**ПРИЛОЖЕНИЕ А**

***(обязательное)***

**Листинг кода**

ФайлEmployeeListResultSetExtractor.java

package by.bsuir.phoneshop.core.dao.extractors;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import org.springframework.dao.DataAccessException;

import org.springframework.jdbc.core.ResultSetExtractor;

import org.springframework.stereotype.Component;

import by.bsuir.phoneshop.core.models.Employee;

import by.bsuir.phoneshop.core.models.enums.EmployeeType;

@Component

public class EmployeeListResultSetExtractor implements ResultSetExtractor<List<Employee>>

{

@Override

public List<Employee> extractData(final ResultSet resultSet) throws SQLException, DataAccessException

{

final List<Employee> employees = new ArrayList<>();

while (resultSet.next())

{

final Employee employee = new Employee();

employee.setId(resultSet.getLong("id"));

employee.setFirstName(resultSet.getString("firstName"));

employee.setLastName(resultSet.getString("lastName"));

employee.setPosition(resultSet.getString("position"));

employee.setEmployeeType(EmployeeType.valueOf(resultSet.getString("employeeType")));

employees.add(employee);

}

return employees;

}

}

Файл OrderListResultSetExtractor.java

package by.bsuir.phoneshop.core.dao.extractors;

import java.math.BigDecimal;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Timestamp;

import java.util.ArrayList;

import java.util.LinkedHashMap;

import java.util.List;

import java.util.Map;

import javax.annotation.Resource;

import org.springframework.dao.DataAccessException;

import org.springframework.jdbc.core.ResultSetExtractor;

import org.springframework.stereotype.Component;

import by.bsuir.phoneshop.core.models.Order;

import by.bsuir.phoneshop.core.models.OrderItem;

import by.bsuir.phoneshop.core.dao.PhoneDao;

import by.bsuir.phoneshop.core.models.enums.OrderStatus;

@Component

public class OrderListResultSetExtractor implements ResultSetExtractor<List<Order>>

{

@Resource

private PhoneDao jdbcPhoneDao;

@Override

public List<Order> extractData(final ResultSet resultSet) throws SQLException, DataAccessException

{

final Map<Order, List<OrderItem>> data = new LinkedHashMap<>();

final List<Order> orders = new ArrayList<>();

while (resultSet.next())

{

final Order order = configureOrderFields(resultSet);

final Long orderItemId = resultSet.getLong("orderItems.id");

final Long phoneId = resultSet.getLong("orderItems.phoneId");

final Long quantity = resultSet.getLong("orderItems.quantity");

data.putIfAbsent(order, new ArrayList<>());

OrderItem orderItem = new OrderItem(orderItemId, jdbcPhoneDao.get(phoneId).orElse(null), order, quantity);

data.get(order).add(orderItem);

}

for (Map.Entry<Order, List<OrderItem>> entry : data.entrySet())

{

final Order orderToAdd = entry.getKey();

orderToAdd.setOrderItems(entry.getValue());

orders.add(orderToAdd);

}

return orders;

}

private Order configureOrderFields(final ResultSet resultSet) throws SQLException

{

final Order order = new Order();

order.setId(resultSet.getLong("orders.id"));

order.setSubtotal(BigDecimal.valueOf(resultSet.getFloat("orders.subtotal")));

order.setDeliveryPrice(BigDecimal.valueOf(resultSet.getFloat("orders.deliveryPrice")));

order.setTotalPrice(BigDecimal.valueOf(resultSet.getFloat("orders.totalPrice")));

order.setFirstName(resultSet.getString("orders.firstName"));

order.setLastName(resultSet.getString("orders.lastName"));

order.setDeliveryAddress(resultSet.getString("orders.deliveryAddress"));

order.setContactPhoneNo(resultSet.getString("orders.contactPhoneNo"));

order.setAdditionalInfo(resultSet.getString("orders.additionalInfo"));

order.setStatus(OrderStatus.valueOf(resultSet.getString("orders.status")));

order.setDate(resultSet.getObject("orders.date", Timestamp.class));

return order;

}

}

Файл OrderResultSetExtractor.java

package by.bsuir.phoneshop.core.dao.extractors;

import java.math.BigDecimal;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Timestamp;

import java.util.ArrayList;

import java.util.List;

import javax.annotation.Resource;

import org.springframework.dao.DataAccessException;

import org.springframework.jdbc.core.ResultSetExtractor;

import org.springframework.stereotype.Component;

import by.bsuir.phoneshop.core.models.Order;

import by.bsuir.phoneshop.core.models.OrderItem;

import by.bsuir.phoneshop.core.dao.PhoneDao;

import by.bsuir.phoneshop.core.models.enums.OrderStatus;

@Component

public class OrderResultSetExtractor implements ResultSetExtractor<Order>

{

@Resource

private PhoneDao jdbcPhoneDao;

@Override

public Order extractData(final ResultSet resultSet) throws SQLException, DataAccessException

{

resultSet.next();

List<OrderItem> orderItems = new ArrayList<>();

final Order order = configureOrderFields(resultSet);

do

{

final Long orderItemId = resultSet.getLong("orderItems.id");

final Long phoneId = resultSet.getLong("orderItems.phoneId");

final Long quantity = resultSet.getLong("orderItems.quantity");

orderItems.add(new OrderItem(orderItemId, jdbcPhoneDao.get(phoneId).orElse(null), order, quantity));

} while ((resultSet.next()));

order.setOrderItems(orderItems);

return order;

}

private Order configureOrderFields(final ResultSet resultSet) throws SQLException

{

final Order order = new Order();

order.setId(resultSet.getLong("orders.id"));

order.setSubtotal(BigDecimal.valueOf(resultSet.getFloat("orders.subtotal")));

order.setDeliveryPrice(BigDecimal.valueOf(resultSet.getFloat("orders.deliveryPrice")));

order.setTotalPrice(BigDecimal.valueOf(resultSet.getFloat("orders.totalPrice")));

order.setFirstName(resultSet.getString("orders.firstName"));

order.setLastName(resultSet.getString("orders.lastName"));

order.setDeliveryAddress(resultSet.getString("orders.deliveryAddress"));

order.setContactPhoneNo(resultSet.getString("orders.contactPhoneNo"));

order.setAdditionalInfo(resultSet.getString("orders.additionalInfo"));

order.setStatus(OrderStatus.valueOf(resultSet.getString("orders.status")));

order.setDate(resultSet.getObject("orders.date", Timestamp.class));

return order;

}

}

Файл PhoneResultSetExtractor.java

package by.bsuir.phoneshop.core.dao.extractors;

import java.math.BigDecimal;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.HashSet;

import java.util.LinkedHashMap;

import java.util.List;

import java.util.Map;

import java.util.Set;

import org.springframework.dao.DataAccessException;

import org.springframework.jdbc.core.ResultSetExtractor;

import org.springframework.stereotype.Component;

import by.bsuir.phoneshop.core.models.Color;

import by.bsuir.phoneshop.core.models.Phone;

@Component

public class PhoneResultSetExtractor implements ResultSetExtractor<List<Phone>>

{

@Override

public List<Phone> extractData(final ResultSet resultSet) throws SQLException, DataAccessException

{

final Map<Phone, Set<Color>> data = new LinkedHashMap<>();

while (resultSet.next())

{

final Phone phone = configurePhoneFields(resultSet);

data.putIfAbsent(phone, new HashSet<>());

final Color color = new Color();

color.setId(resultSet.getLong("colors.id"));

color.setCode(resultSet.getString("code"));

data.get(phone).add(color);

}

final List<Phone> phoneList = new ArrayList<>();

for (Map.Entry<Phone, Set<Color>> entry : data.entrySet())

{

final Phone phoneToAdd = entry.getKey();

phoneToAdd.setColors(entry.getValue());

phoneList.add(phoneToAdd);

}

return phoneList;

}

private Phone configurePhoneFields(final ResultSet resultSet) throws SQLException

{

final Phone phone = new Phone();

phone.setId(resultSet.getLong("id"));

phone.setBrand(resultSet.getString("brand"));

phone.setModel(resultSet.getString("model"));

phone.setPrice(BigDecimal.valueOf(resultSet.getLong("price")));

phone.setDiscountPercent(resultSet.getInt("discountPercent"));

phone.setActualPrice();

phone.setDisplaySizeInches(BigDecimal.valueOf(resultSet.getLong("displaySizeInches")));

phone.setWeightGr(resultSet.getInt("weightGr"));

phone.setWidthMm(BigDecimal.valueOf(resultSet.getFloat("lengthMm")));

phone.setHeightMm(BigDecimal.valueOf(resultSet.getFloat("heightMm")));

phone.setDeviceType(resultSet.getString("deviceType"));

phone.setDisplayResolution(resultSet.getString("displayResolution"));

phone.setPixelDensity(resultSet.getInt("pixelDensity"));

phone.setDisplayTechnology(resultSet.getString("displayTechnology"));

phone.setBackCameraMegapixels(BigDecimal.valueOf(resultSet.getFloat("backCameraMegapixels")));

phone.setFrontCameraMegapixels(BigDecimal.valueOf(resultSet.getFloat("frontCameraMegapixels")));

phone.setInternalStorageGb(BigDecimal.valueOf(resultSet.getFloat("internalStorageGb")));

phone.setBatteryCapacityMah(resultSet.getInt("batteryCapacityMah"));

phone.setTalkTimeHours(BigDecimal.valueOf(resultSet.getFloat("talkTimeHours")));

phone.setStandByTimeHours(BigDecimal.valueOf(resultSet.getFloat("standByTimeHours")));

phone.setBluetooth(resultSet.getString("bluetooth"));

phone.setImageUrl(resultSet.getString("imageUrl"));

phone.setDescription(resultSet.getString("description"));

return phone;

}

}

Файл UserListResultSetExtractor.java

package by.bsuir.phoneshop.core.dao.extractors;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import org.springframework.dao.DataAccessException;

import org.springframework.jdbc.core.ResultSetExtractor;

import org.springframework.stereotype.Component;

import by.bsuir.phoneshop.core.models.User;

@Component

public class UserListResultSetExtractor implements ResultSetExtractor<List<User>>

{

@Override

public List<User> extractData(final ResultSet resultSet) throws SQLException, DataAccessException

{

final List<User> users = new ArrayList<>();

while (resultSet.next())

{

final User user = new User();

user.setUserName(resultSet.getString("userName"));

user.setPassword(resultSet.getString("password"));

user.setRole(resultSet.getString("role"));

user.setIsAccountNonLocked(resultSet.getInt("isAccountNonLocked"));

users.add(user);

}

return users;

}

}

Файл JdbcColorDao.java

package by.bsuir.phoneshop.core.dao.impl;

import java.util.Optional;

import javax.annotation.Resource;

import org.springframework.jdbc.core.BeanPropertyRowMapper;

import org.springframework.jdbc.core.JdbcTemplate;

import org.springframework.stereotype.Component;

import by.bsuir.phoneshop.core.dao.ColorDAO;

import by.bsuir.phoneshop.core.models.Color;

@Component

public class JdbcColorDAO implements ColorDAO

{

@Resource

private JdbcTemplate jdbcTemplate;

private static final String SQL\_GET\_COLOR = "select \* from colors where id = ";

private static final String SQL\_UPDATE\_COLOR = "insert into colors(id,code) values (?,?)";

@Override

public Optional<Color> get(final Long key)

{

return Optional.ofNullable(jdbcTemplate.queryForObject(SQL\_GET\_COLOR + key,

new BeanPropertyRowMapper<Color>(Color.class)));

}

@Override

public void save(final Color color)

{

jdbcTemplate.update(SQL\_UPDATE\_COLOR, color.getId(), color.getCode());

}

}

Файл JdbcOrderDao.java

package by.bsuir.phoneshop.core.dao.impl;

import java.util.List;

import java.util.Optional;

import javax.annotation.Resource;

import org.springframework.jdbc.core.JdbcTemplate;

import org.springframework.jdbc.core.namedparam.MapSqlParameterSource;

import org.springframework.jdbc.core.namedparam.NamedParameterJdbcTemplate;

import org.springframework.stereotype.Component;

import by.bsuir.phoneshop.core.dao.OrderDao;

import by.bsuir.phoneshop.core.models.Order;

import by.bsuir.phoneshop.core.models.OrderItem;

import by.bsuir.phoneshop.core.dao.extractors.OrderListResultSetExtractor;

import by.bsuir.phoneshop.core.dao.extractors.OrderResultSetExtractor;

import by.bsuir.phoneshop.core.models.enums.OrderStatus;

@Component

public class JdbcOrderDao implements OrderDao

{

public static final String SELECT\_ORDER\_SQL = "select \* from orders left join orderItems on orders.id = orderItems.orderId where orders.id= ";

public static final String INSERT\_ORDER\_SQL = " insert into orders(id,subtotal,deliveryPrice,totalPrice,firstName," +

"lastName,deliveryAddress,contactPhoneNo,additionalInfo,status,date) values (:id,:subtotal,:deliveryPrice," +

":totalPrice,:firstName,:lastName,:deliveryAddress,:contactPhoneNo,:additionalInfo,:status,:date)";

public static final String SQL\_INSERT\_INTO\_ORDER\_ITEMS = "insert into orderItems (phoneId,orderId,quantity) " +

"values (?,?,?)";

public static final String SELECT\_ORDERS = "select \* from orders left join orderItems on orders.id = orderItems.orderId";

public static final String UPDATE\_ORDER\_STATUS\_SQL = "update orders set orders.status = '";

public static final String SQL\_NEEDED\_ORDER = "' where orders.id = ";

@Resource

private NamedParameterJdbcTemplate namedParameterJdbcTemplate;

@Resource

private JdbcTemplate jdbcTemplate;

@Resource

private OrderResultSetExtractor orderResultSetExtractor;

@Resource

private OrderListResultSetExtractor orderListResultSetExtractor;

@Override

public Optional<Order> get(final Long key)

{

String query = SELECT\_ORDER\_SQL + key;

return Optional.ofNullable(jdbcTemplate.query(query, orderResultSetExtractor));

}

@Override

public Long save(final Order order)

{

MapSqlParameterSource in = new MapSqlParameterSource();

in.addValue("id", order.getId());

in.addValue("subtotal", order.getSubtotal());

in.addValue("deliveryPrice", order.getDeliveryPrice());

in.addValue("totalPrice", order.getTotalPrice());

in.addValue("firstName", order.getFirstName());

in.addValue("lastName", order.getLastName());

in.addValue("deliveryAddress", order.getDeliveryAddress());

in.addValue("contactPhoneNo", order.getContactPhoneNo());

in.addValue("additionalInfo", order.getAdditionalInfo());

in.addValue("status", order.getStatus().name());

in.addValue("date", order.getDate());

namedParameterJdbcTemplate.update(INSERT\_ORDER\_SQL, in);

