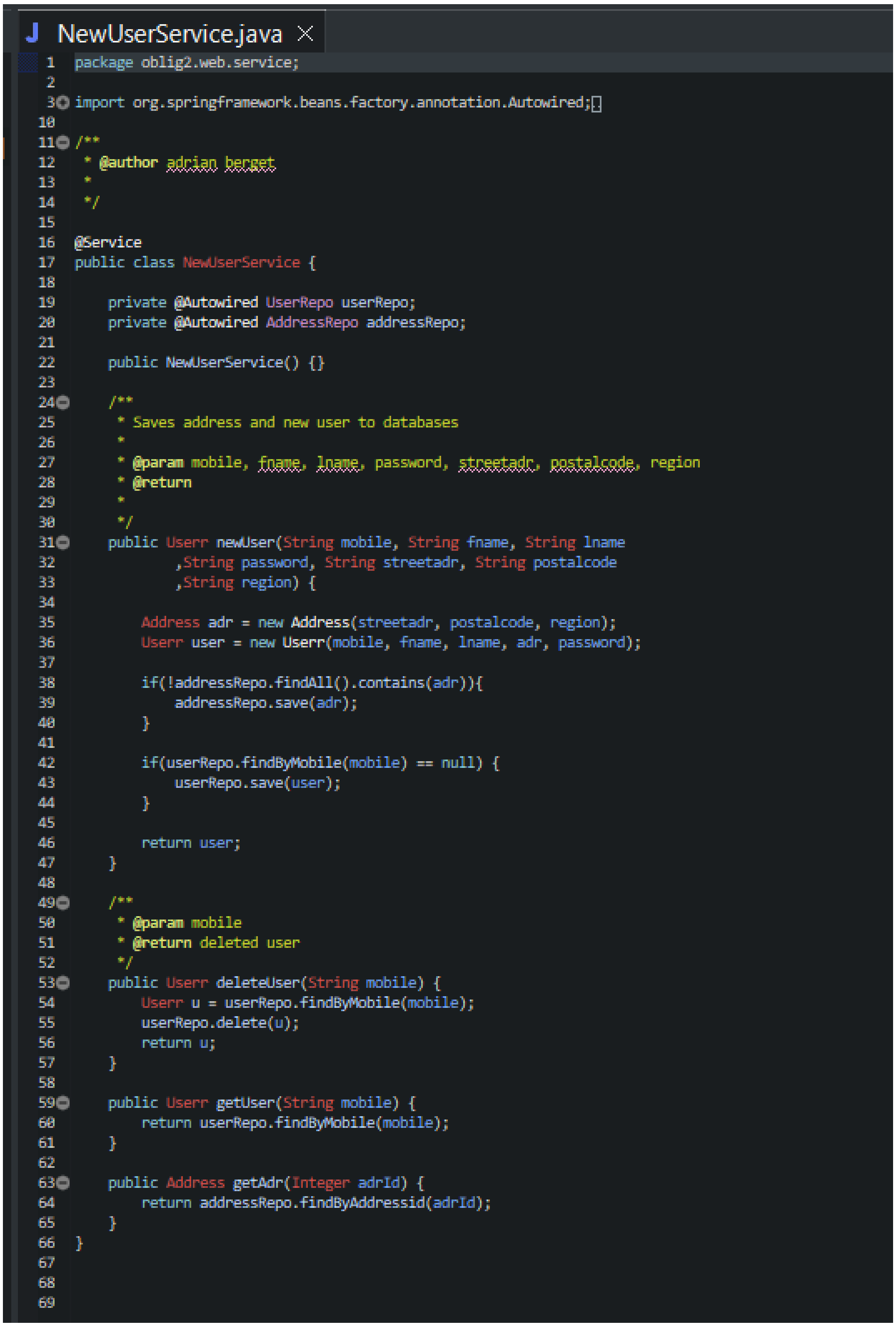


```java
package oblig2.web.service;

import java.time.LocalDateTime;
/**
 *
 * @author AmundFremming, DennisOsmani
 *
 */

@Service
public class CarRentalService {

    private @Autowired CarRepo carRepo;
    private @Autowired UserRepo userRepo;
    private @Autowired RentalOfficeRepo roRepo;
    private @Autowired RentalRepo rentalRepo;

    /**
     * Checks the database for cars with given regnr
     *
     * @param regnr
     * @return Car object
     */
    public Car getCar(String regnr) {

        return carRepo.findByRegnr(regnr);

    }

    /**
     * @param regnr
     * @return true if car is available
     */
    public boolean isCarAvailable(String regnr) {

        Car car = carRepo.findByRegnr(regnr);

        if (car.isAvailable()) {
            return true;
        }

        return false;
    }

    /**
     * Sets the availablility of the given car
     *
     * @param value
     */
    public void setAvailability(boolean value, String regnr) {

        Car car = carRepo.findByRegnr(regnr);

        car.setAvailable(value);
    }

    /**
     * @param mobile
     * @return the User that is renting the car
     */
    public Userr getRentalUser(String mobile) {
        return null;
        // TODO
        // Må sjekke rental entity og car entoity for å returnere riktig bruker
    }
```


