Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт Космических и информационных технологий
институт
Кафедра «Информатика»
кафедра

# ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №6

Конфигурация Spring Security
тема

| | | |
|---|---|---|
| Преподаватель | | А.С. Черниговский |
| | подпись, дата | инициалы, фамилия |
| Студент КИ18-16б 031831229 | | В.А. Прекель |
| номер группы, зачетной книжки | подпись, дата | инициалы, фамилия |

Красноярск 2020

## 1 Цель работы

Познакомиться с настройкой безопасности в Spring

## 2 Общая постановка задачи

Взять практическое задание №5 и добавить

следующий функционал:

1) Добавить простейшую страницу регистрации. Пользователь вводит

свои логин и пароль и данная информация вносится в базу данных,

пользователю присваивается роль пользователя (User) приложения.

2) Добавить простейшую форму аутентификации. Форма должна быть

создана студентом, а не автоматически сгенерированной Spring.

3) В приложении должен быть предусмотрен пользователь —

администратор, с, отличной от роли User, ролью администратора (Admin).

4) Разграничить уровни доступа к страницам приложения. Пользователь

(User) имеет доступ только к страницам просмотра всех записей и запросов.

Администратор (Admin) имеет возможность добавлять, редактировать и удалять

записи.

5) Информация о пользователях и их ролях должна храниться в базе

данных. Способ хранения — на усмотрение студента.

6) Предусмотреть возможность пользователю выйти из приложения

(logout).

7) Продемонстрировать умение настраивать безопасность на уровне

представлений. Для этого реализуйте приветствие пользователя после его входа

и отображение элемента на основе его роли.

## 3 Исходный код

Листинг 1 – Lab06\src\main\java\com\github\prekel\JavaSpring\Lab06\component\FurnitureRepository.java

```
      package com.github.prekel.JavaSpring.Lab06.component;

import com.github.prekel.JavaSpring.Lab06.data.FurnitureDao;
import com.github.prekel.JavaSpring.Lab06.entity.Furniture;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;
import org.springframework.transaction.annotation.Transactional;

import java.util.List;
import java.util.Optional;

@Repository
public interface FurnitureRepository extends JpaRepository<Furniture, Integer>,
FurnitureDao {
    List<Furniture> findByType(String type);

    Optional<Furniture> findById(int id);

    default void updateById(int id, Furniture furniture) {
        furniture.setId(id);
        save(furniture);
    }

    default int insert(Furniture furniture) {
        return save(furniture).getId();
    }

    @Transactional
    void removeById(int id);
}
```

Листинг 2 – Lab06\src\main\java\com\github\prekel\JavaSpring\Lab06\component\UserRepository.java

```
      package com.github.prekel.JavaSpring.Lab06.component;

import com.github.prekel.JavaSpring.Lab06.entity.User;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;

@Repository("userRepository")
public interface UserRepository extends JpaRepository<User, Integer> {
    User findByUsername(String username);
}
```

Листинг 3 – Lab06\src\main\java\com\github\prekel\JavaSpring\Lab06\controller\AddController.java

```
      package com.github.prekel.JavaSpring.Lab06.controller;

import com.github.prekel.JavaSpring.Lab06.data.FurnitureDao;
import com.github.prekel.JavaSpring.Lab06.entity.Furniture;
import com.github.prekel.JavaSpring.Lab06.form.FurnitureForm;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;

import javax.validation.Valid;

@Controller
@RequestMapping("/add")
public class AddController {
    private static final Logger LOG = 
LoggerFactory.getLogger(AddController.class);
    private final FurnitureDao furnitureDao;

    public AddController(@Qualifier("furnitureJdbcDao") FurnitureDao 
furnitureDao) {
        this.furnitureDao = furnitureDao;
    }

    @GetMapping
    public String getForm(Model model) {
        model.addAttribute("furnitureForm", new FurnitureForm());
        return "add";
    }

    @PostMapping
    public String addByForm(@Valid FurnitureForm furnitureForm, BindingResult 
bindingResult, Model model) {
        if (bindingResult.hasErrors()) {
            return "add";
        }
        var id = furnitureDao.insert(new Furniture(furnitureForm.getType(), 
furnitureForm.getModel(),
                furnitureForm.getManufacturer(), furnitureForm.getCost(), 
furnitureForm.getHeight()));
        return "redirect:/view?id=" + id;
    }
}
```

Листинг 4 – Lab06\src\main\java\com\github\prekel\JavaSpring\Lab06\controller\DeleteController.java

```
      package com.github.prekel.JavaSpring.Lab06.controller;

import com.github.prekel.JavaSpring.Lab06.data.FurnitureDao;
import com.github.prekel.JavaSpring.Lab06.form.IdForm;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;

import javax.validation.Valid;

@Controller
@RequestMapping("/delete")
public class DeleteController {
    private static final Logger LOG =
LoggerFactory.getLogger(DeleteController.class);
    private final FurnitureDao furnitureDao;

    public DeleteController(@Qualifier("furnitureJdbcDao") FurnitureDao
furnitureDao) {
        this.furnitureDao = furnitureDao;
    }

    @GetMapping
    public String getForm(Model model) {
        model.addAttribute("idForm", new IdForm());
        return "delete";
    }

    @PostMapping
    public String addByForm(@Valid IdForm idForm, BindingResult bindingResult,
Model model) {
        if (bindingResult.hasErrors()) {
            return "delete";
        }
        var id = idForm.getId();
        furnitureDao.removeById(id);
        return "redirect:/view?id=" + id;
    }
}
```