List<OrderItem> orderItemList = order.getOrderItems();

orderItemList.forEach(orderItem -> jdbcTemplate

.update(SQL\_INSERT\_INTO\_ORDER\_ITEMS, orderItem.getPhone().getId(), order.getId(), orderItem.getQuantity()));

return order.getId();

}

@Override

public List<Order> getOrders(final int limit, final int offset)

{

String query = SELECT\_ORDERS + " order by orders.date DESC " + " limit " + limit + " offset " + offset;

return jdbcTemplate.query(query, orderListResultSetExtractor);

}

@Override

public void updateStatus(OrderStatus orderStatus, Long key)

{

String query = UPDATE\_ORDER\_STATUS\_SQL + orderStatus.name() + SQL\_NEEDED\_ORDER + key;

jdbcTemplate.update(query);

}

}

Файл JdbcPhoneDao.java

package by.bsuir.phoneshop.core.dao.impl;

import java.sql.Types;

import java.util.ArrayList;

import java.util.HashSet;

import java.util.List;

import java.util.Optional;

import java.util.Set;

import javax.annotation.Resource;

import org.springframework.dao.EmptyResultDataAccessException;

import org.springframework.jdbc.core.BeanPropertyRowMapper;

import org.springframework.jdbc.core.JdbcTemplate;

import org.springframework.jdbc.core.RowMapper;

import org.springframework.jdbc.core.namedparam.MapSqlParameterSource;

import org.springframework.jdbc.core.namedparam.NamedParameterJdbcTemplate;

import org.springframework.stereotype.Component;

import org.springframework.transaction.annotation.Transactional;

import by.bsuir.phoneshop.core.models.Color;

import by.bsuir.phoneshop.core.models.ParamsForSearch;

import by.bsuir.phoneshop.core.models.Phone;

import by.bsuir.phoneshop.core.dao.ColorDAO;

import by.bsuir.phoneshop.core.dao.PhoneDao;

import by.bsuir.phoneshop.core.dao.extractors.PhoneResultSetExtractor;

@Component

public class JdbcPhoneDao implements PhoneDao

{

private static final String SQL\_INSERT\_PHONE = "insert into phones (id,brand,model,price,discountPercent,displaySizeInches,weightGr," +

"lengthMm,widthMm,heightMm,deviceType,displayResolution,pixelDensity,displayTechnology," +

"backCameraMegapixels,frontCameraMegapixels,internalStorageGb,batteryCapacityMah,talkTimeHours," +

"standByTimeHours,bluetooth,imageUrl,description) values (:id,:brand,:model,:price,:discountPercent,:displaySizeInches,:weightGr," +

":lengthMm,:widthMm,:heightMm,:deviceType,:displayResolution,:pixelDensity," +

":displayTechnology,:backCameraMegapixels,:frontCameraMegapixels,:internalStorageGb," +

":batteryCapacityMah,:talkTimeHours,:standByTimeHours,:bluetooth,:imageUrl,:description)";

private static final String SQL\_GET\_ALL\_PHONES = "select \* from phones left join phone2color " +

"on phones.id=phone2color.phoneId left join colors on colors.id = phone2color.colorId ";

private static final String SQL\_GET\_MAX\_DISCOUNTED\_PHONES = "select \* from phones left join phone2color " +

"on phones.id=phone2color.phoneId left join colors on colors.id = phone2color.colorId where phones.discountPercent!=0 ORDER BY phones.discountPercent DESC";

public static final String SQL\_GET\_PHONE = "select \* from phones where id= ";

private static final String SQL\_SELECT\_COUNT\_FIND\_ALL\_EXTENDED = "select count(\*) from phones ";

private static final String SQL\_WHERE\_SEARCH = "where (phones.id in (select phoneId from stocks) and " +

"(SELECT STOCK FROM STOCKS WHERE PHONEID = PHONES.ID AND STOCK > 0) and price is not null) ";

public static final String SQL\_SELECT\_COLOR\_IDS = "select colorId from phone2color where phoneId= ";

public static final String SQL\_GET\_PHONE\_BY\_MODEL = "select \* from phones where phones.model = '";

public static final String DELETE\_PHONE\_SQL = "delete from phones where phones.id=?";

public static final String SQL\_UPDATE\_PHONE = "update phones set brand= :brand,model= :model,price= :price,discountPercent= :discountPercent," +

"displaySizeInches= :displaySizeInches,weightGr= :weightGr,lengthMm= :lengthMm,widthMm= :widthMm,heightMm= :heightMm,deviceType= :deviceType,displayResolution= :displayResolution,pixelDensity= :pixelDensity," +

"displayTechnology= :displayTechnology,backCameraMegapixels= :backCameraMegapixels,frontCameraMegapixels= :frontCameraMegapixels,internalStorageGb= :internalStorageGb,batteryCapacityMah= :batteryCapacityMah," +

"talkTimeHours= :talkTimeHours,standByTimeHours= :standByTimeHours,bluetooth= :bluetooth,imageUrl= :imageUrl,description= :description where id= :id";

@Resource

private JdbcTemplate jdbcTemplate;

@Resource

private NamedParameterJdbcTemplate namedParameterJdbcTemplate;

@Resource

private PhoneResultSetExtractor phoneResultSetExtractor;

@Resource

private ColorDAO jdbcColorDAO;

@Override

public Optional<Phone> get(final Long key)

{

final String query = SQL\_GET\_PHONE + key;

return getPhoneAndColors(query);

}

@Override

public Optional<Phone> get(final String model)

{

final String query = SQL\_GET\_PHONE\_BY\_MODEL + model + "'";

return getPhoneAndColors(query);

}

private Optional<Phone> getPhoneAndColors(final String query)

{

Phone phone;

try

{

phone = jdbcTemplate.queryForObject(query, new BeanPropertyRowMapper<>(Phone.class));

phone.setActualPrice();

}

catch (EmptyResultDataAccessException e)

{

return Optional.empty();

}

final RowMapper<Long> idRowMapper = (rs, rowNum) -> rs.getLong("colorId");

final String colorIdQuery = SQL\_SELECT\_COLOR\_IDS + phone.getId();

final List<Long> colorIds = jdbcTemplate.query(colorIdQuery, idRowMapper);

if (colorIds != null && !colorIds.isEmpty())

{

final Set<Color> colorSet = new HashSet<>();

for (Long colorId : colorIds)

{

final Optional<Color> colorOptional = jdbcColorDAO.get(colorId);

colorOptional.ifPresent(colorSet::add);

}

phone.setColors(colorSet);

}

return Optional.ofNullable(phone);

}

@Override

@Transactional

public void save(final Phone phone)

{

namedParameterJdbcTemplate.update(SQL\_INSERT\_PHONE, populateNamedParams(phone));

}

@Override

@Transactional(readOnly = true)

public List<Phone> findAll(final int offset, final int limit)

{

String query = SQL\_GET\_ALL\_PHONES + " offset " + offset + " limit " + limit;

return jdbcTemplate.query(query, phoneResultSetExtractor);

}

@Override

@Transactional(readOnly = true)

public List<Phone> findAll(final ParamsForSearch paramsForSearch)

{

final String search = paramsForSearch.getSearch();

final String sortField = paramsForSearch.getSortField();

final String order = paramsForSearch.getOrder();

final int offset = paramsForSearch.getOffset();

final int limit = paramsForSearch.getLimit();

final List<Object> objects = new ArrayList<>();

final List<Integer> types = new ArrayList<>();

String query = SQL\_GET\_ALL\_PHONES + SQL\_WHERE\_SEARCH;

if (search != null && !search.isEmpty())

{

query = query + "and lower(model) like lower(?) ";

objects.add("%" + search + "%");

types.add(Types.VARCHAR);

}

if (sortField != null && order != null)

{

query = query + String.format("group by phones.id,phone2color.colorId order by %s %s ", sortField, order);

}

query = query + " limit " + limit + " offset " + offset;

final int[] typesArray = types.stream()

.mapToInt(i -> i)

.toArray();

return jdbcTemplate.query(query, objects.toArray(), typesArray, phoneResultSetExtractor);

}

@Override

public Long count(final ParamsForSearch paramsForSearch)

{

final String search = paramsForSearch.getSearch();

String request = SQL\_SELECT\_COUNT\_FIND\_ALL\_EXTENDED + SQL\_WHERE\_SEARCH;

if (search != null && !search.isEmpty())

{

request = request + "and (lower(model) like lower(?))";

return jdbcTemplate.queryForObject(request, new Object[]{"%" + search +

"%"}, new int[]{Types.VARCHAR}, Long.class);

}

else

{

return jdbcTemplate.queryForObject(request, Long.class);

}

}

@Override

public List<Phone> findMaxDiscountPercentPhones()

{

return jdbcTemplate.query(SQL\_GET\_MAX\_DISCOUNTED\_PHONES, phoneResultSetExtractor);

}

@Override

public void deletePhone(final Long id)

{

jdbcTemplate.update(DELETE\_PHONE\_SQL, id);

}

@Override

@Transactional

public void updateDevice(final Phone phone)

{

namedParameterJdbcTemplate.update(SQL\_UPDATE\_PHONE, populateNamedParams(phone));

}

private MapSqlParameterSource populateNamedParams(final Phone phone)

{

MapSqlParameterSource in = new MapSqlParameterSource();

in.addValue("id", phone.getId());

in.addValue("brand", phone.getBrand());

in.addValue("model", phone.getModel());

in.addValue("price", phone.getPrice());

in.addValue("discountPercent", phone.getDiscountPercent());

in.addValue("displaySizeInches", phone.getDisplaySizeInches());

in.addValue("weightGr", phone.getWeightGr());

in.addValue("lengthMm", phone.getLengthMm());

in.addValue("widthMm", phone.getWidthMm());

in.addValue("heightMm", phone.getHeightMm());

in.addValue("deviceType", phone.getDeviceType());

in.addValue("displayResolution", phone.getDisplayResolution());

in.addValue("pixelDensity", phone.getPixelDensity());

in.addValue("displayTechnology", phone.getDisplayTechnology());

in.addValue("backCameraMegapixels", phone.getBackCameraMegapixels());

in.addValue("frontCameraMegapixels", phone.getFrontCameraMegapixels());

in.addValue("internalStorageGb", phone.getInternalStorageGb());

in.addValue("batteryCapacityMah", phone.getBatteryCapacityMah());

in.addValue("talkTimeHours", phone.getTalkTimeHours());

in.addValue("standByTimeHours", phone.getStandByTimeHours());

in.addValue("bluetooth", phone.getBluetooth());

in.addValue("imageUrl", phone.getImageUrl());

in.addValue("description", phone.getDescription());

return in;

}

}

Файл JdbcStockDao.java

package by.bsuir.phoneshop.core.dao.impl;

import java.util.List;

import java.util.Optional;

import javax.annotation.Resource;

import org.springframework.jdbc.core.BeanPropertyRowMapper;

import org.springframework.jdbc.core.JdbcTemplate;

import org.springframework.stereotype.Component;

import by.bsuir.phoneshop.core.models.Stock;

import by.bsuir.phoneshop.core.dao.StockDao;

@Component

public class JdbcStockDao implements StockDao

{

@Resource

private JdbcTemplate jdbcTemplate;

private static final String SQL\_GET\_STOCK = "select \* from stocks where phoneId = ";

private static final String SQL\_UPDATE = "update stocks set stock = %d, reserved = %d where phoneId = %d";

private static final String SQL\_INSERT = "insert into stocks(phoneId,stock,reserved) values (?,?,?)";

@Override

public Optional<Stock> get(final Long key)

{

final List<Stock> stocks = jdbcTemplate.query(SQL\_GET\_STOCK + key, new BeanPropertyRowMapper<>(Stock.class));

return stocks.stream().findFirst();

}

@Override

public void update(final Long key, final Long stock, final Long reserved, boolean isNewPhone)

{

String query;

if (!isNewPhone)

{

query = String.format(SQL\_UPDATE, stock, reserved, key);

jdbcTemplate.update(query);

}

else

{

jdbcTemplate.update(SQL\_INSERT, key, stock, reserved);

}

}

}

Файл JdbcUserDao.java

package by.bsuir.phoneshop.core.dao.impl;

import java.util.List;

import java.util.Optional;

import javax.annotation.Resource;

import org.springframework.dao.EmptyResultDataAccessException;

import org.springframework.jdbc.core.BeanPropertyRowMapper;

import org.springframework.jdbc.core.JdbcTemplate;

import org.springframework.stereotype.Service;

import by.bsuir.phoneshop.core.models.Employee;

import by.bsuir.phoneshop.core.models.User;

import by.bsuir.phoneshop.core.dao.UserDao;

import by.bsuir.phoneshop.core.dao.extractors.EmployeeListResultSetExtractor;

import by.bsuir.phoneshop.core.dao.extractors.UserListResultSetExtractor;

@Service

public class JdbcUserDao implements UserDao

{

@Resource

private JdbcTemplate jdbcTemplate;

@Resource

private UserListResultSetExtractor resultSetExtractor;

@Resource

private EmployeeListResultSetExtractor employeeListResultSetExtractor;

private static final String SQL\_GET\_EMPLOYEES = "select \* from employees";

private static final String SQL\_GET\_USER = "select \* from users where userName=";

private static final String SQL\_GET\_USERS\_EXCEPT\_ADMIN = "select \* from users where userName not like 'admin'";

private static final String SQL\_UPDATE\_USER\_STATUS = "update users set isAccountNonLocked= ? where userName = ?";

private static final String SQL\_INSERT\_NEW\_USER = "insert into users(userName,password,isAccountNonLocked,role) values (?,?,?,?)";

@Override

public Optional<User> getUserByUsername(final String username) throws EmptyResultDataAccessException

{

return Optional.ofNullable(jdbcTemplate.queryForObject(SQL\_GET\_USER + "'" + username + "'", new BeanPropertyRowMapper<User>(User.class)));

}

@Override

public List<User> getUsersExceptAdmin(int limit, int offset)

{

return jdbcTemplate.query(SQL\_GET\_USERS\_EXCEPT\_ADMIN + " limit " + limit + " offset " + offset, resultSetExtractor);

}

@Override

public void updateUserStatus(int status, String username)

{

jdbcTemplate.update(SQL\_UPDATE\_USER\_STATUS, status, username);

}

@Override

public void addNewUserToDb(final User user)

{

jdbcTemplate.update(SQL\_INSERT\_NEW\_USER, user.getUserName(), user.getPassword(), user.getIsAccountNonLocked(), user.getRole());

}

@Override

public List<Employee> getMainEmployees()

{

return jdbcTemplate.query(SQL\_GET\_EMPLOYEES, employeeListResultSetExtractor);

}

}

Файл ColorDao.java

package by.bsuir.phoneshop.core.dao;

import java.util.Optional;

import by.bsuir.phoneshop.core.models.Color;

public interface ColorDAO

{

Optional<Color> get(final Long key);

void save(final Color color);