CarRentalService.java

```java

    /**
     * @param id
     * @return the id of the office where
     * the rental is being done
     */
    public RentalOffice getRentalOffice(int id) {
         return roRepo.findByOfficeid(id);
    }

    /**
     * create a new rental object
     * @param mobile, regnr, creditcard, km_start, dt_start, dt_end
     * @return the Rental object
     */
    public Rental newRental(String mobile, String regnr, String creditcard,
            String km_start, String dt_start, String dt_end) {


        Car car = carRepo.findByRegnr(regnr);
        car.setAvailable(false);
        carRepo.save(car);

        Userr k = userRepo.findByMobile(mobile);

        Rental rental = new Rental(k, car, creditcard, km_start, dt_start, km_start, dt_end);
        rentalRepo.save(rental);

        return rental; //h

    }

    /**
     * deliver a rental object
     * @param mobile, regnr, km_driven
     * @return the rental objectS
     */
    public Rental deliverRental(String mobile, String regnr, String km_driven) {
         Car car = carRepo.findByRegnr(regnr);
         List <Rental> list = rentalRepo.findByRegnr(car);

         if(list.size() < 0) {
             return null;
         }

         Rental rent = null;

         for(Rental r : list) {
             if(r.getMobile().getMobile().equals(mobile)) {
                 rent = r;
             }
         }

         int km_end = Integer.parseInt(rent.getKm_start());
         int driven = Integer.parseInt(km_driven);
         rent.setKm_end(String.valueOf(km_end+driven));

         car.setAvailable(true);
         carRepo.save(car);

         return rent;
    }

}