Листинг 5 – Lab06\src\main\java\com\github\prekel\JavaSpring\Lab06\controller\EditController.java

```
      package com.github.prekel.JavaSpring.Lab06.controller;

import com.github.prekel.JavaSpring.Lab06.data.FurnitureDao;
import com.github.prekel.JavaSpring.Lab06.entity.Furniture;
import com.github.prekel.JavaSpring.Lab06.form.FurnitureForm;
import com.github.prekel.JavaSpring.Lab06.form.IdForm;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;

import javax.validation.Valid;

@Controller
@RequestMapping("/edit")
public class EditController {
    private static final Logger LOG =
LoggerFactory.getLogger(EditController.class);
    private final FurnitureDao furnitureDao;

    public EditController(@Qualifier("furnitureRepository") FurnitureDao
furnitureDao) {
        this.furnitureDao = furnitureDao;
    }

    @GetMapping
    public String getForm(Model model) {
        model.addAttribute("idForm", new IdForm());
        model.addAttribute("furnitureForm", new FurnitureForm());
        return "edit";
    }

    @GetMapping(params = "id")
    public String fillById(@Valid IdForm idForm, BindingResult bindingResult,
Model model) {
        if (bindingResult.hasErrors()) {
            model.addAttribute("furnitureForm", new FurnitureForm());
            return "edit";
        }
        if (furnitureDao.findById(idForm.getId()).isEmpty()) {
            model.addAttribute("notFound", true);
            model.addAttribute("furnitureForm", new FurnitureForm());
            return "edit";
        }
        var form = new FurnitureForm();
        form.setId(idForm.getId());
        var f = furnitureDao.findById(idForm.getId()).get();
        form.setType(f.getType());
        form.setModel(f.getModel());
        form.setManufacturer(f.getManufacturer());
        form.setCost(f.getCost());
        form.setHeight(f.getHeight());
        model.addAttribute("furnitureForm", form);
        return "edit";
    }

    @PostMapping
    public String editByForm(@Valid FurnitureForm furnitureForm, BindingResult
bindingResult, Model model) {
        if (bindingResult.hasErrors()) {
            return "edit";
        }
        if (furnitureDao.findById(furnitureForm.getId()).isEmpty()) {
            model.addAttribute("notFound", true);
            return "edit";
        }
        furnitureDao.updateById(furnitureForm.getId(), new
Furniture(furnitureForm.getType(), furnitureForm.getModel(),
                furnitureForm.getManufacturer(), furnitureForm.getCost(),
furnitureForm.getHeight()));
        return "redirect:/view?id=" + furnitureForm.getId();
    }
}
```

Листинг 6 – Lab06\src\main\java\com\github\prekel\JavaSpring\Lab06\controller\ErrorController.java

```
      package com.github.prekel.JavaSpring.Lab06.controller;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
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;

@Controller
@RequestMapping("/error")
public class ErrorController {
    private static final Logger LOG =
LoggerFactory.getLogger(ErrorController.class);

    @GetMapping
    public String showError() {
        return "error";
    }

    @PostMapping
    public String errorConfirm() {
        return "redirect:/";
    }
}
```

Листинг 7 – Lab06\src\main\java\com\github\prekel\JavaSpring\Lab06\controller\IndexController.java

```
      package com.github.prekel.JavaSpring.Lab06.controller;


import com.github.prekel.JavaSpring.Lab06.data.FurnitureDao;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;

@Controller
@RequestMapping("/")
public class IndexController {
    private static final Logger LOG =
LoggerFactory.getLogger(IndexController.class);
    private final FurnitureDao furnitureDao;

    public IndexController(@Qualifier("furnitureJdbcDao") FurnitureDao
furnitureDao) {
        this.furnitureDao = furnitureDao;
    }

    @GetMapping
    public String showHomePage(Model model) {
        model.addAttribute("count", furnitureDao.count());
        return "index";
    }

    @GetMapping("/admin")
    public String privateHome() {
        return "private";
    }
}
```