}

Файл OrderDao.java

package by.bsuir.phoneshop.core.dao;

import java.util.List;

import java.util.Optional;

import by.bsuir.phoneshop.core.models.Order;

import by.bsuir.phoneshop.core.models.enums.OrderStatus;

public interface OrderDao

{

Optional<Order> get(final Long key);

Long save(final Order order);

List<Order> getOrders(final int limit, final int offset);

void updateStatus(final OrderStatus orderStatus, final Long key);

}

Файл PhoneDao.java

package by.bsuir.phoneshop.core.dao;

import java.util.List;

import java.util.Optional;

import by.bsuir.phoneshop.core.models.ParamsForSearch;

import by.bsuir.phoneshop.core.models.Phone;

public interface PhoneDao

{

Optional<Phone> get(final Long key);

Optional<Phone> get(final String model);

void save(final Phone phone);

List<Phone> findAll(final int offset, final int limit);

List<Phone> findAll(final ParamsForSearch paramsForSearch);

Long count(final ParamsForSearch paramsForSearch);

List<Phone> findMaxDiscountPercentPhones();

void deletePhone(final Long id);

void updateDevice(final Phone phone);

}

Файл StockDao.java

package by.bsuir.phoneshop.core.dao;

import java.util.Optional;

import by.bsuir.phoneshop.core.models.Stock;

public interface StockDao

{

Optional<Stock> get(final Long key);

void update(final Long key, final Long stock,final Long reserved,final boolean isNewPhone);

}

Файл UserDao.java

package by.bsuir.phoneshop.core.dao;

import java.util.List;

import java.util.Optional;

import by.bsuir.phoneshop.core.models.Employee;

import by.bsuir.phoneshop.core.models.User;

public interface UserDao

{

Optional<User> getUserByUsername(final String username);

List<User> getUsersExceptAdmin(int limit, int offset);

void updateUserStatus(final int status,final String username);

void addNewUserToDb(final User user);

List<Employee> getMainEmployees();

}

Файл OrderDataDto.java

package by.bsuir.phoneshop.core.dto;

public class OrderDataDto

{

private String firstName;

private String lastName;

private String address;

private String phone;

private String additionalInfo;

public String getFirstName()

{

return firstName;

}

public void setFirstName(final String firstName)

{

this.firstName = firstName;

}

public String getLastName()

{

return lastName;

}

public void setLastName(final String lastName)

{

this.lastName = lastName;

}

public String getAddress()

{

return address;

}

public void setAddress(final String address)

{

this.address = address;

}

public String getPhone()

{

return phone;

}

public void setPhone(final String phone)

{

this.phone = phone;

}

public String getAdditionalInfo()

{

return additionalInfo;

}

public void setAdditionalInfo(final String additionalInfo)

{

this.additionalInfo = additionalInfo;

}

}

Файл PhoneAddDto.java

package by.bsuir.phoneshop.core.dto;

import java.math.BigDecimal;

import lombok.Getter;

import lombok.Setter;

@Getter

@Setter

public class PhoneAddDto

{

private String brand;

private String model;

private BigDecimal price;

private Integer discountPercent;

private String imageUrl;

private Long quantity;

private String description;

private BigDecimal displaySizeInches;

private String displayResolution;

private Integer pixelDensity;

private String displayTechnology;

private Integer weightGr;

private BigDecimal lengthMm;

private BigDecimal widthMm;

private String colors;

private BigDecimal heightMm;

private String deviceType;

private BigDecimal backCameraMegapixels;

private BigDecimal frontCameraMegapixels;

private BigDecimal internalStorageGb;

private Integer batteryCapacityMah;

private BigDecimal talkTimeHours;

private BigDecimal standByTimeHours;

private String bluetooth;

}

Файл PhoneArrayDto.java

package by.bsuir.phoneshop.core.dto;

public class PhoneArrayDto

{

private String[] phoneId;

private String[] quantity;

public String[] getPhoneId()

{

return phoneId;

}

public void setPhoneId(final String[] phoneId)

{

this.phoneId = phoneId;

}

public String[] getQuantity()

{

return quantity;

}

public void setQuantity(final String[] quantity)

{

this.quantity = quantity;

}

}

Файл PhoneDto.java

package by.bsuir.phoneshop.core.dto;

public class PhoneDto

{

private Long id;

private Long quantity;

public Long getId(){ return id;}

public void setId(final Long id)

{

this.id = id;

}

public Long getQuantity()

{

return quantity;

}

public void setQuantity(final Long quantity)

{

this.quantity = quantity;

}

}

Файл QuickOrderElement.java

package by.bsuir.phoneshop.core.dto;

public class QuickOrderElement

{

private String model;

private Long quantity;

public String getModel()

{

return model;

}

public void setModel(final String model)

{

this.model = model;

}

public Long getQuantity()

{

return quantity;

}

public void setQuantity(final Long quantity)

{

this.quantity = quantity;

}

}

Файл QuickOrderElementsDto.java

package by.bsuir.phoneshop.core.dto;

import java.util.ArrayList;

import java.util.List;

public class QuickOrderElementsDto

{

private List<QuickOrderElement> quickOrderElements;

public QuickOrderElementsDto()

{

this.quickOrderElements = new ArrayList<QuickOrderElement>();

}

public List<QuickOrderElement> getQuickOrderElements()

{

return quickOrderElements;

}

public void setQuickOrderElements(final List<QuickOrderElement> quickOrderElements)

{

this.quickOrderElements = quickOrderElements;

}

}

Файл UserDto.java

package by.bsuir.phoneshop.core.dto;

public class UserDto

{

private String userName;

private Integer status;

public String getUserName()

{

return userName;

}

public void setUserName(String userName)

{

this.userName = userName;

}

public Integer getStatus()

{

return status;

}

public void setStatus(Integer status)

{

this.status = status;

}

}

Файл UserRegisterDto.java

package by.bsuir.phoneshop.core.dto;

public class UserRegisterDto

{

private String username;

private 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;

}

}

Файл ResponseErrors.java

package by.bsuir.phoneshop.core.exception.errors;

public class ResponseErrors

{

private String errorMessage;

public ResponseErrors(final String errorMessage)

{

this.errorMessage = errorMessage;

}

public String getErrorsMessage()

{

return errorMessage;

}

public void setErrorsMessage(final String errorMessage)

{

this.errorMessage = errorMessage;

}

}

Файл ValidationErrors.java

package by.bsuir.phoneshop.core.exception.errors;

import java.util.List;

import org.springframework.validation.ObjectError;

public class ValidationErrors

{

private List<ObjectError> errors;

public ValidationErrors(final List<ObjectError> errors)

{

this.errors = errors;

}

public List<ObjectError> getErrors()

{

return errors;

}

public void setErrors(final List<ObjectError> errors)

{

this.errors = errors;

}

}

Файл EmployeeType.java

package by.bsuir.phoneshop.core.models.enums;

public enum EmployeeType

{

MALE,FEMALE, GEEK\_MALE

}

Файл OrderStatus.java

package by.bsuir.phoneshop.core.models.enums;

public enum OrderStatus

{

NEW, DELIVERED,IN\_DELIVERY, REJECTED

}

Файл SortField.java

package by.bsuir.phoneshop.core.models.enums;

public enum SortField

{

BRAND, MODEL, DISPLAYSIZEINCHES, PRICE

}

Файл SortOrder.java

package by.bsuir.phoneshop.core.models.enums;

public enum SortOrder

{

ASC, DESC

}

Файл Cart.java

package by.bsuir.phoneshop.core.models;

import java.math.BigDecimal;

import java.util.ArrayList;

import java.util.List;

public class Cart

{

private List<CartItem> cartItems;

private Long totalQuantity;

private BigDecimal totalCost;

public Cart()

{

cartItems = new ArrayList<>();

totalQuantity = 0L;

totalCost = BigDecimal.ZERO;

}

public List<CartItem> getCartItems()

{

return cartItems;

}

public void setCartItems(final List<CartItem> cartItems)

{

this.cartItems = cartItems;

}

public Long getTotalQuantity()

{

return totalQuantity;

}

public void setTotalQuantity(final Long totalQuantity)

{

this.totalQuantity = totalQuantity;

}

public BigDecimal getTotalCost()

{

return totalCost;

}

public void setTotalCost(final BigDecimal totalCost)

{

this.totalCost = totalCost;

}

}

Файл CartItem.java

package by.bsuir.phoneshop.core.models;

import java.math.BigDecimal;

public class CartItem

{

private Phone phone;

private Long quantity;

private BigDecimal price;

public CartItem(final Phone phone, final Long quantity, final BigDecimal price)

{

this.phone = phone;

this.quantity = quantity;

this.price = price;

}

public Phone getPhone()

{

return phone;

}

public void setPhone(final Phone phone)

{

this.phone = phone;

}

public Long getQuantity()

{

return quantity;

}

public void setQuantity(final Long quantity)

{

this.quantity = quantity;

}

public BigDecimal getPrice()

{

return price;

}

public void setPrice(final BigDecimal price)

{

this.price = price;

}

}

Файл Color.java

package by.bsuir.phoneshop.core.models;

public class Color

{

private Long id;

private String code;

public Long getId()

{

return id;

}

public void setId(final Long id)

{

this.id = id;

}

public String getCode()

{

return code;

}

public void setCode(final String code)

{

this.code = code;

}

}

Файл Employee.java

package by.bsuir.phoneshop.core.models;

import by.bsuir.phoneshop.core.models.enums.EmployeeType;

public class Employee

{

private Long id;

private String firstName;

private String lastName;

private String position;

private EmployeeType employeeType;

public Long getId()

{

return id;

}

public void setId(Long id)

{

this.id = id;

}

public String getFirstName()

{

return firstName;

}

public void setFirstName(String firstName)

{

this.firstName = firstName;

}

public String getLastName()

{

return lastName;

}

public void setLastName(String lastName)

{

this.lastName = lastName;

}

public String getPosition()

{

return position;

}

public void setPosition(String position)

{

this.position = position;

}

public EmployeeType getEmployeeType()

{

return employeeType;

}

public void setEmployeeType(EmployeeType employeeType)

{

this.employeeType = employeeType;

}

}

Файл Order.java

package by.bsuir.phoneshop.core.models;

import java.math.BigDecimal;

import java.sql.Timestamp;

import java.time.Instant;

import java.util.ArrayList;

import java.util.List;

import by.bsuir.phoneshop.core.models.enums.OrderStatus;

public class Order

{

private Long id;

private List<OrderItem> orderItems;

private BigDecimal subtotal;

private BigDecimal deliveryPrice;

private BigDecimal totalPrice;

private String firstName;

private String lastName;

private String deliveryAddress;

private String contactPhoneNo;

private String additionalInfo;

private Timestamp date;

private OrderStatus status;

public Order()

{

orderItems = new ArrayList<>();

subtotal = BigDecimal.ZERO;

totalPrice = BigDecimal.ZERO;

deliveryPrice = BigDecimal.ZERO;

date = Timestamp.from(Instant.now());

}

public Long getId()

{

return id;

}

public void setId(final Long id)

{

this.id = id;

}

public List<OrderItem> getOrderItems()

{

return orderItems;

}

public void setOrderItems(final List<OrderItem> orderItems)

{

this.orderItems = orderItems;

}

public BigDecimal getSubtotal()

{

return subtotal;

}

public void setSubtotal(final BigDecimal subtotal)

{

this.subtotal = subtotal;

}

public BigDecimal getDeliveryPrice()

{

return deliveryPrice;

}

public void setDeliveryPrice(final BigDecimal deliveryPrice)

{

this.deliveryPrice = deliveryPrice;

}

public BigDecimal getTotalPrice()

{

return totalPrice;

}

public void setTotalPrice(final BigDecimal totalPrice)

{

this.totalPrice = totalPrice;

}

public String getFirstName()

{

return firstName;

}

public void setFirstName(final String firstName)

{

this.firstName = firstName;

}

public String getLastName()

{

return lastName;

}

public void setLastName(final String lastName)

{

this.lastName = lastName;

}

public String getDeliveryAddress()

{

return deliveryAddress;

}

public void setDeliveryAddress(final String deliveryAddress)

{

this.deliveryAddress = deliveryAddress;

}

public String getContactPhoneNo()

{

return contactPhoneNo;

}

public void setContactPhoneNo(final String contactPhoneNo)

{

this.contactPhoneNo = contactPhoneNo;

}

public OrderStatus getStatus()

{

return status;

}

public Timestamp getDate()

{

return date;

}

public void setDate(final Timestamp date)

{

this.date = date;

}

public void setStatus(final OrderStatus status)

{

this.status = status;

}

public String getAdditionalInfo()

{

return additionalInfo;

}

public void setAdditionalInfo(final String additionalInfo)

{

this.additionalInfo = additionalInfo;

}

@Override

public int hashCode()

{

final int prime = 31;

int result = 1;

result = prime \* result + id.hashCode() + date.hashCode() + totalPrice.hashCode();

return result;

}

@Override

public boolean equals(final Object obj)

{

if (obj == this)

{

return true;

}

if (obj == null || obj.getClass() != this.getClass())

{

return false;

}

Order order = (Order) obj;

return this.id.equals(order.id);

}

}

Файл OrderItem.java

package by.bsuir.phoneshop.core.models;

public class OrderItem

{

private Long id;

private Phone phone;

private Order order;

private Long quantity;

public OrderItem(final Long id, final Phone phone, final Order order, final Long quantity)

{

this.id = id;

this.phone = phone;

this.order = order;

this.quantity = quantity;

}

public OrderItem(final CartItem cartItem, final Order order)

{

this.phone = cartItem.getPhone();

this.quantity = cartItem.getQuantity();

this.order = order;

}

public Long getId()

{

return id;

}

public void setId(final Long id)

{

this.id = id;

}

public Phone getPhone()

{

return phone;

}

public void setPhone(final Phone phone)

{

this.phone = phone;

}

public Order getOrder()

{

return order;

}

public void setOrder(final Order order)

{

this.order = order;

}

public Long getQuantity()

{

return quantity;

}

public void setQuantity(final Long quantity)

{

this.quantity = quantity;

}

}

Файл ParamsForSearch.java

package by.bsuir.phoneshop.core.models;

public class ParamsForSearch

{

private String search;

private String sortField;

private String order;

private int offset;

private int limit;

public ParamsForSearch(final String search, final String sortField, final String order, final int offset, final int limit)

{

this.search = search;

this.sortField = sortField;

this.order = order;

this.offset = offset;

this.limit = limit;

}

public String getSearch()

{

return search;

}

public void setSearch(final String search)

{

this.search = search;

}

public String getSortField()

{

return sortField;

}

public void setSortField(final String sortField)

{

this.sortField = sortField;

}

public String getOrder()

{

return order;

}

public void setOrder(final String order)

{

this.order = order;

}

public int getOffset()