```

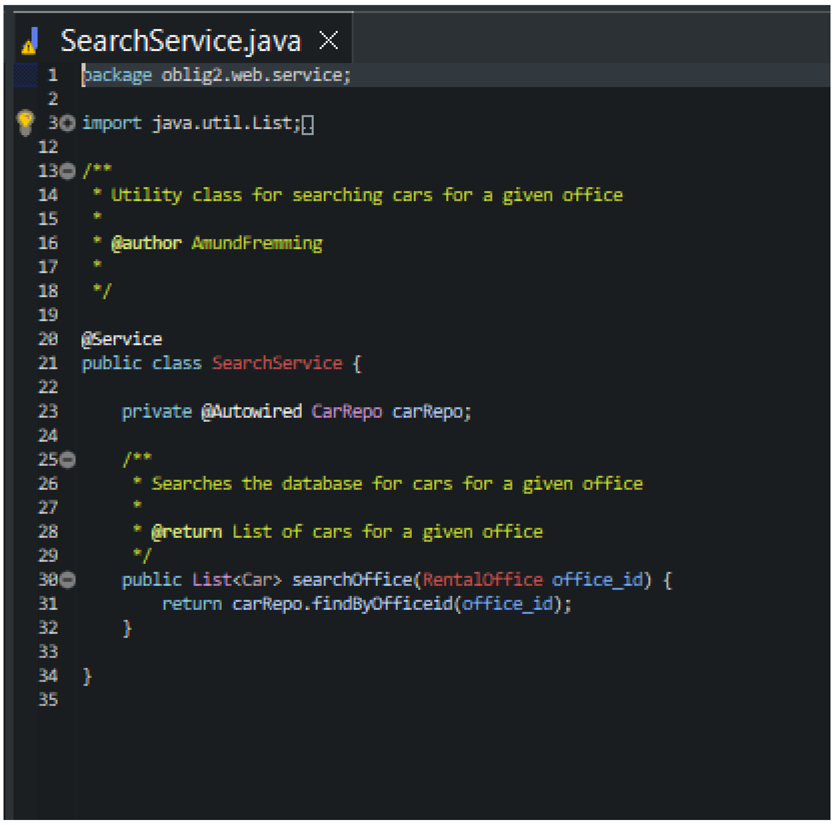
4.2.2. NewUserService



NewUserService.java

```java
package oblig2.web.service;

import org.springframework.beans.factory.annotation.Autowired;[]

/**
 * @author adrian berget
 *
 */

@Service
public class NewUserService {

    private @Autowired UserRepo userRepo;
    private @Autowired AddressRepo addressRepo;

    public NewUserService() {}

    /**
     * Saves address and new user to databases
     *
     * @param mobile, fname, lname, password, streetadr, postalcode, region
     * @return
     *
     */
    public Userr newUser(String mobile, String fname, String lname
            ,String password, String streetadr, String postalcode
            ,String region) {

        Address adr = new Address(streetadr, postalcode, region);
        Userr user = new Userr(mobile, fname, lname, adr, password);

        if(!addressRepo.findAll().contains(adr)){
            addressRepo.save(adr);
        }

        if(userRepo.findByMobile(mobile) == null) {
            userRepo.save(user);
        }

        return user;
    }

    /**
     * @param mobile
     * @return deleted user
     */
    public Userr deleteUser(String mobile) {
        Userr u = userRepo.findByMobile(mobile);
        userRepo.delete(u);
        return u;
    }

    public Userr getUser(String mobile) {
        return userRepo.findByMobile(mobile);
    }

    public Address getAdr(Integer adrId) {
        return addressRepo.findByAddressid(adrId);
    }
}
```