Листинг 8 – Lab06\src\main\java\com\github\prekel\JavaSpring\Lab06\controller\RegistrationController.java

```
      package com.github.prekel.JavaSpring.Lab06.controller;

import com.github.prekel.JavaSpring.Lab06.component.UserRepository;
import com.github.prekel.JavaSpring.Lab06.entity.User;
import com.github.prekel.JavaSpring.Lab06.form.UserForm;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;

import javax.validation.Valid;

@Controller
@RequestMapping("/registration")
public class RegistrationController {
    private static final Logger LOG =
LoggerFactory.getLogger(RegistrationController.class);
    public final BCryptPasswordEncoder passwordEncoder;
    private final UserRepository repository;

    public RegistrationController(@Qualifier("userRepository") UserRepository
repository, @Qualifier("passwordEncoder") BCryptPasswordEncoder passwordEncoder)
{
        this.repository = repository;
        this.passwordEncoder = passwordEncoder;
    }

    @GetMapping
    public String showAddShoesForm(Model model) {
        model.addAttribute("userForm", new UserForm());
        return "registration";
    }

    @PostMapping
    public String saveUserToDb(@Valid UserForm userForm, BindingResult
bindingResult, Model model) {
        try {
            User newUser = new User(userForm.getUsername(),
passwordEncoder.encode(userForm.getPassword()));
            repository.save(newUser);
            System.out.println("New user created");
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
        return "redirect:/";
    }
}
```

Листинг 9 – Lab06\src\main\java\com\github\prekel\JavaSpring\Lab06\controller\ViewController.java

```
      package com.github.prekel.JavaSpring.Lab06.controller;

import com.github.prekel.JavaSpring.Lab06.data.FurnitureDao;
import com.github.prekel.JavaSpring.Lab06.form.IdForm;
import com.github.prekel.JavaSpring.Lab06.form.TypeForm;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;

import javax.validation.Valid;
import java.util.stream.Collectors;

@Controller
@RequestMapping("/view")
public class ViewController {
    private static final Logger LOG =
LoggerFactory.getLogger(ViewController.class);
    private final FurnitureDao furnitureDao;

    public ViewController(@Qualifier("furnitureRepository") FurnitureDao
furnitureDao) {
        this.furnitureDao = furnitureDao;
    }

    @GetMapping
    public String viewByIdOrAll(Model model) {
        model.addAttribute("idForm", new IdForm());
        model.addAttribute("typeForm", new TypeForm());

        var furnitures = furnitureDao.findAll();
        model.addAttribute("furnitures", furnitures);
        return "view";
    }

    @GetMapping(params = "id")
    public String viewByIdOrAll(@Valid IdForm idForm, BindingResult
bindingResultIdForm, TypeForm typeForm, Model model) {
        if (bindingResultIdForm.hasErrors()) {
            return "view";
        }
        if (idForm.getId() == 0) {
            return "redirect:/view";
        }
        var furnitures =
furnitureDao.findById(idForm.getId()).stream().collect(Collectors.toList());
        model.addAttribute("furnitures", furnitures);
        return "view";
    }

    @GetMapping(params = "type")
    public String viewByType(@Valid TypeForm typeForm, BindingResult
bindingResultTypeForm, IdForm idForm, Model model) {
        if (bindingResultTypeForm.hasErrors()) {
            return "view";
        }
        var furnitures = furnitureDao.findByType(typeForm.getType());
        model.addAttribute("furnitures", furnitures);
        return "view";
    }
}
```

Листинг 10 – Lab06\src\main\java\com\github\prekel\JavaSpring\Lab06\data\FurnitureDao.java

```
      package com.github.prekel.JavaSpring.Lab06.data;

import com.github.prekel.JavaSpring.Lab06.entity.Furniture;

import java.util.List;
import java.util.Optional;

public interface FurnitureDao {
    List<Furniture> findAll();

    List<Furniture> findByType(String type);

    Optional<Furniture> findById(int id);

    void updateById(int id, Furniture furniture);

    int insert(Furniture furniture);

    void removeById(int id);

    long count();
}
```

Листинг 11 – Lab06\src\main\java\com\github\prekel\JavaSpring\Lab06\data\FurnitureJdbcDao.java

```
      package com.github.prekel.JavaSpring.Lab06.data;

import com.github.prekel.JavaSpring.Lab06.entity.Furniture;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.BeanPropertyRowMapper;
import org.springframework.jdbc.core.JdbcTemplate;

import javax.sql.DataSource;
import java.sql.Types;
import java.util.List;
import java.util.Optional;

public class FurnitureJdbcDao implements FurnitureDao {
    private JdbcTemplate jdbcTemplate;

    @Autowired
    public void setDataSource(DataSource dataSource) {
        this.jdbcTemplate = new JdbcTemplate(dataSource);
    }

    @Override
    public List<Furniture> findAll() {
        return jdbcTemplate.query("SELECT * FROM Furniture", new
BeanPropertyRowMapper<>(Furniture.class));
    }
    @Override
    public List<Furniture> findByType(String type) {
        return jdbcTemplate.query("SELECT * FROM Furniture WHERE type = ?", new
Object[]{type}, new BeanPropertyRowMapper<>(Furniture.class));
    }

    @Override
    public Optional<Furniture> findById(int id) {
        var ret = jdbcTemplate.query("SELECT * FROM Furniture WHERE id = ?", new
Object[]{id}, new BeanPropertyRowMapper<>(Furniture.class));
        return ret.stream().findFirst();
    }

    @Override
    public void updateById(int id, Furniture furniture) {
        jdbcTemplate.update("UPDATE Furniture SET type = ?, model = ?,
manufacturer = ?, cost = ?, height = ? WHERE id = ?",
                furniture.getType(), furniture.getModel(),
furniture.getManufacturer(), furniture.getCost(), furniture.getHeight(), id);
    }

    @Override
    public int insert(Furniture furniture) {
        return jdbcTemplate.queryForObject("INSERT INTO Furniture (id, type,
model, manufacturer, cost, height) VALUES (DEFAULT,?,?,?,?,?) RETURNING id",
                new Object[]{furniture.getType(), furniture.getModel(),
furniture.getManufacturer(), furniture.getCost(), furniture.getHeight()},
                new int[]{Types.VARCHAR, Types.VARCHAR, Types.VARCHAR,
Types.NUMERIC, Types.DOUBLE}, Integer.class);
    }

    @Override
    public void removeById(int id) {
        jdbcTemplate.update("DELETE FROM Furniture WHERE id = ?", id);
    }

    @Override
    public long count() {
        return jdbcTemplate.queryForObject("SELECT count(*) FROM Furniture",
Long.class);
    }
}
```