{

return offset;

}

public void setOffset(final int offset)

{

this.offset = offset;

}

public int getLimit()

{

return limit;

}

public void setLimit(final int limit)

{

this.limit = limit;

}

}

Файл Phone.java

package by.bsuir.phoneshop.core.models;

import java.math.BigDecimal;

import java.util.Set;

import lombok.AccessLevel;

import lombok.Getter;

import lombok.Setter;

@Getter

@Setter

public class Phone

{

private Long id;

private String brand;

private String model;

private BigDecimal price;

private Integer discountPercent;

private String imageUrl;

private Long quantity;

private String description;

private BigDecimal displaySizeInches;

private String displayResolution;

private Integer pixelDensity;

private String displayTechnology;

private Integer weightGr;

private BigDecimal lengthMm;

private BigDecimal widthMm;

private BigDecimal heightMm;

private String deviceType;

private BigDecimal backCameraMegapixels;

private BigDecimal frontCameraMegapixels;

private BigDecimal internalStorageGb;

private Integer batteryCapacityMah;

private BigDecimal talkTimeHours;

private BigDecimal standByTimeHours;

private String bluetooth;

@Setter(AccessLevel.NONE)

private BigDecimal actualPrice;

private Set<Color> colors;

public void setActualPrice()

{

if (discountPercent != null)

{

final float discountPercent = this.discountPercent;

final BigDecimal discountAmount = (this.price).multiply(BigDecimal.valueOf(discountPercent / 100.0));

this.actualPrice = this.price.subtract(discountAmount);

}

else

{

this.actualPrice = this.price;

}

}

@Override

public int hashCode()

{

final int prime = 31;

int result = 1;

result = prime \* result + id.hashCode() + brand.hashCode() + model.hashCode();

return result;

}

@Override

public boolean equals(final Object obj)

{

if (obj == this)

{

return true;

}

if (obj == null || obj.getClass() != this.getClass())

{

return false;

}

Phone phone = (Phone) obj;

return this.id.equals(phone.id) && this.brand.equals(phone.brand) && this.model.equals(phone.model);

}

}

Файл Stock.java

package by.bsuir.phoneshop.core.models;

public class Stock

{

private Phone phone;

private Integer stock;

private Integer reserved;

public Phone getPhone()

{

return phone;

}

public void setPhone(final Phone phone)

{

this.phone = phone;

}

public Integer getStock()

{

return stock;

}

public void setStock(final Integer stock)

{

this.stock = stock;

}

public Integer getReserved()

{

return reserved;

}

public void setReserved(final Integer reserved)

{

this.reserved = reserved;

}

}

Файл User.java

package by.bsuir.phoneshop.core.models;

public class User

{

private String userName;

private String password;

private int isAccountNonLocked;

private String role;

public String getUserName()

{

return userName;

}

public void setUserName(final String userName)

{

this.userName = userName;

}

public String getPassword()

{

return password;

}

public void setPassword(final String password)

{

this.password = password;

}

public int getIsAccountNonLocked()

{

return isAccountNonLocked;

}

public void setIsAccountNonLocked(int isAccountNonLocked)

{

this.isAccountNonLocked = isAccountNonLocked;

}

public String getRole()

{

return role;

}

public void setRole(final String role)

{

this.role = role;

}

}

Файл AuthProviver.java

package by.bsuir.phoneshop.core.service.impl;

import java.util.Arrays;

import javax.annotation.Resource;

import org.springframework.beans.factory.annotation.Value;

import org.springframework.security.authentication.AuthenticationProvider;

import org.springframework.security.authentication.BadCredentialsException;

import org.springframework.security.authentication.LockedException;

import org.springframework.security.authentication.UsernamePasswordAuthenticationToken;

import org.springframework.security.core.Authentication;

import org.springframework.security.core.AuthenticationException;

import org.springframework.security.core.authority.SimpleGrantedAuthority;

import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

import org.springframework.stereotype.Component;

import by.bsuir.phoneshop.core.models.User;

import by.bsuir.phoneshop.core.service.UserService;

@Component

public class AuthProvider implements AuthenticationProvider

{

@Resource

private UserService userDetailsService;

@Value("${user.password.encryption.strength}")

private int ENCRYPTION\_STRENGTH;

@Override

public Authentication authenticate(final Authentication authentication) throws AuthenticationException

{

final BCryptPasswordEncoder passwordEncoder = new BCryptPasswordEncoder(ENCRYPTION\_STRENGTH);

final String username = authentication.getName();

User user;

try

{

user = userDetailsService.loadUserByUsername(username);

}

catch (final Exception e)

{

throw new BadCredentialsException("Username or password is invalid");

}

if (user.getIsAccountNonLocked() == 0)

{

throw new LockedException("You are blocked!");

}

else

{

if (passwordEncoder.matches(authentication.getCredentials().toString(), user.getPassword()))

{

return new UsernamePasswordAuthenticationToken(authentication.getName(), authentication.getCredentials()

.toString(), Arrays.asList(new SimpleGrantedAuthority(user.getRole())));

}

else throw new BadCredentialsException("Username or password is invalid");

}

}

@Override

public boolean supports(final Class<?> aClass)

{

return aClass.equals(UsernamePasswordAuthenticationToken.class);

}

}

Файл HttpSessionCartService.java

package by.bsuir.phoneshop.core.service.impl;

import java.math.BigDecimal;

import java.util.Map;

import java.util.Optional;

import javax.annotation.Resource;

import javax.servlet.http.HttpSession;

import org.springframework.dao.DataAccessException;

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

import by.bsuir.phoneshop.core.dao.PhoneDao;

import by.bsuir.phoneshop.core.dao.StockDao;

import by.bsuir.phoneshop.core.models.Cart;

import by.bsuir.phoneshop.core.models.CartItem;

import by.bsuir.phoneshop.core.models.Phone;

import by.bsuir.phoneshop.core.models.Stock;

import by.bsuir.phoneshop.core.service.CartService;

@Service

public class HttpSessionCartService implements CartService

{

public static final String CART\_SESSION\_ATTR = "cart";

@Resource

private PhoneDao jdbcPhoneDao;

@Resource

private StockDao jdbcStockDao;

@Override

@Transactional(readOnly = true)

public synchronized Cart getCart(final HttpSession httpSession)

{

Cart cart = (Cart) httpSession.getAttribute(CART\_SESSION\_ATTR);

if (cart == null)

{

cart = new Cart();

httpSession.setAttribute(CART\_SESSION\_ATTR, cart);

}

return cart;

}

@Override

@Transactional(rollbackFor = DataAccessException.class)

public synchronized void addPhone(final Long phoneId, final Long quantity, final Cart cart) throws RuntimeException

{

final Optional<Phone> optionalPhone = jdbcPhoneDao.get(phoneId);

final Optional<Stock> optionalStock = jdbcStockDao.get(phoneId);

if (optionalPhone.isPresent() && optionalStock.isPresent())

{

final Stock stock = optionalStock.get();

final Phone phone = optionalPhone.get();

if (stock.getStock() - stock.getReserved() >= quantity)

{

addToCart(quantity, stock, phone, cart);

}

else

{

throw new RuntimeException();

}

}

else

{

throw new IllegalArgumentException("No stock or phone in data base");

}

}

private void addToCart(final Long quantity, final Stock stock, final Phone phone, Cart cart)

{

jdbcStockDao.update(phone.getId(), stock.getStock().longValue(), stock.getReserved() + quantity, false);

final Optional<CartItem> cartItem = findCartItem(phone.getId(), cart);

if (cartItem.isPresent())

{

final CartItem existingCartItem = cartItem.get();

existingCartItem.setQuantity(existingCartItem.getQuantity() + quantity);

}

else

{

final BigDecimal price = phone.getActualPrice().multiply(BigDecimal.valueOf(quantity));

cart.getCartItems().add(new CartItem(phone, quantity, price));

}

calculateCart(cart);

}

@Override

@Transactional(rollbackFor = DataAccessException.class)

public synchronized void update(final Map<Long, Long> items, final Cart cart)

{

items.keySet().stream()

.map(phoneId -> findCartItem(phoneId, cart))

.filter(Optional::isPresent)

.map(Optional::get)

.forEach(cartItem ->

{

final Long phoneId = cartItem.getPhone().getId();

final Long quantity = items.get(cartItem.getPhone().getId());

final Long quantityDifference = quantity - cartItem.getQuantity();

if (checkQuantity(phoneId, quantityDifference))

{

cartItem.setQuantity(cartItem.getQuantity() + quantityDifference);

}

});

calculateCart(cart);

}

private boolean checkQuantity(final Long phoneId, final Long quantityDifference)

{

final Optional<Stock> optionalStock = jdbcStockDao.get(phoneId);

if (optionalStock.isPresent())

{

final Stock stock = optionalStock.get();

if (stock.getStock() - stock.getReserved() >= quantityDifference)

{

jdbcStockDao.update(phoneId, stock.getStock().longValue(),

stock.getReserved() + quantityDifference, false);

return true;

}

}

return false;

}

@Override

@Transactional(rollbackFor = DataAccessException.class)

public synchronized void remove(final Long phoneId, final Cart cart)

{

final Optional<CartItem> optionalCartItem = findCartItem(phoneId, cart);

if (optionalCartItem.isPresent())

{

cart.getCartItems().remove(optionalCartItem.get());

}

else

{

throw new IllegalArgumentException(phoneId.toString());

}

final Optional<Stock> optionalStock = jdbcStockDao.get(phoneId);

if (optionalStock.isPresent() && optionalCartItem.isPresent())

{

final Stock stock = optionalStock.get();

final CartItem cartItem = optionalCartItem.get();

jdbcStockDao.update(phoneId, stock.getStock() + cartItem.getQuantity(),

stock.getReserved() - cartItem.getQuantity(), false);

}

calculateCart(cart);

}

@Override

public void deleteCart(final HttpSession httpSession)

{

httpSession.removeAttribute("cart");

}

private Optional<CartItem> findCartItem(final Long phoneId, final Cart cart)

{

return cart.getCartItems().stream()

.filter(cartItem -> cartItem.getPhone().getId().equals(phoneId))

.findFirst();

}

private void calculateCart(final Cart cart)

{

final Long totalQuantity = cart.getCartItems().stream()

.mapToLong(CartItem::getQuantity)

.sum();

cart.setTotalQuantity(totalQuantity);

final BigDecimal totalCost = cart.getCartItems().stream()

.map(cartItem -> cartItem.getPhone().getActualPrice().multiply(BigDecimal.valueOf(cartItem.getQuantity())))

.reduce(BigDecimal.ZERO, BigDecimal::add);

cart.setTotalCost(totalCost);

}

}

Файл OrderServiceImpl.java

package by.bsuir.phoneshop.core.service.impl;

import java.math.BigDecimal;

import java.util.List;

import java.util.Optional;

import java.util.UUID;

import java.util.concurrent.atomic.AtomicBoolean;

import javax.annotation.Resource;

import org.springframework.dao.DataAccessException;

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

import by.bsuir.phoneshop.core.models.Cart;

import by.bsuir.phoneshop.core.models.Order;

import by.bsuir.phoneshop.core.dao.OrderDao;

import by.bsuir.phoneshop.core.dto.OrderDataDto;

import by.bsuir.phoneshop.core.models.OrderItem;

import by.bsuir.phoneshop.core.models.enums.OrderStatus;

import by.bsuir.phoneshop.core.models.Stock;

import by.bsuir.phoneshop.core.dao.StockDao;

import by.bsuir.phoneshop.core.service.OrderService;

@Service

public class OrderServiceImpl implements OrderService

{

@Resource

private OrderDao jdbcOrderDao;

@Resource

private StockDao jdbcStockDao;

@Override

@Transactional(rollbackFor = DataAccessException.class)

public Long placeOrder(final Cart cart, OrderDataDto orderDataDto, final Long deliveryPrice) throws RuntimeException

{

final AtomicBoolean isOutOfStock = new AtomicBoolean(false);

final Order order = createOrderFromCart(cart, orderDataDto, deliveryPrice);

order.getOrderItems().forEach(orderItem ->

{

final Stock stock = jdbcStockDao.get(orderItem.getPhone().getId()).orElse(null);

if (stock != null && stock.getStock() - orderItem.getQuantity() > 0)

{

jdbcStockDao.update(orderItem.getPhone().getId(), stock.getStock() - orderItem.getQuantity(), stock.getReserved() - orderItem.getQuantity(), false);

}

else

{

isOutOfStock.set(true);

}

});

if (isOutOfStock.get())

{

throw new RuntimeException();

}

order.setId(UUID.randomUUID().getMostSignificantBits() & Long.MAX\_VALUE);

return jdbcOrderDao.save(order);

}

@Override

@Transactional(readOnly = true)

public List<Order> getOrders(final int limit,final int offset)

{

return jdbcOrderDao.getOrders(limit, offset);

}

@Override

public void updateStatus(final OrderStatus orderStatus, final Long orderId)

{

jdbcOrderDao.updateStatus(orderStatus, orderId);

}

@Override

public Optional<Order> getOrderById(final Long id)

{

return jdbcOrderDao.get(id);

}

private Order createOrderFromCart(final Cart cart, final OrderDataDto orderDataDto, final Long deliveryPrice)

{

final Order order = new Order();

order.setSubtotal(cart.getTotalCost());

order.setDeliveryPrice(BigDecimal.valueOf(deliveryPrice));

order.setTotalPrice(order.getSubtotal().add(order.getDeliveryPrice()));

order.setStatus(OrderStatus.NEW);

order.setFirstName(orderDataDto.getFirstName());

order.setLastName(orderDataDto.getLastName());

order.setContactPhoneNo(orderDataDto.getPhone());

order.setDeliveryAddress(orderDataDto.getAddress());

order.setAdditionalInfo(orderDataDto.getAdditionalInfo());

cart.getCartItems().forEach(cartItem ->

{

final OrderItem orderItem = new OrderItem(cartItem, order);

order.getOrderItems().add(orderItem);

});

return order;

}

}

Файл PhoneServiceImpl.java

package by.bsuir.phoneshop.core.service.impl;

import java.util.ArrayList;

import java.util.List;

import java.util.Optional;

import javax.annotation.Resource;

import org.springframework.jdbc.core.JdbcTemplate;

import org.springframework.stereotype.Component;

import by.bsuir.phoneshop.core.models.ParamsForSearch;

import by.bsuir.phoneshop.core.models.Phone;

import by.bsuir.phoneshop.core.dao.PhoneDao;

import by.bsuir.phoneshop.core.dao.StockDao;

import by.bsuir.phoneshop.core.dto.PhoneAddDto;

import by.bsuir.phoneshop.core.service.PhoneService;

@Component

public class PhoneServiceImpl implements PhoneService

{

@Resource

private PhoneDao jdbcPhoneDao;

@Resource

private StockDao jdbcStockDao;