4.2.3. SearchService

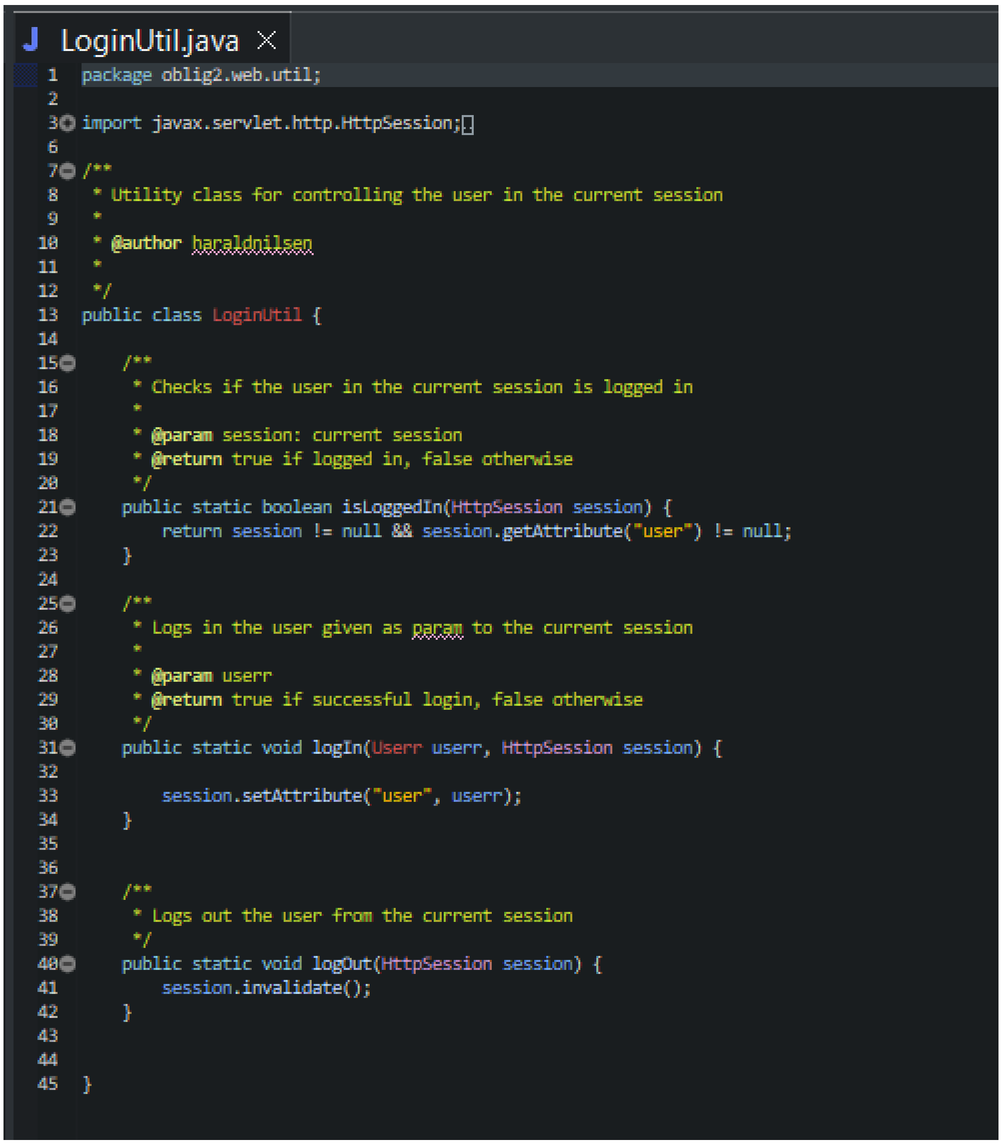


SearchService.java

```java
package oblig2.web.service;

import java.util.List;

/**
 * Utility class for searching cars for a given office
 *
 * @author AmundFremming
 *
 */

@Service
public class SearchService {

    private @Autowired CarRepo carRepo;

    /**
     * Searches the database for cars for a given office
     *
     * @return List of cars for a given office
     */
    public List<Car> searchOffice(RentalOffice office_id) {
        return carRepo.findByOfficeid(office_id);
    }

}
```