Листинг 12 – Lab06\src\main\java\com\github\prekel\JavaSpring\Lab06\DatabaseConfig.java

```
      package com.github.prekel.JavaSpring.Lab06;

import com.github.prekel.JavaSpring.Lab06.data.FurnitureDao;
import com.github.prekel.JavaSpring.Lab06.data.FurnitureJdbcDao;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;
import org.springframework.context.annotation.PropertySource;
import org.springframework.core.env.Environment;
import org.springframework.data.jpa.repository.config.EnableJpaRepositories;
import org.springframework.jdbc.datasource.DriverManagerDataSource;
import org.springframework.orm.jpa.JpaTransactionManager;
import org.springframework.orm.jpa.LocalContainerEntityManagerFactoryBean;
import org.springframework.orm.jpa.vendor.HibernateJpaVendorAdapter;
import org.springframework.transaction.PlatformTransactionManager;
import org.springframework.transaction.annotation.EnableTransactionManagement;

import javax.persistence.EntityManagerFactory;
import javax.sql.DataSource;

@Configuration
@EnableJpaRepositories
@EnableTransactionManagement
@ComponentScan("com.github.prekel.JavaSpring.Lab06.component")
@PropertySource("classpath:application.properties")
public class DatabaseConfig {
    private static final Logger LOG = 
LoggerFactory.getLogger(DatabaseConfig.class);
    @Autowired
    private Environment env;

    @Bean
    public FurnitureDao furnitureJdbcDao() {
        return new FurnitureJdbcDao();
    }

    @Bean
    public PlatformTransactionManager transactionManager() {
        var txManager = new JpaTransactionManager();
        txManager.setEntityManagerFactory(entityManagerFactory());
        return txManager;
    }

    @Bean
    public EntityManagerFactory entityManagerFactory() {
        var vendorAdapter = new HibernateJpaVendorAdapter();
        vendorAdapter.setGenerateDdl(true);
        var factory = new LocalContainerEntityManagerFactoryBean();
        factory.setJpaVendorAdapter(vendorAdapter);
        factory.setPackagesToScan("com.github.prekel.JavaSpring.Lab06.entity");
        factory.setDataSource(dataSource());
        factory.afterPropertiesSet();
        return factory.getObject();
    }

    @Bean
    public DataSource dataSource() {
        var dataSource = new DriverManagerDataSource();

dataSource.setDriverClassName(env.getProperty("dataSource.driverClassName"));
        dataSource.setUrl(env.getProperty("dataSource.url"));
        dataSource.setUsername(env.getProperty("dataSource.username"));
        dataSource.setPassword(env.getProperty("dataSource.password"));

        return dataSource;
    }
}
```

Листинг 13 – Lab06\src\main\java\com\github\prekel\JavaSpring\Lab06\entity\Furniture.java

```
      package com.github.prekel.JavaSpring.Lab06.entity;

import javax.persistence.*;
import java.math.BigDecimal;
import java.util.StringJoiner;

@Entity
public class Furniture {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;
    @Column
    private String type;
    @Column
    private String model;
    @Column
    private String manufacturer;
    @Column
    private BigDecimal cost;
    @Column
    private double height;

    public Furniture() {

    }

    public Furniture(String type, String model, String manufacturer, BigDecimal
cost, double height) {
        this.type = type;
        this.model = model;
        this.manufacturer = manufacturer;
        this.cost = cost;
        this.height = height;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getType() {
        return type;
    }

    public void setType(String type) {
        this.type = type;
    }

    public String getModel() {
        return model;
    }

    public void setModel(String model) {
        this.model = model;
    }

    public String getManufacturer() {
        return manufacturer;
    }

    public void setManufacturer(String manufacturer) {
        this.manufacturer = manufacturer;
    }

    public BigDecimal getCost() {
        return cost;
    }

    public void setCost(BigDecimal cost) {
        this.cost = cost;
    }

    public double getHeight() {
        return height;
    }

    public void setHeight(double height) {
        this.height = height;
    }

    @Override
    public String toString() {
        return new StringJoiner(", ", Furniture.class.getSimpleName() + "[",
"]")
                .add("id=" + id)
                .add("type='" + type + "'")
                .add("model='" + model + "'")
                .add("manufacturer='" + manufacturer + "'")
                .add("cost=" + cost)
                .add("height=" + height)
                .toString();
    }
}
```