@Resource

private JdbcTemplate jdbcTemplate;

@Override

public Optional<Phone> getPhoneByKey(final Long key)

{

return jdbcPhoneDao.get(key);

}

@Override

public Optional<Phone> getPhoneByModel(final String model)

{

return jdbcPhoneDao.get(model);

}

@Override

public void savePhone(final Phone phone, final Long quantity, final String[] colors)

{

jdbcPhoneDao.save(phone);

final Phone savedPhone = getPhoneByModel(phone.getModel()).get();

jdbcStockDao.update(savedPhone.getId(), quantity, 0L, true);

for (final String color : colors)

{

jdbcTemplate.update("insert into phone2color(phoneId,colorId) values (?,?)", savedPhone.getId(), color);

}

}

@Override

public List<Phone> getPhones(final int offset, final int limit)

{

return jdbcPhoneDao.findAll(offset, limit);

}

@Override

public List<Phone> getPhones(final ParamsForSearch paramsForSearch)

{

return jdbcPhoneDao.findAll(paramsForSearch);

}

@Override

public Long countPhoneQuantity(final ParamsForSearch paramsForSearch)

{

return jdbcPhoneDao.count(paramsForSearch);

}

@Override

public List<Phone> getHotPricedPhones(final int amount)

{

final List<Phone> phones = jdbcPhoneDao.findMaxDiscountPercentPhones();

final List<Phone> limitedPhoneList = new ArrayList<>();

for (int i = 0; i < phones.size(); i++)

{

if (i < amount)

{

limitedPhoneList.add(phones.get(i));

}

else

{

break;

}

}

return limitedPhoneList;

}

@Override

public void deletePhone(final Long id)

{

jdbcPhoneDao.deletePhone(id);

}

@Override

public Phone populateFields(final PhoneAddDto phoneAddDto)

{

final Phone phone = new Phone();

phone.setBrand(phoneAddDto.getBrand());

phone.setModel(phoneAddDto.getModel());

phone.setPrice(phoneAddDto.getPrice());

if (phoneAddDto.getDiscountPercent() != null)

{

phone.setDiscountPercent(phoneAddDto.getDiscountPercent());

}

else

{

phone.setDiscountPercent(0);

}

phone.setImageUrl(phoneAddDto.getImageUrl());

phone.setDescription(phoneAddDto.getDescription());

phone.setDisplaySizeInches(phoneAddDto.getDisplaySizeInches());

phone.setDisplayResolution(phoneAddDto.getDisplayResolution());

phone.setPixelDensity(phoneAddDto.getPixelDensity());

phone.setDisplayTechnology(phoneAddDto.getDisplayTechnology());

phone.setWeightGr(phoneAddDto.getWeightGr());

phone.setLengthMm(phoneAddDto.getLengthMm());

phone.setWidthMm(phoneAddDto.getWidthMm());

phone.setHeightMm(phoneAddDto.getHeightMm());

phone.setDeviceType(phoneAddDto.getDeviceType());

phone.setBackCameraMegapixels(phoneAddDto.getBackCameraMegapixels());

phone.setFrontCameraMegapixels(phoneAddDto.getFrontCameraMegapixels());

phone.setInternalStorageGb(phoneAddDto.getInternalStorageGb());

phone.setBatteryCapacityMah(phoneAddDto.getBatteryCapacityMah());

phone.setTalkTimeHours(phoneAddDto.getTalkTimeHours());

phone.setStandByTimeHours(phoneAddDto.getStandByTimeHours());

phone.setBluetooth(phoneAddDto.getBluetooth());

return phone;

}

@Override

public void updateDevice(final Phone phone, final Long quantity, final String[] colors, final Long id)

{

phone.setId(id);

jdbcPhoneDao.updateDevice(phone);

final Phone savedPhone = getPhoneByModel(phone.getModel()).get();

jdbcStockDao.update(savedPhone.getId(), quantity, 0L, false);

for (final String color : colors)

{

jdbcTemplate.update("delete from phone2color where phoneId=?", savedPhone.getId());

jdbcTemplate.update("insert into phone2color(phoneId,colorId) values (?,?)", savedPhone.getId(), color);

}

}

}

Файл UserServiceImpl.java

package by.bsuir.phoneshop.core.service.impl;

import java.util.List;

import javax.annotation.Resource;

import org.springframework.beans.factory.annotation.Value;

import org.springframework.security.core.userdetails.UsernameNotFoundException;

import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

import org.springframework.stereotype.Service;

import by.bsuir.phoneshop.core.models.Employee;

import by.bsuir.phoneshop.core.models.User;

import by.bsuir.phoneshop.core.dao.UserDao;

import by.bsuir.phoneshop.core.service.UserService;

@Service

public class UserServiceImpl implements UserService

{

@Resource

private UserDao jdbcUserDao;

@Value("${user.password.encryption.strength}")

private int ENCRYPTION\_STRENGTH;

public User loadUserByUsername(final String username)

throws UsernameNotFoundException

{

return jdbcUserDao.getUserByUsername(username)

.orElseThrow(() -> new UsernameNotFoundException("User not present"));

}

@Override

public List<User> getUsersExceptAdmin(final int limit, final int offset)

{

return jdbcUserDao.getUsersExceptAdmin(limit, offset);

}

@Override

public void updateUserStatus(final int status, final String userName)

{

jdbcUserDao.updateUserStatus(status, userName);

}

@Override

public void registerUser(final String username, final String password)

{

final BCryptPasswordEncoder bCryptPasswordEncoder = new BCryptPasswordEncoder(ENCRYPTION\_STRENGTH);

final User user = new User();

user.setUserName(username);

user.setPassword(bCryptPasswordEncoder.encode(password));

user.setIsAccountNonLocked(1);

user.setRole("ROLE\_CUSTOMER");

jdbcUserDao.addNewUserToDb(user);

}

@Override

public List<Employee> getEmployees()

{

return jdbcUserDao.getMainEmployees();

}

}

Файл CartService.java

package by.bsuir.phoneshop.core.service;

import java.util.Map;

import javax.servlet.http.HttpSession;

import by.bsuir.phoneshop.core.models.Cart;

public interface CartService

{

Cart getCart(final HttpSession httpSession);

void addPhone(final Long phoneId, final Long quantity, final Cart cart) throws RuntimeException;

void update(final Map<Long, Long> items, final Cart cart);

void remove(final Long phoneId, final Cart cart);

void deleteCart(final HttpSession httpSession);

}

Файл OrderService.java

package by.bsuir.phoneshop.core.service;

import java.util.List;

import java.util.Optional;

import by.bsuir.phoneshop.core.models.Cart;

import by.bsuir.phoneshop.core.models.Order;

import by.bsuir.phoneshop.core.dto.OrderDataDto;

import by.bsuir.phoneshop.core.models.enums.OrderStatus;

public interface OrderService

{

Long placeOrder(final Cart cart, final OrderDataDto orderDataDto, final Long deliveryPrice) throws RuntimeException;

List<Order> getOrders(final int limit, final int offset);

void updateStatus(final OrderStatus orderStatus, final Long orderId);

Optional<Order> getOrderById(final Long id);

}

Файл PhoneService.java

package by.bsuir.phoneshop.core.service;

import java.util.List;

import java.util.Optional;

import by.bsuir.phoneshop.core.models.ParamsForSearch;

import by.bsuir.phoneshop.core.models.Phone;

import by.bsuir.phoneshop.core.dto.PhoneAddDto;

public interface PhoneService

{

Optional<Phone> getPhoneByKey(final Long key);

Optional<Phone> getPhoneByModel(final String model);

void savePhone(final Phone phone, Long stock, final String[] colors);

List<Phone> getPhones(final int offset, final int limit);

List<Phone> getPhones(final ParamsForSearch paramsForSearch);

Long countPhoneQuantity(final ParamsForSearch paramsForSearch);

List<Phone> getHotPricedPhones(final int amount);

void deletePhone(final Long id);

Phone populateFields(final PhoneAddDto phoneAddDto);

void updateDevice(final Phone phone,final Long stock, final String[] colors, final Long id);

}

Файл UserService.java

package by.bsuir.phoneshop.core.service;

import java.util.List;

import by.bsuir.phoneshop.core.models.Employee;

import by.bsuir.phoneshop.core.models.User;

public interface UserService

{

User loadUserByUsername(final String username);

List<User> getUsersExceptAdmin(int limit, int offset);

void updateUserStatus(final int status, final String userName);

void registerUser(final String username, final String password);

List<Employee> getEmployees();

}

Файл OrderDataDtoValidator.java

package by.bsuir.phoneshop.core.validator;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

import org.springframework.stereotype.Service;

import org.springframework.validation.Errors;

import org.springframework.validation.ValidationUtils;

import org.springframework.validation.Validator;

import by.bsuir.phoneshop.core.dto.OrderDataDto;

@Service

public class OrderDataDtoValidator implements Validator

{

private static final String REGEX\_PHONE = "^\\\\+\\\\d{12}$";

@Override

public boolean supports(final Class<?> aClass)

{

return OrderDataDto.class.equals(aClass);

}

@Override

public void validate(final Object o, final Errors errors)

{

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "firstName", "Empty first name");

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "lastName", "Empty last name");

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "address", "Empty address");

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "phone", "Empty phone field");

if (!errors.hasErrors())

{

OrderDataDto orderDataDto = (OrderDataDto) o;

if (validateWithRegex(orderDataDto.getPhone()))

{

errors.rejectValue("phone", "phone");

}

}

}

private boolean validateWithRegex(final String parameter)

{

Pattern pattern = Pattern.compile(OrderDataDtoValidator.REGEX\_PHONE, Pattern.CASE\_INSENSITIVE);

Matcher matcher = pattern.matcher(parameter);

return matcher.find();

}

}

Файл PhoneAddDtoValidator.java

package by.bsuir.phoneshop.core.validator;

import java.math.BigDecimal;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

import org.springframework.stereotype.Service;

import org.springframework.validation.Errors;

import org.springframework.validation.ValidationUtils;

import org.springframework.validation.Validator;

import by.bsuir.phoneshop.core.dto.PhoneAddDto;

@Service

public class PhoneAddDtoValidator implements Validator

{

@Override

public boolean supports(Class<?> aClass)

{

return PhoneAddDto.class.equals(aClass);

}

@Override

public void validate(Object o, Errors errors)

{

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "brand", "page.admin.validation.empty.brand");

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "model", "page.admin.validation.empty.brand");

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "price", "page.admin.validation.empty.price");

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "imageUrl", "page.admin.validation.empty.imageUrl");

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "quantity", "page.admin.validation.empty.quantity");

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "description", "page.admin.validation.empty.description");

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "displaySizeInches", "page.admin.validation.empty.displaySizeInches");

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "displayResolution", "page.admin.validation.empty.displayResolution");

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "pixelDensity", "page.admin.validation.empty.pixelDensity");

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "displayTechnology", "page.admin.validation.empty.displayTechnology");

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "weightGr", "page.admin.validation.empty.weightGr");

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "lengthMm", "page.admin.validation.empty.lengthMm");

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "widthMm", "page.admin.validation.empty.widthMm");

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "colors", "page.admin.validation.empty.colors");

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "deviceType", "page.admin.validation.empty.deviceType");

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "backCameraMegapixels", "page.admin.validation.empty.backCameraMegapixels");

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "frontCameraMegapixels", "page.admin.validation.empty.frontCameraMegapixels");

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "batteryCapacityMah", "page.admin.validation.empty.batteryCapacityMah");

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "talkTimeHours", "page.admin.validation.empty.talkTimeHours");

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "standByTimeHours", "page.admin.validation.empty.standByTimeHours");

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "bluetooth", "page.admin.validation.empty.bluetooth");

if (!errors.hasErrors())

{

final PhoneAddDto phoneAddDto = (PhoneAddDto) o;

if (phoneAddDto.getPrice() == null || phoneAddDto.getPrice().compareTo(BigDecimal.valueOf(0L)) <= 0)

{

errors.rejectValue("price", "page.admin.wrong.price");

}

if (phoneAddDto.getQuantity() == null || phoneAddDto.getQuantity() <= 0)

{

errors.rejectValue("quantity", "page.admin.wrong.quantity");

}

if (phoneAddDto.getDisplaySizeInches() == null || phoneAddDto.getDisplaySizeInches().compareTo(BigDecimal.ZERO) <= 0)

{

errors.rejectValue("displaySizeInches", "page.admin.wrong.displaySizeInches");

}

if (phoneAddDto.getWeightGr() == null || phoneAddDto.getWeightGr() <= 0)

{

errors.rejectValue("weightGr", "page.admin.wrong.weightGr");

}

if (phoneAddDto.getLengthMm() == null || phoneAddDto.getLengthMm().compareTo(BigDecimal.ZERO) <= 0)

{

errors.rejectValue("lengthMm", "page.admin.wrong.lengthMm");

}

if (phoneAddDto.getWidthMm() == null || phoneAddDto.getWidthMm().compareTo(BigDecimal.ZERO) <= 0)

{

errors.rejectValue("widthMm", "page.admin.wrong.widthMm");

}

if (phoneAddDto.getPixelDensity() == null || phoneAddDto.getPixelDensity() <= 0)

{

errors.rejectValue("pixelDensity", "page.admin.wrong.pixelDensity");

}

if (phoneAddDto.getBackCameraMegapixels() == null || phoneAddDto.getBackCameraMegapixels().compareTo(BigDecimal.ZERO) <= 0)

{

errors.rejectValue("backCameraMegapixels", "page.admin.wrong.backCameraMegapixels");

}

if (phoneAddDto.getFrontCameraMegapixels() == null || phoneAddDto.getFrontCameraMegapixels().compareTo(BigDecimal.ZERO) <= 0)

{

errors.rejectValue("frontCameraMegapixels", "page.admin.wrong.frontCameraMegapixels");

}

if (phoneAddDto.getBatteryCapacityMah() == null || phoneAddDto.getBatteryCapacityMah() <= 0)

{

errors.rejectValue("batteryCapacityMah", "page.admin.wrong.batteryCapacityMah");

}

if (phoneAddDto.getTalkTimeHours() == null || phoneAddDto.getTalkTimeHours().compareTo(BigDecimal.ZERO) <= 0)

{

errors.rejectValue("talkTimeHours", "page.admin.wrong.talkTimeHours");

}

if (phoneAddDto.getStandByTimeHours() == null || phoneAddDto.getStandByTimeHours().compareTo(BigDecimal.ZERO) <= 0)

{

errors.rejectValue("standByTimeHours", "page.admin.wrong.standByTimeHours");

}

if (!validateColorWithRegex(phoneAddDto.getColors()))

{

errors.rejectValue("colors", "bad.color");

}

}

}

private boolean validateColorWithRegex(final String parameter)

{

Pattern pattern = Pattern.compile("^\\d[\\d,]+?\\d$", Pattern.CASE\_INSENSITIVE);

