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

Институт Космических и информационных технологий
институт
Кафедра «Информатика»
кафедра

ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №5

Spring MVC
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 шаблоном MVC в Spring и тем как он используется при создании web-приложений.

2 Общая постановка задачи

Изменить практическую работу №4 таким образом, чтобы она представляла собой web-приложение.

Web-приложение должно иметь следующие страницы:

1) Главная страница, содержит приветствие и ссылки на другие (которые дублируют по функционалу пункты меню из работы №4).

2) Страница просмотра таблицы записей.

3) Страница добавления новой записи в таблицу.

4) Страница редактирования записи.

5) Страница удаления записи из таблицы БД.

6) Страница просмотра записей согласно некоторому критерию (аналогично пункту в работе №4).

Помимо всего должны быть осуществлены проверки (не менее двух) входных данных, сопровождающиеся соответствующими сообщениями об ошибках.

3 Исходный код

Листинг 1 – Lab05\src\main\java\com\github\prekel\JavaSpring\Lab05\component\FurnitureRepository.java

```
package com.github.prekel.JavaSpring.Lab05.component;

import com.github.prekel.JavaSpring.Lab05.data.FurnitureDao;
import com.github.prekel.JavaSpring.Lab05.entity.Furniture;
import org.springframework.data.domain.Example;
import org.springframework.data.domain.ExampleMatcher;
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);

    @Transactional
    void removeById(int id);

    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();
    }
}
```

Листинг 2 – Lab05\src\main\java\com\github\prekel\JavaSpring\Lab05\controller\AddController.java

```
      package com.github.prekel.JavaSpring.Lab05.controller;

import com.github.prekel.JavaSpring.Lab05.data.FurnitureDao;
import com.github.prekel.JavaSpring.Lab05.entity.Furniture;
import com.github.prekel.JavaSpring.Lab05.form.FurnitureForm;
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 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 3 – Lab05\src\main\java\com\github\prekel\JavaSpring\Lab05\controller\DeleteController.java

```
      package com.github.prekel.JavaSpring.Lab05.controller;

import com.github.prekel.JavaSpring.Lab05.data.FurnitureDao;
import com.github.prekel.JavaSpring.Lab05.form.IdForm;
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 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 4 – Lab05\src\main\java\com\github\prekel\JavaSpring\Lab05\controller\EditController.java

```
      package com.github.prekel.JavaSpring.Lab05.controller;

import com.github.prekel.JavaSpring.Lab05.data.FurnitureDao;
import com.github.prekel.JavaSpring.Lab05.entity.Furniture;
import com.github.prekel.JavaSpring.Lab05.form.FurnitureForm;
import com.github.prekel.JavaSpring.Lab05.form.IdForm;
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 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 5 – Lab05\src\main\java\com\github\prekel\JavaSpring\Lab05\controller\IndexController.java

```
      package com.github.prekel.JavaSpring.Lab05.controller;


import com.github.prekel.JavaSpring.Lab05.data.FurnitureDao;
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 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";
    }
}
```

Листинг 6 – Lab05\src\main\java\com\github\prekel\JavaSpring\Lab05\controller\ViewController.java

```
      package com.github.prekel.JavaSpring.Lab05.controller;

import com.github.prekel.JavaSpring.Lab05.data.FurnitureDao;
import com.github.prekel.JavaSpring.Lab05.form.IdForm;
import com.github.prekel.JavaSpring.Lab05.form.TypeForm;
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 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 7 – Lab05\src\main\java\com\github\prekel\JavaSpring\Lab05\data\FurnitureDao.java

```
      package com.github.prekel.JavaSpring.Lab05.data;

import com.github.prekel.JavaSpring.Lab05.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 8 – Lab05\src\main\java\com\github\prekel\JavaSpring\Lab05\data\FurnitureJdbcDao.java

```
      package com.github.prekel.JavaSpring.Lab05.data;

import com.github.prekel.JavaSpring.Lab05.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 9 –

Lab05\src\main\java\com\github\prekel\JavaSpring\Lab05\entity\Furniture.java

```
      package com.github.prekel.JavaSpring.Lab05.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 10 – Lab05\src\main\java\com\github\prekel\JavaSpring\Lab05\form\FurnitureForm.java

```
      package com.github.prekel.JavaSpring.Lab05.form;

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

public class FurnitureForm {
    @NotNull(message = "PµPsP»Pµ PSPµ PrPsP»P¶PSPs P±C‹C,CЊ PïCѓCЃC,C‹Pj")
    @PositiveOrZero(message = "P”PsP»P¶PSPs P±C‹C,CЊ PSPµ
PsC,CЪPëC†P°C,PµP»CЊPSC‹Pj")
    private int id;
    @NotBlank(message = "PµPsP»Pµ PSPµ PrPsP»P¶PSPs P±C‹C,CЊ PïCѓCЃC,C‹Pj")
    private String type;
    @NotBlank(message = "PµPsP»Pµ PSPµ PrPsP»P¶PSPs P±C‹C,CЊ PïCѓCЃC,C‹Pj")
    private String model;
    @NotBlank(message = "PµPsP»Pµ PSPµ PrPsP»P¶PSPs P±C‹C,CЊ PïCѓCЃC,C‹Pj")
    private String manufacturer;
```