Листинг 14 – Lab06\src\main\java\com\github\prekel\JavaSpring\Lab06\entity\User.java

```
      package com.github.prekel.JavaSpring.Lab06.entity;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.Table;

@Table(name = "users")
@Entity
public class User {

    @Id
    @GeneratedValue
```

```
private int id;

private String username;
private String password;
private String role = "user";
private byte enabled = 1;

public User() {
}

public User(String username, String password) {
    this.username = username;
    this.password = password;
}

public byte getEnabled() {
    return enabled;
}

public void setEnabled(byte enabled) {
    this.enabled = enabled;
}

public int getId() {
    return id;
}

public void setId(int id) {
    this.id = id;
}

public String getUsername() {
    return username;
}

public void setUsername(String username) {
    this.username = username;
}

public String getPassword() {
    return password;
}

public void setPassword(String password) {
    this.password = password;
}

public String getRole() {
    return role;
}

public void setRole(String role) {
    this.role = role;
}

@Override
public String toString() {
    return "Users" +
            "id=" + id +
            ", username='" + username + '\'' +
            ", password='" + password + '\'' +
            ", role='" + role + '\'';
}
```

```
}
```

Листинг 15 – Lab06\src\main\java\com\github\prekel\JavaSpring\Lab06\form\FurnitureForm.java

```
      package com.github.prekel.JavaSpring.Lab06.form;

import javax.validation.constraints.*;
import java.math.BigDecimal;

public class FurnitureForm {
    @NotNull(message = "РџРѕР»Рµ РЅРµ РґРѕР»Р¶РЅРѕ Р±С‹С‚СЊ РїСѓСЃС‚С‹Рј")
    @PositiveOrZero(message = "Р”РѕР»Р¶РЅРѕ Р±С‹С‚СЊ РЅРµ
РѕС‚СЂРёС†Р°С‚РµР»СЊРЅС‹Рј")
    private int id;
    @NotBlank(message = "РџРѕР»Рµ РЅРµ РґРѕР»Р¶РЅРѕ Р±С‹С‚СЊ РїСѓСЃС‚С‹Рј")
    private String type;
    @NotBlank(message = "РџРѕР»Рµ РЅРµ РґРѕР»Р¶РЅРѕ Р±С‹С‚СЊ РїСѓСЃС‚С‹Рј")
    private String model;
    @NotBlank(message = "РџРѕР»Рµ РЅРµ РґРѕР»Р¶РЅРѕ Р±С‹С‚СЊ РїСѓСЃС‚С‹Рј")
    private String manufacturer;
    @NotNull(message = "РџРѕР»Рµ РЅРµ РґРѕР»Р¶РЅРѕ Р±С‹С‚СЊ РїСѓСЃС‚С‹Рј")
    @Positive(message = "Р”РѕР»Р¶РЅРѕ Р±С‹С‚СЊ РїРѕР»РѕР¶РёС‚РµР»СЊРЅС‹Рј")
    private BigDecimal cost;
    @NotNull(message = "РџРѕР»Рµ РЅРµ РґРѕР»Р¶РЅРѕ Р±С‹С‚СЊ РїСѓСЃС‚С‹Рј")
    @Min(value = 10, message = "РќРµ РґРѕР»Р¶РЅРѕ Р±С‹С‚СЊ РјРµРЅРµРµ 10")
    @Max(value = 1000, message = "РќРµ РґРѕР»Р¶РЅРѕ Р±С‹С‚СЊ Р±РѕР»РµРµ 1000")
    private double height;

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getType() {
        return type;
    }

    public void setType(String type) {
        this.type = type;
    }

    public String getModel() {
        return model;
    }

    public void setModel(String model) {
        this.model = model;
    }

    public String getManufacturer() {
        return manufacturer;
    }
    public void setManufacturer(String manufacturer) {
        this.manufacturer = manufacturer;
    }

    public BigDecimal getCost() {
        return cost;
    }

    public void setCost(BigDecimal cost) {
        this.cost = cost;
    }

    public double getHeight() {
        return height;
    }

    public void setHeight(double height) {
        this.height = height;
    }
}
```