Matcher matcher = pattern.matcher(parameter);

return matcher.find();

}

}

Файл PhoneArrayDtoValidator.java

package by.bsuir.phoneshop.core.validator;

import javax.annotation.Resource;

import org.springframework.stereotype.Service;

import org.springframework.validation.Errors;

import org.springframework.validation.Validator;

import by.bsuir.phoneshop.core.dto.PhoneArrayDto;

import by.bsuir.phoneshop.core.models.Stock;

import by.bsuir.phoneshop.core.dao.StockDao;

@Service

public class PhoneArrayDtoValidator implements Validator

{

@Resource

private StockDao jdbcStockDao;

@Override

public boolean supports(final Class<?> aClass)

{

return PhoneArrayDto.class.equals(aClass);

}

@Override

public void validate(final Object o, final Errors errors)

{

PhoneArrayDto dto = (PhoneArrayDto) o;

for (int i = 0; i < dto.getQuantity().length; i++)

{

try

{

final Stock stock = jdbcStockDao.get(Long.parseLong(dto.getPhoneId()[i])).orElse(null);

final long quantity = Long.parseLong(dto.getQuantity()[i]);

if (quantity <= 0 || stock == null || quantity > (stock.getStock() - stock.getReserved()))

{

errors.rejectValue("quantity", dto.getPhoneId()[i]);

}

}

catch (NumberFormatException e)

{

errors.rejectValue("quantity", dto.getPhoneId()[i]);

}

}

}

}

Файл PhoneDtoValidator.java

package by.bsuir.phoneshop.core.validator;

import javax.annotation.Resource;

import org.springframework.stereotype.Service;

import org.springframework.validation.Errors;

import org.springframework.validation.ValidationUtils;

import org.springframework.validation.Validator;

import by.bsuir.phoneshop.core.dto.PhoneDto;

import by.bsuir.phoneshop.core.models.Stock;

import by.bsuir.phoneshop.core.dao.StockDao;

@Service

public class PhoneDtoValidator implements Validator

{

@Resource

private StockDao jdbcStockDao;

@Override

public boolean supports(final Class<?> aClass)

{

return PhoneDto.class.equals(aClass);

}

@Override

public void validate(final Object o, final Errors errors)

{

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "quantity", "Empty quantity");

ValidationUtils.rejectIfEmptyOrWhitespace(errors, "id", "Empty id");

if (!errors.hasErrors())

{

final PhoneDto dto = (PhoneDto) o;

final Stock stock = jdbcStockDao.get(dto.getId()).orElse(null);

if (dto.getQuantity() <= 0)

{

errors.rejectValue("quantity", "Wrong input");

}

if (stock == null || dto.getQuantity() > (stock.getStock() - stock.getReserved()))

{

errors.rejectValue("quantity", "Out of stock!");

}

}

}

}

Файл WebSecurityConfig.java

package by.bsuir.phoneshop.web.controller.config;

import javax.annotation.Resource;

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

import org.springframework.context.annotation.Configuration;

import org.springframework.security.config.annotation.authentication.builders.AuthenticationManagerBuilder;

import org.springframework.security.config.annotation.web.builders.HttpSecurity;

import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;

import org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapter;

import by.bsuir.phoneshop.core.service.impl.AuthProvider;

@Configuration

@EnableWebSecurity

public class WebSecurityConfig extends WebSecurityConfigurerAdapter

{

@Resource

private AuthProvider authProvider;

@Override

protected void configure(final HttpSecurity http) throws Exception

{

http.csrf().disable()

.authorizeRequests()

.antMatchers("/admin/\*\*").hasRole("ADMIN")

.and()

.authorizeRequests()

.antMatchers("/ajax-cart", "/order","/quick-order","/cart").hasRole("CUSTOMER")

.and()

.formLogin()

.loginPage("/login")

.defaultSuccessUrl("/product-list")

.failureForwardUrl("/login")

.permitAll()

.and()

.logout()

.logoutUrl("/logout")

.logoutSuccessUrl("/product-list")

.permitAll()

.and()

.exceptionHandling().accessDeniedPage("/error/access-denied");

}

@Autowired

protected void configure(final AuthenticationManagerBuilder auth) throws Exception

{

auth.authenticationProvider(authProvider);

}

}

Файл PhoneshopPages.java

package by.bsuir.phoneshop.web.controller.constants;

public interface PhoneshopPages

{

String OrdersPage = "orders";

String OrderOverviewPageAdmin = "order-overview-admin";

String AdminMainPage = "admin-main";

String AddDevicePage = "add-device";

String UpdateDeletePage = "update-delete";

String UserManagementPage = "user-management";

String HotPricesPage = "hot-prices";

String LoginPage = "login";

String CartPage = "cart";

String OrderOverviewPage = "order-overview";

String OrderPage = "order";

String ProductDetailsPage = "product-details";

String ProductListPage = "product-list";

String QuickOrderPage = "quick-order";

String ContactsPage = "contacts-page";

String AboutUsPage = "about-us";

String RegisterPage = "registration-page";

}

Файл AddDeviceController.java

package by.bsuir.phoneshop.web.controller.pages.admin.devicemanagement;

import javax.annotation.Resource;

import org.springframework.context.MessageSource;

import org.springframework.context.i18n.LocaleContextHolder;

import org.springframework.http.MediaType;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.validation.BindingResult;

import org.springframework.validation.Validator;

import org.springframework.validation.annotation.Validated;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestMapping;

import by.bsuir.phoneshop.core.dto.PhoneAddDto;

import by.bsuir.phoneshop.core.service.PhoneService;

import by.bsuir.phoneshop.web.controller.constants.PhoneshopPages;

@Controller

@RequestMapping("/admin/device-operations/add")

public class AddDeviceController

{

@Resource

private PhoneService phoneServiceImpl;

@Resource(name = "phoneAddDtoValidator")

private Validator phoneAddDtoValidator;

@Resource

private MessageSource messageSource;

@GetMapping

public String showAddDevicePage(final Model model)

{

model.addAttribute("phoneAddDto", new PhoneAddDto());

return PhoneshopPages.UserPages.AddDevicePage;

}

@PostMapping(consumes = MediaType.APPLICATION\_FORM\_URLENCODED\_VALUE)

public String addDeviceToDataBase(final @Validated @ModelAttribute(name = "phoneAddDto") PhoneAddDto phoneAddDto, final BindingResult bindingResult, final Model model)

{

phoneAddDtoValidator.validate(phoneAddDto, bindingResult);

final String[] stringColors = phoneAddDto.getColors().split(",");

if (bindingResult.hasErrors())

{

model.addAttribute("recentData", phoneAddDto);

model.addAttribute("errors", bindingResult);

model.addAttribute("error", messageSource.getMessage("page.admin.error.while.adding.device.to.db", null, LocaleContextHolder.getLocale()));

return PhoneshopPages.UserPages.AddDevicePage;

}

phoneServiceImpl.savePhone(phoneServiceImpl.populateFields(phoneAddDto), phoneAddDto.getQuantity(), stringColors);

model.addAttribute("success", messageSource.getMessage("page.admin.success.while.adding.device.to.db", null, LocaleContextHolder.getLocale()));

return PhoneshopPages.UserPages.AddDevicePage;

}

}

Файл UpdateDeleteDeviceController.java

package by.bsuir.phoneshop.web.controller.pages.admin.devicemanagement;

import java.util.List;

import javax.annotation.Resource;

import org.springframework.context.MessageSource;

import org.springframework.context.i18n.LocaleContextHolder;

import org.springframework.http.MediaType;

import org.springframework.http.ResponseEntity;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.validation.BindingResult;

import org.springframework.validation.Validator;

import org.springframework.validation.annotation.Validated;

import org.springframework.web.bind.annotation.ExceptionHandler;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import by.bsuir.phoneshop.core.models.Color;

import by.bsuir.phoneshop.core.models.ParamsForSearch;

import by.bsuir.phoneshop.core.models.Phone;

import by.bsuir.phoneshop.core.exception.errors.ResponseErrors;

import by.bsuir.phoneshop.core.dao.StockDao;

import by.bsuir.phoneshop.core.dto.PhoneAddDto;

import by.bsuir.phoneshop.core.service.PhoneService;

import by.bsuir.phoneshop.web.controller.constants.PhoneshopPages;

@Controller

@RequestMapping(value = "/admin/device-operations/update-delete")

public class UpdateDeleteDeviceController

{

@Resource

private PhoneService phoneServiceImpl;

@Resource(name = "phoneAddDtoValidator")

private Validator phoneAddDtoValidator;

@Resource

private StockDao jdbcStockDao;

@Resource

private MessageSource messageSource;

private static final Long QUANTITY\_ON\_PAGE = 10L;

@GetMapping

public String showDeleteUpdatePage(final String search, Long page, final Model model)

{

if (page == null)

{

page = 1L;

}

long offset = (page - 1) \* QUANTITY\_ON\_PAGE;

final ParamsForSearch paramsForSearch = new ParamsForSearch(search, null, null, (int) offset,

QUANTITY\_ON\_PAGE.intValue());

final List<Phone> phoneList = phoneServiceImpl.getPhones(paramsForSearch);

final Long phoneQuantity = phoneServiceImpl.countPhoneQuantity(paramsForSearch);

long numOfPages = phoneQuantity / QUANTITY\_ON\_PAGE;

long lastPage;

if (phoneQuantity % QUANTITY\_ON\_PAGE != 0)

{

lastPage = numOfPages + 2;

}

else

{

lastPage = numOfPages + 1;

}

model.addAttribute("phones", phoneList);

model.addAttribute("pages", lastPage);

model.addAttribute("phoneQuantity", phoneQuantity);

return PhoneshopPages.UserPages.UpdateDeletePage;

}

@PostMapping(value = "/delete", consumes = "application/json")

public ResponseEntity<?> delete(final @RequestBody Long id)

{

try

{

phoneServiceImpl.deletePhone(id);

return ResponseEntity.ok().build();

}

catch (Exception e)

{

final ResponseErrors errors = new ResponseErrors(e.getMessage());

return ResponseEntity.badRequest().body(errors);

}

}

@GetMapping(value = "/update/{id}")

public String showUpdatePage(final Model model, final @PathVariable Long id)

{

final Phone phone = phoneServiceImpl.getPhoneByKey(id).orElse(null);

if (phone == null)

{

throw new RuntimeException();

}

else

{

model.addAttribute("isUpdatePage", true);

model.addAttribute("recentData", phone);

model.addAttribute("id", phone.getId());

StringBuilder colors = new StringBuilder("");

for (final Color color : phone.getColors())

{

colors.append(color.getId()).append(",");

}

colors.replace(colors.length() - 1, colors.length(), "");

model.addAttribute("quantity", jdbcStockDao.get(phone.getId()).get());

model.addAttribute("colors", colors.toString());

}

return PhoneshopPages.UserPages.AddDevicePage;

}

@PostMapping(value = "/update/{id}", consumes = MediaType.APPLICATION\_FORM\_URLENCODED\_VALUE)

public String updateDeviceInDb(final @Validated @ModelAttribute(name = "phoneAddDto") PhoneAddDto phoneAddDto, final BindingResult bindingResult, final Model model, @PathVariable Long id, final RedirectAttributes redirectAttributes)

{

phoneAddDtoValidator.validate(phoneAddDto, bindingResult);

final String[] stringColors = phoneAddDto.getColors().split(",");

if (bindingResult.hasErrors())

{

model.addAttribute("recentData", phoneAddDto);

model.addAttribute("errors", bindingResult);

model.addAttribute("error", messageSource.getMessage("page.admin.error.while.updating.device.to.db", null, LocaleContextHolder.getLocale()));

return PhoneshopPages.UserPages.AddDevicePage;

}

phoneServiceImpl.updateDevice(phoneServiceImpl.populateFields(phoneAddDto), phoneAddDto.getQuantity(), stringColors, id);

redirectAttributes.addFlashAttribute("success", messageSource.getMessage("page.admin.success.while.updating.device.to.db", null, LocaleContextHolder.getLocale()));

return "redirect:/admin/device-operations/update-delete/update/" + id;

}

@ExceptionHandler(Exception.class)

public String handleError()

{

return "redirect:/error?message=" + messageSource.getMessage("page.admin.error.updating", null, LocaleContextHolder.getLocale());

}

}

Файл AminPageController.java

package by.bsuir.phoneshop.web.controller.pages.admin;

import org.springframework.security.core.Authentication;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RequestMapping;

import by.bsuir.phoneshop.web.controller.constants.PhoneshopPages;

@Controller

@RequestMapping(value = "/admin/main")

public class AdminMainPageController

{

@GetMapping

public String showMainAdminPage(final Model model, final Authentication principal)

{

model.addAttribute("username", principal.getName());

return PhoneshopPages.UserPages.AdminMainPage;

}

}

Файл LoginPageController.java

package by.bsuir.phoneshop.web.controller.pages.admin;

import javax.annotation.Resource;

import org.springframework.context.MessageSource;

import org.springframework.context.i18n.LocaleContextHolder;

import org.springframework.security.core.Authentication;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

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 by.bsuir.phoneshop.web.controller.constants.PhoneshopPages;

@Controller

@RequestMapping("/login")

public class LoginPageController

{

@Resource

private MessageSource messageSource;

@GetMapping

public String getLoginPage(final @RequestParam(required = false) String error, final Authentication authentication,

final Model model)

{

if (authentication != null && authentication.isAuthenticated())

{

return "redirect:/orders";

}

if (error != null)

{

model.addAttribute("error", messageSource.getMessage("invalidCredentials", null, LocaleContextHolder.getLocale()));

}

return PhoneshopPages.UserPages.LoginPage;

}

@PostMapping

public String login(final Model model)

{

model.addAttribute("error", messageSource.getMessage("invalidCredentials", null, LocaleContextHolder.getLocale()));

return PhoneshopPages.UserPages.LoginPage;

}

}

Файл OrderOvervievPageControllerAdmin.java

package by.bsuir.phoneshop.web.controller.pages.admin;

import javax.annotation.Resource;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.ExceptionHandler;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.RequestParam;

import by.bsuir.phoneshop.core.models.enums.OrderStatus;

import by.bsuir.phoneshop.core.service.OrderService;

import by.bsuir.phoneshop.web.controller.constants.PhoneshopPages;

@Controller

@RequestMapping(value = "/admin/orders/{id}")

public class OrderOverviewPageControllerAdmin

{

@Resource

private OrderService orderServiceImpl;

@RequestMapping(method = RequestMethod.GET)

public String getOrderOverview(final @PathVariable("id") Long id, final Model model)

{

orderServiceImpl.getOrderById(id).ifPresent(order -> model.addAttribute("order", order));

model.addAttribute("id", id);

return PhoneshopPages.UserPages.OrderOverviewPageAdmin;