```
    @NotNull(message = "PµPsP»Pµ PSPµ PrPsP»P¶PSPs P±C‹C,CЊ PïCґCЃC,C‹Pj")
    @Positive(message = "P”PsP»P¶PSPs P±C‹C,CЊ PïPsP»PsP¶PëC,PµP»CЊPSC‹Pj")
    private BigDecimal cost;
    @NotNull(message = "PµPsP»Pµ PSPµ PrPsP»P¶PSPs P±C‹C,CЊ PïCґCЃC,C‹Pj")
    @Min(value = 10, message = "PќPµ PrPsP»P¶PSPs P±C‹C,CЊ PjPµPSPµPµ 10")
    @Max(value = 1000, message = "PќPµ PrPsP»P¶PSPs P±C‹C,CЊ P±PsP»PµPµ 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 11 – Lab05\src\main\java\com\github\prekel\JavaSpring\Lab05\form\IdForm.java

```
      package com.github.prekel.JavaSpring.Lab05.form;

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

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

    public int getId() {
        return id;
    }

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

Листинг 12 – Lab05\src\main\java\com\github\prekel\JavaSpring\Lab05\form\TypeForm.java

```
      package com.github.prekel.JavaSpring.Lab05.form;

import javax.validation.constraints.NotBlank;

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

    public String getType() {
        return type;
    }

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

Листинг 13 – Lab05\src\main\java\com\github\prekel\JavaSpring\Lab05\Program.java

```
      package com.github.prekel.JavaSpring.Lab05;

import com.github.prekel.JavaSpring.Lab05.data.FurnitureDao;
import com.github.prekel.JavaSpring.Lab05.data.FurnitureJdbcDao;
```

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.boot.autoconfigure.jdbc.DataSourceAutoConfiguration;
import
org.springframework.context.annotation.AnnotationConfigApplicationContext;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.ComponentScan;
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;

@SpringBootApplication(exclude = {DataSourceAutoConfiguration.class})
@ComponentScan("com.github.prekel.JavaSpring.Lab05.controller")
@PropertySource("classpath:application.properties")
@EnableTransactionManagement
@EnableJpaRepositories
public class Program {
    private static final Logger LOG = LoggerFactory.getLogger(Program.class);
    @Autowired
    private Environment env;

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

        LOG.info("Ended");
    }

    @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.Lab05.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 14 – Lab05\src\main\java\com\github\prekel\JavaSpring\Lab05\ReadWithCheckBuilder.java

```
      package com.github.prekel.JavaSpring.Lab05;

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 РїСЂРѕРjРµР¶СѓС,РєРµ РёР»Рё
РїСѓСЃС‚С,Р°СЏ СѓС,СЂРѕРєР°");
                }
                return parsed;
            } catch (Exception e) {
                System.err.println(e.getMessage());
            }
        }
    }
}
```

Листинг 15 – Lab05\src\main\java\com\github\prekel\JavaSpring\Lab05\SpringConfig.java

```
      package com.github.prekel.JavaSpring.Lab05;

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("springConfig")
@ComponentScan("com.github.prekel.JavaSpring.Lab05.component")
@PropertySource("classpath:application.properties")
@EnableTransactionManagement
@EnableJpaRepositories
public class SpringConfig {
    @Autowired
    private Environment env;

    @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.Lab05.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 16 – Lab05\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
```

Листинг 17 – Lab05\src\main\resources\templates\add.html

```
      <!DOCTYPE HTML>
<html xmlns:th="https://www.thymeleaf.org" lang="ru">
<head>
    <title>Р”РѕРґР°РІРёС‚СЊ РјРµР±РµР»СЊ</title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
</head>
<body>
<header>
    <h1>Р”РѕРґР°РІРёС‚СЊ РјРµР±РµР»СЊ</h1>
</header>
<article>
    <form class="form" th:object="${furnitureForm}" method="post">
        <div>
            <p>РўРёРї</p>
            <label>
                <input type="text" name="type" th:field="*{type}"
placeholder="type"/>
            </label>
```

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

Листинг 18 – Lab05\src\main\resources\templates\index.html

```
      <!DOCTYPE HTML>
<html xmlns:th="https://www.thymeleaf.org" lang="ru">
<head>
    <title>РњPµP±PµP»CЊ</title>
```

```
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
</head>
<body>
<header>
    <h1>РњРµР±РµР»Ь</h1>
</header>
<article>
    <p th:text="'РњРµР±РµР»Ё PI РєР°С,Р°Р»PsPiPµ: ' + ${count}">РњРµР±РµР»Ё PI
РєР°С,Р°Р»PsPiPµ:</p>
    <ul>
        <li><a href="/add">Р”PsР±Р°PIPёС,Сњ PjPµР±РµР»Сњ</a></li>
        <li><a href="/edit">Р РµРґР°РєС,РёСЂPsPIP°С,Сњ PjPµР±РµР»Сњ</a></li>
        <li><a href="/delete">РJРґР°Р»РёС,Сњ PjPµР±РµР»Сњ</a></li>
        <li><a href="/view">РµPsСЃPjPsС,СЂРµС,Сњ PjPµР±РµР»Сњ</a></li>
    </ul>
</article>
<footer>
    <hr>
    <a href="/">Р”PsPjPsP№</a>
</footer>
</body>
</html>
```

Листинг 19 – Lab05\src\main\webapp\index.jsp

```
      <html>
<body>
<h2>Hello World!</h2>
</body>
</html>
```

Листинг 20 – Lab05\src\main\webapp\WEB-INF\web.xml

```
      <!DOCTYPE web-app PUBLIC
 "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
 "http://java.sun.com/dtd/web-app_2_3.dtd" >

<web-app>
  <display-name>Archetype Created Web Application</display-name>
</web-app>
```