Листинг 16 – Lab06\src\main\java\com\github\prekel\JavaSpring\Lab06\form\IdForm.java

```
      package com.github.prekel.JavaSpring.Lab06.form;

import javax.validation.constraints.NotNull;
import javax.validation.constraints.PositiveOrZero;

public class IdForm {
    @NotNull(message = "PџPsP»Pµ PSPµ PґPsP»P¶PSPs P±C‹C,CЊ PїCѓCЃC,C‹Pj")
    @PositiveOrZero(message = "P”PsP»P¶PSPs P±C‹C,CЊ PSPµ
PsC,CЂPёC†P°C,PµP»CЊPSC‹Pj")
    private int id;

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }
}
```

Листинг 17 – Lab06\src\main\java\com\github\prekel\JavaSpring\Lab06\form\TypeForm.java

```
      package com.github.prekel.JavaSpring.Lab06.form;

import javax.validation.constraints.NotBlank;

public class TypeForm {
    @NotBlank(message = "PџPsP»Pµ PSPµ PґPsP»P¶PSPs P±C‹C‚CЊ PїCѓCЃC‚C‹Pj")
    private String type;

    public String getType() {
        return type;
    }

    public void setType(String type) {
        this.type = type;
    }
}
```

Листинг 18 – Lab06\src\main\java\com\github\prekel\JavaSpring\Lab06\form\UserForm.java

```
      package com.github.prekel.JavaSpring.Lab06.form;

import org.hibernate.validator.constraints.Length;

import javax.validation.constraints.NotBlank;

public class UserForm {
    @NotBlank(message = "PџPsP»Pµ PSPµ PґPsP»P¶PSPs P±C‹C‚CЊ PїCѓCЃC‚C‹Pj")
    @Length(min = 6, max = 20)
    public String username;
    @NotBlank(message = "PџPsP»Pµ PSPµ PґPsP»P¶PSPs P±C‹C‚CЊ PїCѓCЃC‚C‹Pj")
    @Length(min = 6, max = 20)
    public String password;

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }
}
```

Листинг 19 – Lab06\src\main\java\com\github\prekel\JavaSpring\Lab06\Lab06Application.java

```
      package com.github.prekel.JavaSpring.Lab06;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class Lab06Application {
    private static final Logger LOG =
LoggerFactory.getLogger(Lab06Application.class);

    public static void main(String[] args) {
        LOG.info("Started");
        SpringApplication.run(Lab06Application.class, args);
        LOG.info("Ended");
    }
}
```

Листинг 20 – Lab06\src\main\java\com\github\prekel\JavaSpring\Lab06\ReadWithCheckBuilder.java

```
      package com.github.prekel.JavaSpring.Lab06;

import java.io.BufferedReader;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.function.Function;

public class ReadWithCheckBuilder<T> {
    private final BufferedReader reader;
    private String message = "";
    private Function<String, T> parser = (s -> (T) s);
    private Function<T, Boolean> checker = (obj -> true);

    public ReadWithCheckBuilder() {
        this(System.in);
    }

    public ReadWithCheckBuilder(InputStream in) {
        this(new BufferedReader(new InputStreamReader(in)));
    }

    public ReadWithCheckBuilder(BufferedReader reader) {
        this.reader = reader;
    }

    public ReadWithCheckBuilder<T> hasMessage(String message) {
        this.message = message;
        return this;
    }

    public ReadWithCheckBuilder<T> hasParser(Function<String, T> parser) {
        this.parser = parser;
        return this;
    }

    public ReadWithCheckBuilder<T> hasChecker(Function<T, Boolean> checker) {
        this.checker = checker;
        return this;
    }

    public T readCycle() {
        while (true) {
            System.out.print(message);
            try {
                var parsed = parser.apply(reader.readLine());
                if (!checker.apply(parsed)) {
                    throw new Exception("РќРµ РІ РїСЂРѕРјРµР¶СѓС‚РєРµ РёР»Рё
РїСѓСЃС‚Р°СЏ СЃС‚СЂРѕРєР°");
                }
                return parsed;
            } catch (Exception e) {
                System.err.println(e.getMessage());
            }
        }
    }
}
```

Листинг 21 – Lab06\src\main\java\com\github\prekel\JavaSpring\Lab06\service\CustomUserDetails.java

```
      package com.github.prekel.JavaSpring.Lab06.service;

import com.github.prekel.JavaSpring.Lab06.entity.User;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.security.core.GrantedAuthority;
import org.springframework.security.core.authority.SimpleGrantedAuthority;
import org.springframework.security.core.userdetails.UserDetails;

import java.util.Collection;
import java.util.Collections;

public class CustomUserDetails implements UserDetails {
    private static final Logger LOG =
LoggerFactory.getLogger(CustomUserDetails.class);

    private final User user;

    public CustomUserDetails(User user) {
        this.user = user;
    }

    @Override
    public Collection<? extends GrantedAuthority> getAuthorities() {
        SimpleGrantedAuthority authority = new
SimpleGrantedAuthority(user.getRole());
        return Collections.singletonList(authority);
    }

    @Override
    public String getPassword() {
        return user.getPassword();
    }

    @Override
    public String getUsername() {
        return user.getUsername();
    }

    @Override
    public boolean isAccountNonExpired() {
        return true;
    }

    @Override
    public boolean isAccountNonLocked() {
        return true;
    }

    @Override
    public boolean isCredentialsNonExpired() {
        return true;
    }

    @Override
    public boolean isEnabled() {
        return true;
    }
}
```