}

@RequestMapping(method = RequestMethod.POST)

public String changeOrderStatus(final @PathVariable("id") Long orderId,

final @RequestParam(name = "orderStatus") String orderStatus)

{

OrderStatus status = OrderStatus.REJECTED;

if (OrderStatus.DELIVERED.name().equals(orderStatus))

{

status = OrderStatus.DELIVERED;

}

else if (OrderStatus.IN\_DELIVERY.name().equals(orderStatus))

{

status = OrderStatus.IN\_DELIVERY;

}

else

{

status = OrderStatus.REJECTED;

}

orderServiceImpl.updateStatus(status, orderId);

return "redirect:/admin/orders/" + orderId;

}

@ExceptionHandler(RuntimeException.class)

public String handleOutOfStock()

{

return "redirect:/error?message=" + "no such order";

}

}

Файл PhoneDtoValidator.java

package by.bsuir.phoneshop.web.controller.pages.admin;

import java.util.List;

import javax.annotation.Resource;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import by.bsuir.phoneshop.core.models.Order;

import by.bsuir.phoneshop.core.service.OrderService;

import by.bsuir.phoneshop.web.controller.constants.PhoneshopPages;

@Controller

@RequestMapping(value = "/admin/orders")

public class OrdersPageController

{

@Resource

private OrderService orderServiceImpl;

@RequestMapping(method = RequestMethod.GET)

public String getOrders(final Model model)

{

final List<Order> orders = orderServiceImpl.getOrders(Integer.MAX\_VALUE, 0);

if (orders != null)

{

model.addAttribute(orders);

}

return PhoneshopPages.UserPages.OrdersPage;

}

}

Файл UserManagementController.java

package by.bsuir.phoneshop.web.controller.pages.admin;

import java.util.List;

import javax.annotation.Resource;

import org.springframework.http.ResponseEntity;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.RequestParam;

import by.bsuir.phoneshop.core.models.User;

import by.bsuir.phoneshop.core.exception.errors.ResponseErrors;

import by.bsuir.phoneshop.core.dto.UserDto;

import by.bsuir.phoneshop.core.service.UserService;

import by.bsuir.phoneshop.web.controller.constants.PhoneshopPages;

@Controller

@RequestMapping(value = "/admin/user-management")

public class UserManagementController

{

@Resource

private UserService userServiceImpl;

private static final int LIMIT\_ON\_PAGE = 10;

@GetMapping

public String showUserManagementPage(final Model model, @RequestParam(name = "page", required = false) Integer page)

{

if (page == null)

{

page = 0;

}

int offset = page \* LIMIT\_ON\_PAGE;

final List<User> users = userServiceImpl.getUsersExceptAdmin(LIMIT\_ON\_PAGE, offset);

model.addAttribute("users", users);

return PhoneshopPages.UserPages.UserManagementPage;

}

@RequestMapping(method = RequestMethod.POST, value = "/update-status", consumes = "application/json")

public ResponseEntity<?> updateUserStatus(final @RequestBody UserDto userDto)

{

int statusToUpdate = userDto.getStatus() == 0 ? 1 : 0;

try

{

userServiceImpl.updateUserStatus(statusToUpdate, userDto.getUserName());

return ResponseEntity.ok().build();

}

catch (Exception e)

{

final ResponseErrors errors = new ResponseErrors(e.getMessage());

return ResponseEntity.badRequest().body(errors);

}

}

}

Файл AboutUsPageController.java

package by.bsuir.phoneshop.web.controller.pages;

import javax.annotation.Resource;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RequestMapping;

import by.bsuir.phoneshop.core.service.UserService;

import by.bsuir.phoneshop.web.controller.constants.PhoneshopPages;

@Controller

@RequestMapping(value = "/about-us")

public class AboutUsPageController

{

@Resource

private UserService userService;

@GetMapping

public String showAboutUsPage(final Model model)

{

model.addAttribute("employees", userService.getEmployees());

return PhoneshopPages.UserPages.AboutUsPage;

}

}

Файл CartPageController.java

package by.bsuir.phoneshop.web.controller.pages;

import java.util.HashMap;

import java.util.List;

import java.util.Optional;

import java.util.stream.Collectors;

import javax.annotation.Resource;

import javax.servlet.http.HttpSession;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.validation.BindingResult;

import org.springframework.validation.FieldError;

import org.springframework.validation.annotation.Validated;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.RequestParam;

import by.bsuir.phoneshop.core.models.Cart;

import by.bsuir.phoneshop.core.models.Phone;

import by.bsuir.phoneshop.core.dto.PhoneArrayDto;

import by.bsuir.phoneshop.core.service.CartService;

import by.bsuir.phoneshop.core.service.PhoneService;

import by.bsuir.phoneshop.core.validator.PhoneArrayDtoValidator;

import by.bsuir.phoneshop.web.controller.constants.PhoneshopPages;

@Controller

@RequestMapping(value = "/cart")

public class CartPageController

{

public static final String REDIRECT\_TO\_CART\_PAGE = "redirect:/" + PhoneshopPages.UserPages.CartPage;

@Resource

private CartService cartService;

@Resource

private HttpSession httpSession;

@Resource

private PhoneService phoneServiceImpl;

@Resource

private PhoneArrayDtoValidator phoneArrayDtoValidator;

@RequestMapping(method = RequestMethod.GET)

public String getCart(final @RequestParam(value = "successDelete", required = false) boolean successDelete,

final @RequestParam(value = "successUpdate", required = false) boolean successUpdate,

final @RequestParam(value = "isOutOfStock", required = false) boolean isOutOfStock,

final @RequestParam(value = "error", required = false) boolean error,

final @RequestParam(value = "errorsId", required = false) List<Long> errorsId,

final Model model)

{

final Cart cart = cartService.getCart(httpSession);

model.addAttribute("successDelete", successDelete);

model.addAttribute("successUpdate", successUpdate);

model.addAttribute("isOutOfStock", isOutOfStock);

model.addAttribute("error", error);

model.addAttribute("errorsId", errorsId);

model.addAttribute("cart", cart);

return PhoneshopPages.UserPages.CartPage;

}

@RequestMapping(method = RequestMethod.POST)

public String updateCart(final @Validated @ModelAttribute(name = "phoneArrayDto") PhoneArrayDto phoneArrayDto, final Model model, final BindingResult bindingResult)

{

final Cart cart = cartService.getCart(httpSession);

if (cart.getCartItems().isEmpty())

{

return prepareModelForEmptyCart(cart, model);

}

phoneArrayDtoValidator.validate(phoneArrayDto, bindingResult);

if (bindingResult.hasErrors())

{

return validationFailed(cart, bindingResult, model);

}

final HashMap<Long, Long> idAndQuantity = new HashMap<>();

for (int i = 0; i < phoneArrayDto.getQuantity().length; i++)

{

idAndQuantity.put(Long.parseLong(phoneArrayDto.getPhoneId()[i]),

Long.parseLong(phoneArrayDto.getQuantity()[i]));

}

cartService.update(idAndQuantity, cart);

model.addAttribute("successUpdate", true);

model.addAttribute("cart", cart);

return REDIRECT\_TO\_CART\_PAGE;

}

@RequestMapping(value = "{id}", method = RequestMethod.POST)

public String deleteFromCart(final @PathVariable("id") Long id, final Model model) throws IllegalArgumentException

{

final Optional<Phone> optionalPhone = phoneServiceImpl.getPhoneByKey(id);

final Cart cart = cartService.getCart(httpSession);

if (cart.getCartItems().isEmpty())

{

return prepareModelForEmptyCart(cart, model);

}

if (optionalPhone.isPresent())

{

cartService.remove(id, cart);

model.addAttribute("successDelete", true);

model.addAttribute("cart", cart);

}

else

{

model.addAttribute("error", true);

throw new IllegalArgumentException(String.valueOf(id));

}

return REDIRECT\_TO\_CART\_PAGE;

}

private String prepareModelForEmptyCart(final Cart cart, final Model model)

{

model.addAttribute("error", true);

model.addAttribute("cart", cart);

return REDIRECT\_TO\_CART\_PAGE;

}

private String validationFailed(final Cart cart, final BindingResult bindingResult, final Model model)

{

final List<FieldError> errorList = bindingResult.getFieldErrors("quantity");

final List<Long> errorsId = errorList.stream()

.map(item -> Long.parseLong(item.getCode()))

.collect(Collectors.toList());

model.addAttribute("cart", cart);

model.addAttribute("errorsId", errorsId);

model.addAttribute("error", true);

return REDIRECT\_TO\_CART\_PAGE;

}

}

Файл ContactsPageController.java

package by.bsuir.phoneshop.web.controller.pages;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RequestMapping;

import by.bsuir.phoneshop.web.controller.constants.PhoneshopPages;

@Controller

@RequestMapping("/contacts")

public class ContactsPageController

{

@GetMapping

public String getContactsPage()

{

return PhoneshopPages.UserPages.ContactsPage;

}

}

Файл GlobalController.java

package by.bsuir.phoneshop.web.controller.pages;

import java.nio.file.AccessDeniedException;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.springframework.http.HttpStatus;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.ControllerAdvice;

import org.springframework.web.bind.annotation.ExceptionHandler;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.ResponseStatus;

import org.springframework.web.servlet.NoHandlerFoundException;

@ControllerAdvice

@RequestMapping("/error")

public class GlobalController

{

private static final String ADMIN\_MAIL="mailto:antosa2972@gmail.com";

@ExceptionHandler(NoHandlerFoundException.class)

public String exception404(final HttpServletRequest httpRequest, final HttpServletResponse httpServletResponse, final Model model)

{

model.addAttribute("adminMail", ADMIN\_MAIL);

model.addAttribute("status", "404");

return "error";

}

@ExceptionHandler(AccessDeniedException.class)

@GetMapping("/access-denied")

public String accessDenied(final HttpServletRequest httpRequest, final HttpServletResponse httpServletResponse, final Model model)

{

model.addAttribute("adminMail", ADMIN\_MAIL);

model.addAttribute("status", "403");

return "error";

}

@ResponseStatus(HttpStatus.INTERNAL\_SERVER\_ERROR)

@ExceptionHandler(Exception.class)

public String internalServerError(final HttpServletRequest httpRequest, final HttpServletResponse httpServletResponse, final Model model)

{

model.addAttribute("adminMail", ADMIN\_MAIL);

model.addAttribute("status", "500");

return "error";

}

}

Файл HotPricesController.java

package by.bsuir.phoneshop.web.controller.pages;

import java.util.List;

import javax.annotation.Resource;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.util.CollectionUtils;

import org.springframework.web.bind.annotation.GetMapping;

import by.bsuir.phoneshop.core.models.Phone;

import by.bsuir.phoneshop.core.service.PhoneService;

import by.bsuir.phoneshop.web.controller.constants.PhoneshopPages;

@Controller

public class HotPricesPageController

{

private static final int AMOUNT\_TO\_GET = 10;

@Resource

private PhoneService phoneServiceImpl;

@GetMapping("/hot-prices")

public String getHotPricesPage(final Model model)

{

final List<Phone> phonesWithHotPrices = phoneServiceImpl.getHotPricedPhones(AMOUNT\_TO\_GET);

if (phonesWithHotPrices != null && !CollectionUtils.isEmpty(phonesWithHotPrices))

{

model.addAttribute("phones", phonesWithHotPrices);

}

else

{

model.addAttribute("isEmptyPhoneList", true);

}

return PhoneshopPages.UserPages.HotPricesPage;

}

}

Файл OrderOverviewPageController.java

package by.bsuir.phoneshop.web.controller.pages;

import javax.annotation.Resource;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.ExceptionHandler;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import by.bsuir.phoneshop.core.service.OrderService;

import by.bsuir.phoneshop.web.controller.constants.PhoneshopPages;

@Controller

@RequestMapping(value = "/order-overview/{id}")

public class OrderOverviewPageController

{

@Resource

OrderService orderServiceImpl;

@RequestMapping(method = RequestMethod.GET)

public String getOrderOverview(final @PathVariable("id") Long id, final Model model)

{

orderServiceImpl.getOrderById(id).ifPresent(order -> model.addAttribute("order", order));

model.addAttribute("id", id);

return PhoneshopPages.UserPages.OrderOverviewPage;

}

@ExceptionHandler(RuntimeException.class)

public String handleError()

{

return "redirect:/error?message=" + "no such order";

}

}

Файл OrderPageController.java

package by.bsuir.phoneshop.web.controller.pages;

import java.math.BigDecimal;

import javax.annotation.Resource;

import javax.servlet.http.HttpSession;

import org.springframework.context.MessageSource;

import org.springframework.context.annotation.PropertySource;

import org.springframework.context.i18n.LocaleContextHolder;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.validation.BindingResult;

import org.springframework.validation.annotation.Validated;

import org.springframework.web.bind.annotation.ExceptionHandler;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import by.bsuir.phoneshop.core.models.Cart;

import by.bsuir.phoneshop.core.dto.OrderDataDto;

import by.bsuir.phoneshop.core.service.CartService;

import by.bsuir.phoneshop.core.service.OrderService;

import by.bsuir.phoneshop.core.validator.OrderDataDtoValidator;

import by.bsuir.phoneshop.web.controller.constants.PhoneshopPages;

@Controller

@PropertySource("classpath:/messages\_en.properties")

@RequestMapping(value = "/order")

public class OrderPageController

{

@Resource

private OrderService orderService;

@Resource

private CartService cartService;

@Resource

private HttpSession httpSession;

@Resource

private OrderDataDtoValidator orderDataDtoValidator;

@Resource

private MessageSource messageSource;

@RequestMapping(method = RequestMethod.GET)

public String getOrder(final Model model)

{

model.addAttribute("errors", model.asMap().get("errors"));

model.addAttribute("error", model.asMap().get("error"));

Cart cart = cartService.getCart(httpSession);

showCartAsOrder(cart, model);

return PhoneshopPages.UserPages.OrderPage;

}

@RequestMapping(method = RequestMethod.POST)

public String placeOrder(final @Validated @ModelAttribute(name = "orderDataDto") OrderDataDto orderDataDto,

final BindingResult bindingResult,

final RedirectAttributes redirectAttributes) throws RuntimeException

{

final Cart cart = cartService.getCart(httpSession);

if (cart.getCartItems().isEmpty())

{

return prepareModelForEmptyCart(redirectAttributes);

}

orderDataDtoValidator.validate(orderDataDto, bindingResult);

if (bindingResult.hasErrors())

{

return prepareModelForValidationErrors(bindingResult, redirectAttributes);

}

final Long id = orderService.placeOrder(cart, orderDataDto, Long.parseLong(messageSource.getMessage("delivery.price", null, LocaleContextHolder.getLocale())));

cartService.deleteCart(httpSession);

return "redirect:/" + PhoneshopPages.UserPages.OrderOverviewPage + "/" + id;

}

private String prepareModelForValidationErrors(final BindingResult bindingResult,

final RedirectAttributes redirectAttributes)