4.3. Utils

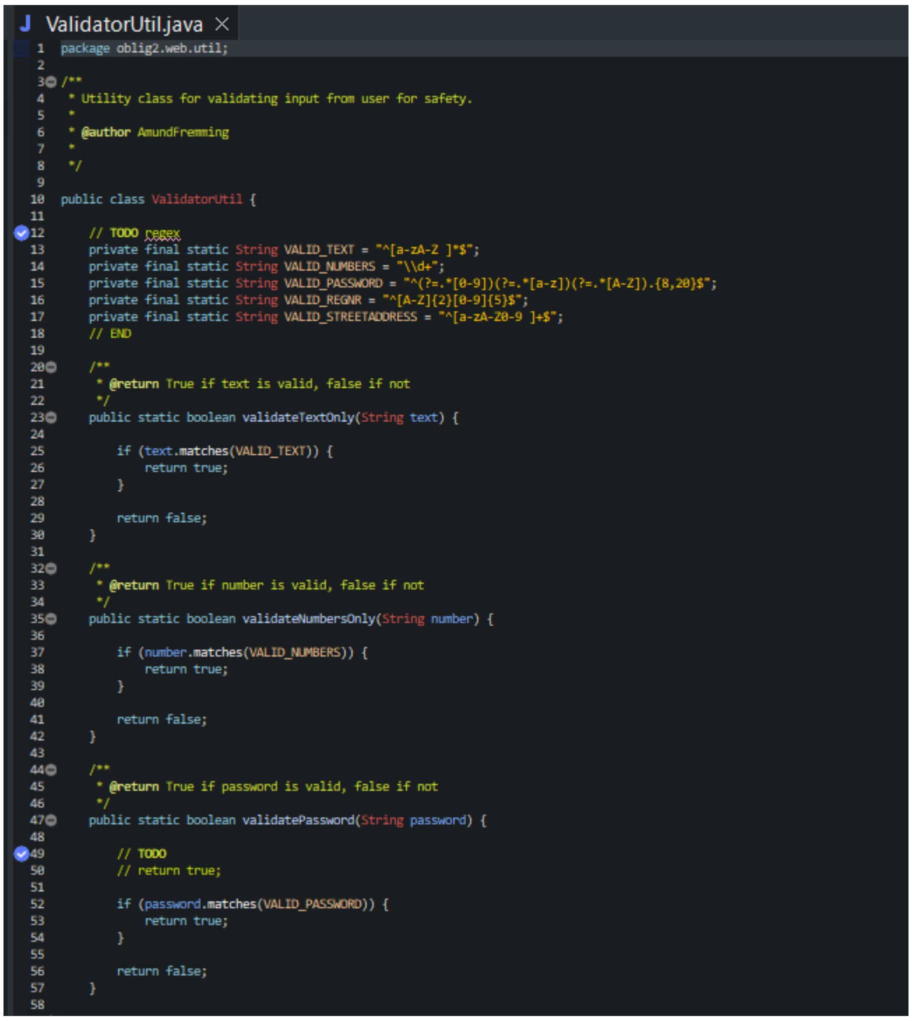
4.3.1. LoginUtil



LoginUtil.java

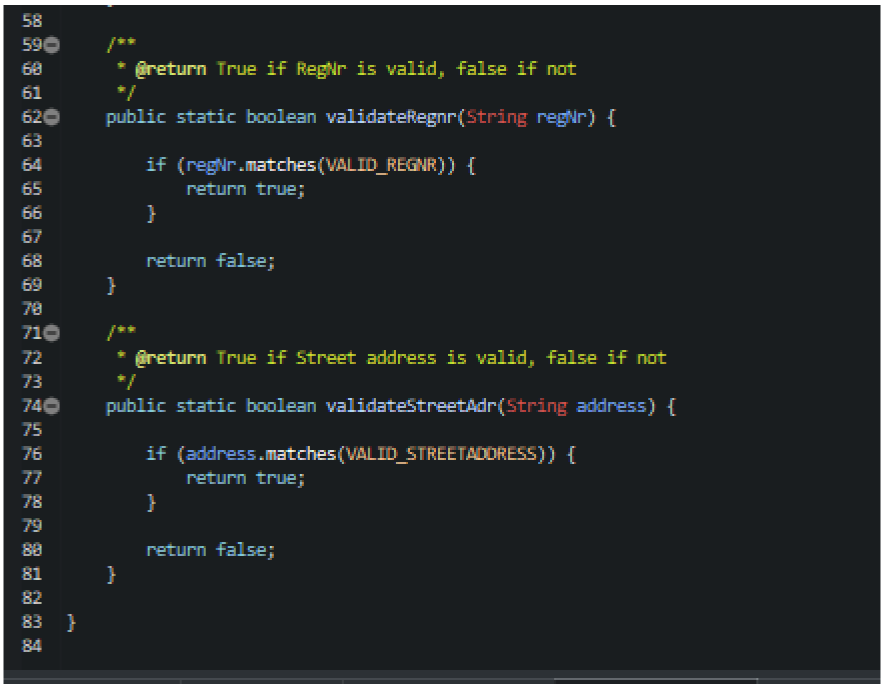
```java
package oblig2.web.util;

import javax.servlet.http.HttpSession;

/**
 * Utility class for controlling the user in the current session
 *
 * @author haraldnilsen
 *
 */
public class LoginUtil {

    /**
     * Checks if the user in the current session is logged in
     *
     * @param session: current session
     * @return true if logged in, false otherwise
     */
    public static boolean isLoggedIn(HttpSession session) {
        return session != null && session.getAttribute("user") != null;
    }

    /**
     * Logs in the user given as param to the current session
     *
     * @param userr
     * @return true if successful login, false otherwise
     */
    public static void logIn(Userr userr, HttpSession session) {

        session.setAttribute("user", userr);
    }


    /**
     * Logs out the user from the current session
     */
    public static void logOut(HttpSession session) {
        session.invalidate();
    }


}
```