Листинг 22 – Lab06\src\main\java\com\github\prekel\JavaSpring\Lab06\service\CustomUserDetailsService.java

```
      package com.github.prekel.JavaSpring.Lab06.service;

import com.github.prekel.JavaSpring.Lab06.component.UserRepository;
import com.github.prekel.JavaSpring.Lab06.entity.User;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.security.core.userdetails.UserDetails;
import org.springframework.security.core.userdetails.UserDetailsService;
import org.springframework.security.core.userdetails.UsernameNotFoundException;
import org.springframework.stereotype.Service;

@Service("customUserDetailsService")
public class CustomUserDetailsService implements UserDetailsService {
    private static final Logger LOG =
LoggerFactory.getLogger(CustomUserDetailsService.class);
    @Qualifier("userRepository")
    @Autowired
    private UserRepository userRepository;

    @Override
    public UserDetails loadUserByUsername(String username)
```

```
            throws UsernameNotFoundException {
        User user = userRepository.findByUsername(username);

        if (user == null) {
            throw new UsernameNotFoundException("Could not find user");
        }

        return new CustomUserDetails(user);
    }
}
```

Листинг 23 – Lab06\src\main\java\com\github\prekel\JavaSpring\Lab06\WebSecurityConfig.java

```
      package com.github.prekel.JavaSpring.Lab06;

import com.github.prekel.JavaSpring.Lab06.service.CustomUserDetailsService;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;
import
org.springframework.security.authentication.dao.DaoAuthenticationProvider;
import
org.springframework.security.config.annotation.authentication.builders.Authentic
ationManagerBuilder;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import
org.springframework.security.config.annotation.web.configuration.EnableWebSecuri
ty;
import
org.springframework.security.config.annotation.web.configuration.WebSecurityConf
igurerAdapter;
import org.springframework.security.core.userdetails.UserDetailsService;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

@Configuration
@EnableWebSecurity
@ComponentScan(basePackageClasses = CustomUserDetailsService.class)
public class WebSecurityConfig extends WebSecurityConfigurerAdapter {
    private static final Logger LOG =
LoggerFactory.getLogger(WebSecurityConfig.class);

    @Qualifier("customUserDetailsService")
    @Autowired
    private UserDetailsService userDetailsService;

    @Override
    protected void configure(AuthenticationManagerBuilder auth) throws Exception
{
        auth.authenticationProvider(authenticationProvider());
    }

    @Bean
    public DaoAuthenticationProvider authenticationProvider() {
        DaoAuthenticationProvider authProvider = new
DaoAuthenticationProvider();
        authProvider.setUserDetailsService(userDetailsService);
        authProvider.setPasswordEncoder(passwordEncoder());

        return authProvider;
    }

    @Bean
    public BCryptPasswordEncoder passwordEncoder() {
        return new BCryptPasswordEncoder();
    }

    @Override
    protected void configure(HttpSecurity http) throws Exception {
        http
                .csrf().disable()
                .authorizeRequests()
                .antMatchers("/", "/registration").permitAll()
                .antMatchers("/view").hasAnyAuthority("admin", "user")
                .antMatchers("/add", "/edit", "/delete").hasAuthority("admin")
                .anyRequest().authenticated()
                .and()
                .exceptionHandling().accessDeniedPage("/error")
                .and()
                .formLogin()
                .permitAll()
                .and()
                .logout()
                .permitAll();
    }
}
```

Листинг 24 – Lab06\src\main\resources\application.properties

```
      #--- Postgres ---
dataSource.driverClassName=org.postgresql.Driver
jpa.database=POSTGRESQL
dataSource.url=jdbc:postgresql://localhost:5432/javaspring
dataSource.username=postgres
dataSource.password=qwerty123
server.port=8888
logging.level.org.springframework.web=DEBUG
spring.mvc.log-request-details=true
```