4.3.2. ValidatorUtil



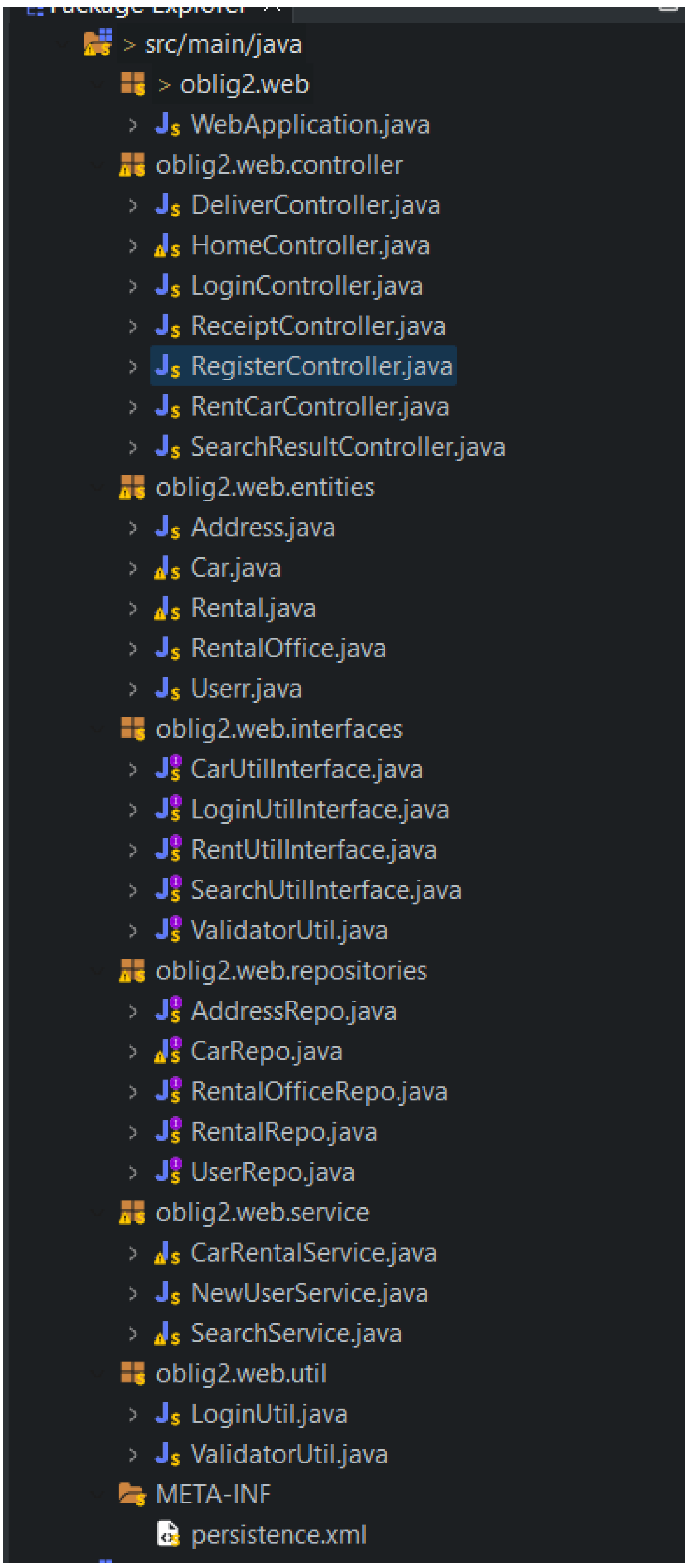
ValidatorUtil.java

```java
package oblig2.web.util;

/**
 * Utility class for validating input from user for safety.
 *
 * @author AmundFremming
 *
 */

public class ValidatorUtil {

    // TODO regex
    private final static String VALID_TEXT = "^[a-zA-Z ]*$";
    private final static String VALID_NUMBERS = "\\d+";
    private final static String VALID_PASSWORD = "^(?=.*[0-9])(?=.*[a-z])(?=.*[A-Z]).{8,20}$";
    private final static String VALID_REGNR = "^[A-Z]{2}[0-9]{5}$";
    private final static String VALID_STREETADDRESS = "^[a-zA-Z0-9 ]+$";
    // END

    /**
     * @return True if text is valid, false if not
     */
    public static boolean validateTextOnly(String text) {

        if (text.matches(VALID_TEXT)) {
            return true;
        }

        return false;
    }

    /**
     * @return True if number is valid, false if not
     */
    public static boolean validateNumbersOnly(String number) {

        if (number.matches(VALID_NUMBERS)) {
            return true;
        }

        return false;
    }

    /**
     * @return True if password is valid, false if not
     */
    public static boolean validatePassword(String password) {

        // TODO
        // return true;

        if (password.matches(VALID_PASSWORD)) {
            return true;
        }

        return false;
    }

```



```java

    /**
     * @return True if RegNr is valid, false if not
     */
    public static boolean validateRegnr(String regNr) {

        if (regNr.matches(VALID_REGNR)) {
            return true;
        }

        return false;
    }

    /**
     * @return True if Street address is valid, false if not
     */
    public static boolean validateStreetAdr(String address) {

        if (address.matches(VALID_STREETADDRESS)) {
            return true;
        }

        return false;
    }

}

```

5. Package Structure



```
src/main/java
  oblig2.web
    WebApplication.java
  oblig2.web.controller
    DeliverController.java
    HomeController.java
    LoginController.java
    ReceiptController.java
    RegisterController.java
    RentCarController.java
    SearchResultController.java
  oblig2.web.entities
    Address.java
    Car.java
    Rental.java
    RentalOffice.java
    Userr.java
  oblig2.web.interfaces
    CarUtilInterface.java
    LoginUtilInterface.java
    RentUtilInterface.java
    SearchUtilInterface.java
    ValidatorUtil.java
  oblig2.web.repositories
    AddressRepo.java
    CarRepo.java
    RentalOfficeRepo.java
    RentalRepo.java
    UserRepo.java
  oblig2.web.service
    CarRentalService.java
    NewUserService.java
    SearchService.java
  oblig2.web.util
    LoginUtil.java
    ValidatorUtil.java
  META-INF
    persistence.xml
```