Листинг 25 – Lab06\src\main\resources\templates\add.html

```
      <!DOCTYPE HTML>
<html lang="ru" xmlns:th="https://www.thymeleaf.org">
<head>
    <title>Р”РѕР±Р°РІРёС‚СЊ РјРµР±РµР»СЊ</title>
    <meta content="text/html; charset=UTF-8" http-equiv="Content-Type"/>
</head>
```

```
<body>
<header>
    <h1>P”PsP±P°PIPëC,CЊ PjPµP±PµP»CЊ</h1>
</header>
<article>
    <form class="form" method="post" th:object="${furnitureForm}">
        <div>
            <p>PўPëPï</p>
            <label>
                <input name="type" placeholder="type" th:field="*{type}"
type="text"/>
            </label>
            <span th:errors="*{type}" th:if="${#fields.hasErrors('type')}">type
error</span>
        </div>
        <div>
            <p>PњPsPґPµP»CЊ</p>
            <label>
                <input name="model" placeholder="model" th:field="*{model}"
type="text"/>
            </label>
            <span th:errors="*{model}"
th:if="${#fields.hasErrors('model')}">model error</span>
        </div>
        <div>
            <p>PџCЂPsPëP·PIPsPґPëC,PµP»CЊ</p>
            <label>
                <input name="manufacturer" placeholder="manufacturer"
th:field="*{manufacturer}" type="text"/>
            </label>
            <span th:errors="*{manufacturer}"
th:if="${#fields.hasErrors('manufacturer')}">Name error</span>
        </div>
        <div>
            <p>P¦PµPSP°</p>
            <label>
                <input name="cost" placeholder="cost" th:field="*{cost}"
type="number"/>
            </label>
            <span th:errors="*{cost}" th:if="${#fields.hasErrors('cost')}">cost
error</span>
        </div>
        <div>
            <p>P’C‹CЃPsC,P°</p>
            <label>
                <input name="height" placeholder="height" th:field="*{height}"
type="number"/>
            </label>
            <span th:errors="*{height}"
th:if="${#fields.hasErrors('height')}">height error</span>
        </div>
        <div class="buttons">
            <br>
            <button type="submit">P”PsP±P°PIPëC,CЊ</button>
        </div>
    </form>
</article>
<footer>
    <hr>
    <a href="/">P”PsPjPsP№</a>
</footer>
</body>
```

```
</html>
```

Листинг 26 – Lab06\src\main\resources\templates\error.html

```
      <!DOCTYPE html>
<html lang="en">
<head>
    <title>РћС€РёР±РєР° РґРѕСЃС‚СѓРїР°</title>
    <meta content="text/html; charset=UTF-8" http-equiv="Content-Type"/>
</head>
<body>
<header>
    <h1>РћС€РёР±РєР° РґРѕСЃС‚СѓРїР°</h1>
</header>
<article>
    <form method="post">
        <button type="submit">Ok</button>
    </form>
</article>
<footer>
    <hr>
    <a href="/">Р”РѕРјРѕР№</a>
</footer>
</body>
</html>
```

Листинг 27 – Lab06\src\main\resources\templates\index.html

```
      <!DOCTYPE HTML>
<html lang="ru"
      xmlns:sec="http://www.thymeleaf.org/extras/spring-security"
      xmlns:th="https://www.thymeleaf.org">
<head>
    <title>РњРµР±РµР»СЊ</title>
    <meta content="text/html; charset=UTF-8" http-equiv="Content-Type"/>
</head>
<body>
<header>
    <h1>РњРµР±РµР»СЊ</h1>
</header>
<article>
    <section>
        <ul>
            <li sec:authorize="isAuthenticated()">
                <form method="post" th:action="@{/logout}">
                    <button type="submit">Log out</button>
                </form>
            </li>
            <li sec:authorize="isAnonymous()"><a
href="/registration">Р РµРіРёСЃС‚СЂР°С†РёСЏ</a></li>
            <li sec:authorize="isAnonymous()"><a href="/login">Р’С…РѕРґ</a></li>
        </ul>
        <p sec:authorize="isAuthenticated()">Welcome,
[[${#httpServletRequest.remoteUser}]]!</p>
```

```
    </section>
    <section>
        <p th:text="'РњРµР±РµР»Рё РІ РєР°С‚Р°Р»РѕРіРµ: ' +
${count}">РњРµР±РµР»Рё РІ РєР°С‚Р°Р»РѕРіРµ:</p>
        <ul>
            <li><a href="/add">Р”РѕР±Р°РІРёС‚СЊ РјРµР±РµР»СЊ</a></li>
            <li><a href="/edit">Р РµРґР°РєС‚РёСЂРѕРІР°С‚СЊ РјРµР±РµР»СЊ</a></li>
            <li><a href="/delete">РЈРґР°Р»РёС‚СЊ РјРµР±РµР»СЊ</a></li>
            <li><a href="/view">РџРѕСЃРјРѕС‚СЂРµС‚СЊ РјРµР±РµР»СЊ</a></li>
        </ul>
    </section>
</article>
<footer>
    <hr>
    <a href="/">Р”РѕРјРѕР№</a>
</footer>
</body>
</html>
```

Листинг 28 – Lab06\src\main\resources\templates\private.html

```
      <!DOCTYPE html>
<html lang="en">
<head>
    <title>РџСЂРёРІР°С‚РЅР°СЏ СЃС‚СЂР°РЅРёС†Р°</title>
    <meta content="text/html; charset=UTF-8" http-equiv="Content-Type"/>
</head>
<body>
<header>
    <h1>РџСЂРёРІР°С‚РЅР°СЏ СЃС‚СЂР°РЅРёС†Р°</h1>
</header>
<article>
</article>
<footer>
    <hr>
    <a href="/">Р”РѕРјРѕР№</a>
</footer>
</body>
</html>
```