{

redirectAttributes.addFlashAttribute("error", true);

redirectAttributes.addFlashAttribute("errors", bindingResult);

return "redirect:/" + PhoneshopPages.UserPages.OrderPage;

}

private String prepareModelForEmptyCart(final RedirectAttributes redirectAttributes)

{

redirectAttributes.addFlashAttribute("error", true);

return "redirect:/" + PhoneshopPages.UserPages.OrderPage;

}

private void showCartAsOrder(final Cart cart, final Model model)

{

model.addAttribute("cart", cart);

final BigDecimal deliveryPrice = BigDecimal.valueOf(Long.parseLong(

messageSource.getMessage("delivery.price", null, LocaleContextHolder.getLocale())));

model.addAttribute("deliveryPrice", deliveryPrice);

model.addAttribute("totalCost", cart.getTotalCost().add(deliveryPrice));

}

@ExceptionHandler(RuntimeException.class)

public String handleError()

{

return "redirect:/error?message=" + messageSource.getMessage("exception.outOfStock", null, LocaleContextHolder.getLocale());

}

}

Файл ProductDetailsPageController.java

package by.bsuir.phoneshop.web.controller.pages;

import javax.annotation.Resource;

import javax.servlet.http.HttpSession;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import by.bsuir.phoneshop.core.models.Phone;

import by.bsuir.phoneshop.core.service.CartService;

import by.bsuir.phoneshop.core.service.PhoneService;

import by.bsuir.phoneshop.web.controller.constants.PhoneshopPages;

@Controller

@RequestMapping(value = "/product-details/{id}")

public class ProductDetailsPageController

{

@Resource

private PhoneService phoneServiceImpl;

@Resource

private CartService httpSessionCartService;

@Resource

private HttpSession httpSession;

@RequestMapping(method = RequestMethod.GET)

public String showProductDetailsInfo(final @PathVariable("id") Long phoneId, Model model)

{

final Phone phone = phoneServiceImpl.getPhoneByKey(phoneId).orElse(null);

model.addAttribute("phone", phone);

model.addAttribute("cart", httpSessionCartService.getCart(httpSession));

return PhoneshopPages.UserPages.ProductDetailsPage;

}

}

Файл ProductListController.java

package by.bsuir.phoneshop.web.controller.pages;

import java.util.List;

import java.util.Locale;

import javax.annotation.Resource;

import javax.servlet.http.HttpSession;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.RequestParam;

import by.bsuir.phoneshop.core.models.ParamsForSearch;

import by.bsuir.phoneshop.core.models.Phone;

import by.bsuir.phoneshop.core.models.enums.SortField;

import by.bsuir.phoneshop.core.models.enums.SortOrder;

import by.bsuir.phoneshop.core.service.CartService;

import by.bsuir.phoneshop.core.service.PhoneService;

import by.bsuir.phoneshop.web.controller.constants.PhoneshopPages;

@Controller

@RequestMapping(value = "/product-list")

public class ProductListPageController

{

public static final Long QUANTITY\_ON\_PAGE = 10L;

@Resource

private PhoneService phoneServiceImpl;

@Resource

private CartService cartService;

@Resource

private HttpSession httpSession;

@RequestMapping(method = RequestMethod.GET)

public String showProductList(final @RequestParam(required = false) String search,

@RequestParam(required = false) String field,

@RequestParam(required = false) String order,

@RequestParam(required = false) Long page, final Model model)

{

if (page == null)

{

page = 1L;

}

if (field != null && order != null)

{

try

{

field = SortField.valueOf(field.toUpperCase(Locale.ROOT)).name();

order = SortOrder.valueOf(order.toUpperCase(Locale.ROOT)).name();

}

catch (IllegalArgumentException e)

{

field = null;

order = null;

}

}

long offset = (page - 1) \* QUANTITY\_ON\_PAGE;

final ParamsForSearch paramsForSearch = new ParamsForSearch(search,

field, order, (int) offset,

QUANTITY\_ON\_PAGE.intValue());

final List<Phone> phoneList = phoneServiceImpl.getPhones(paramsForSearch);

final Long phoneQuantity = phoneServiceImpl.countPhoneQuantity(paramsForSearch);

long numOfPages = phoneQuantity / QUANTITY\_ON\_PAGE;

long lastPage;

if (phoneQuantity % QUANTITY\_ON\_PAGE != 0)

{

lastPage = numOfPages + 2;

}

else

{

lastPage = numOfPages + 1;

}

model.addAttribute("phones", phoneList);

model.addAttribute("cart", cartService.getCart(httpSession));

model.addAttribute("pages", lastPage);

model.addAttribute("phoneQuantity", phoneQuantity);

return PhoneshopPages.UserPages.ProductListPage;

}

}

Файл QuickOrderController.java

package by.bsuir.phoneshop.web.controller.pages;

import java.util.ArrayList;

import java.util.List;

import java.util.Optional;

import javax.annotation.Resource;

import javax.servlet.http.HttpSession;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.validation.BindingResult;

import org.springframework.validation.annotation.Validated;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import by.bsuir.phoneshop.core.models.Cart;

import by.bsuir.phoneshop.core.models.Phone;

import by.bsuir.phoneshop.core.dto.QuickOrderElement;

import by.bsuir.phoneshop.core.dto.QuickOrderElementsDto;

import by.bsuir.phoneshop.core.service.CartService;

import by.bsuir.phoneshop.core.service.PhoneService;

import by.bsuir.phoneshop.web.controller.constants.PhoneshopPages;

@Controller

@RequestMapping(value = "/quick-order")

public class QuickOrderController

{

@Resource

private HttpSession httpSession;

@Resource

private PhoneService phoneServiceImpl;

@Resource

private CartService httpSessionCartService;

private final List<Phone> phoneList;

private QuickOrderController()

{

this.phoneList = new ArrayList<>();

}

@RequestMapping(method = RequestMethod.GET)

public String getQuickOrderPage(final Model model)

{

model.addAttribute("success", model.asMap().get("success"));

model.addAttribute("quickOrderElementsDto", new QuickOrderElementsDto());

return "quick-order";

}

@RequestMapping(method = RequestMethod.POST)

public String makeQuickOrder(final @ModelAttribute(name = "quickOrderElementsDto") @Validated QuickOrderElementsDto quickOrderElementsDto,

final BindingResult bindingResult,

final RedirectAttributes redirectAttributes,

final Model model)

{

Cart cart = httpSessionCartService.getCart(httpSession);

quickOrderElementsDto.getQuickOrderElements().forEach(quickOrderElement ->

{

int indexInList = quickOrderElementsDto.getQuickOrderElements().indexOf(quickOrderElement);

addToCart(quickOrderElementsDto, cart, bindingResult, indexInList);

});

if (bindingResult.hasFieldErrors())

{

model.addAttribute("successfulPhones", phoneList);

model.addAttribute("errors", bindingResult);

return PhoneshopPages.UserPages.QuickOrderPage;

}

redirectAttributes.addFlashAttribute("success", true);

return "redirect:" + PhoneshopPages.UserPages.QuickOrderPage;

}

private void addToCart(final QuickOrderElementsDto quickOrderElementsDto,

final Cart cart, final BindingResult bindingResult, final int indexInList)

{

final QuickOrderElement quickOrderElement = quickOrderElementsDto.getQuickOrderElements().get(indexInList);

final Long quantity = quickOrderElement.getQuantity();

final String modelPhone = quickOrderElement.getModel();

if (quantity == null && modelPhone.isEmpty())

{

return;

}

final Optional<Phone> optionalPhone = phoneServiceImpl.getPhoneByModel(modelPhone);

optionalPhone.ifPresent(phone ->

{

if (quantity != null)

{

try

{

httpSessionCartService.addPhone(phone.getId(), quantity, cart);

quickOrderElementsDto.getQuickOrderElements().set(indexInList, new QuickOrderElement());

}

catch (RuntimeException e)

{

bindingResult.rejectValue("quickOrderElements[" + indexInList + "].quantity",

"errors.quantity.outOfStock", "Out of stock!");

}

phoneList.add(phone);

}

else

{

bindingResult.rejectValue("quickOrderElements[" + indexInList + "].quantity",

"errors.emptyField", "Empty field!");

}

});

if (!optionalPhone.isPresent())

{

bindingResult.rejectValue("quickOrderElements[" + indexInList + "].model",

"errors.product.notFound ", "No such product!");

}

}

}

Файл RegisterPageController.java

package by.bsuir.phoneshop.web.controller.pages;

import javax.annotation.Resource;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

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 by.bsuir.phoneshop.core.service.UserService;

import by.bsuir.phoneshop.web.controller.constants.PhoneshopPages;

@Controller

@RequestMapping(value = "/register")

public class RegisterPageController

{

@Resource

private UserService userServiceImpl;

@GetMapping

public String showRegisterPage()

{

return PhoneshopPages.UserPages.RegisterPage;

}

@PostMapping

public String registerUser(final @RequestParam String username, @RequestParam final String password1, final Model model)

{

try

{

if (username == null || username.isEmpty() || password1 == null || password1.isEmpty() || username.length() < 5 || password1.length() < 5)

{

model.addAttribute("errorEmpty", true);

return PhoneshopPages.UserPages.RegisterPage;

}

userServiceImpl.registerUser(username, password1);

}

catch (final Exception e)

{

model.addAttribute("error", true);

return PhoneshopPages.UserPages.RegisterPage;

}

model.addAttribute("isRegistered", true);

return PhoneshopPages.UserPages.LoginPage;

}

}

Файл AjaxCartController.java

package by.bsuir.phoneshop.web.controller;

import static org.springframework.http.ResponseEntity.badRequest;

import javax.annotation.Resource;

import javax.servlet.http.HttpSession;

import org.springframework.http.ResponseEntity;

import org.springframework.stereotype.Controller;

import org.springframework.validation.BindingResult;

import org.springframework.validation.Validator;

import org.springframework.validation.annotation.Validated;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import by.bsuir.phoneshop.core.models.Cart;

import by.bsuir.phoneshop.core.exception.errors.ResponseErrors;

import by.bsuir.phoneshop.core.exception.errors.ValidationErrors;

import by.bsuir.phoneshop.core.dto.PhoneDto;

import by.bsuir.phoneshop.core.service.CartService;

@Controller

@RequestMapping(value = "/ajax-cart")

public class AjaxCartController

{

@Resource(name = "phoneDtoValidator")

private Validator quantityValidator;

@Resource

private CartService cartService;

@Resource

private HttpSession httpSession;

@RequestMapping(method = RequestMethod.POST, consumes = "application/json")

public ResponseEntity<?> addPhone(final @Validated @RequestBody PhoneDto phoneDto, final BindingResult bindingResult)

{

quantityValidator.validate(phoneDto, bindingResult);

if (bindingResult.hasErrors())

{

ValidationErrors validationErrors = new ValidationErrors(bindingResult.getAllErrors());

return badRequest().body(validationErrors.getErrors().get(0).getCode());

}

try

{

final Cart currentCart = cartService.getCart(httpSession);

cartService.addPhone(phoneDto.getId(), phoneDto.getQuantity(), currentCart);

return ResponseEntity.ok(cartService.getCart(httpSession));

}

catch (final RuntimeException e)

{

final ResponseErrors errors = new ResponseErrors(e.getMessage());

return ResponseEntity.badRequest().body(errors);

}

catch (final Exception e)

{

return ResponseEntity.status(500).build();

}

}

}

Файл about-us.jsp

<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>  
<%@ taglib prefix="tags" tagdir="/WEB-INF/tags" %>  
<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>  
<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form" %>  
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn" %>  
<%@ taglib prefix="sprng" uri="http://java.sun.com/jsp/jstl/fmt" %>  
<tags:master pageTitle="Cart">  
 <div class="bg-light">  
 <div class="container py-5">  
 <div class="row h-100 align-items-center py-5">  
 <div class="col-lg-6">  
 <h1 class="display-4"><spring:message code="page.about.us.page.name"/></h1>  
 <p class="lead text-muted mb-0"><spring:message code="page.about.us.first.text"/></p>  
 </div>  
 <div class="col-lg-6 d-none d-lg-block"><img src="https://bootstrapious.com/i/snippets/sn-about/illus.png"  
 alt="" class="img-fluid"></div>  
 </div>  
 </div>  
 </div>  
  
 <div class="bg-white py-5">  
 <div class="container py-5">  
 <div class="row align-items-center mb-5">  
 <div class="col-lg-6 order-2 order-lg-1"><i class="fa fa-bar-chart fa-2x mb-3 text-primary"></i>  
 <h2 class="font-weight-light"><spring:message code="page.about.us.discounts.info.name"/></h2>  
 <p class="font-italic text-muted mb-4"><spring:message code="page.about.us.discounts.info"/></p><a  
 href="${pageContext.servletContext.contextPath}/hot-prices"  
 class="btn btn-light px-5 rounded-pill shadow-sm">  
 <spring:message code="page.hot.prices"/></a>  
 </div>  
 <div class="col-lg-5 px-5 mx-auto order-1 order-lg-2"><img  
 src="https://bootstrapious.com/i/snippets/sn-about/img-1.jpg" alt="" class="img-fluid mb-4 mb-lg-0">  
 </div>  
 </div>  
 <div class="row align-items-center">  
 <div class="col-lg-5 px-5 mx-auto"><img src="https://bootstrapious.com/i/snippets/sn-about/img-2.jpg" alt=""  
 class="img-fluid mb-4 mb-lg-0"></div>  
 <div class="col-lg-6"><i class="fa fa-leaf fa-2x mb-3 text-primary"></i>  
 <h2 class="font-weight-light"><spring:message code="page.about.us.for.business.info.name"/></h2>  
 <p class="font-italic text-muted mb-4"><spring:message code="page.about.us.for.business.info"/></p>  
 <a  
 href="${pageContext.servletContext.contextPath}/contacts"  
 class="btn btn-light px-5 rounded-pill shadow-sm">  
 <spring:message code="page.about.us.for.business.info.for.suppliers"/> </a>  
 <a  
 href="${pageContext.servletContext.contextPath}/quick-order"  
 class="btn btn-light px-5 rounded-pill shadow-sm">  
 <spring:message code="page.about.us.for.business.info.for.for.buyers"/></a>  
 </div>  
 </div>  
 </div>  
 </div>  
  
 <div class="bg-light py-5">  
 <div class="container py-5">  
 <div class="row mb-4">  
 <div class="col-lg-5">  
 <h2 class="display-4 font-weight-light"><spring:message  
 code="page.about.us.out.founders.and.employees"/></h2>  
 <p class="font-italic text-muted"><spring:message  
 code="page.about.us.out.founders.and.employees.description"/></p>  
 </div>  
 </div>  
 <div class="row text-center">  
 <c:forEach var="employee" items="${employees}">  
 <c:choose>  
 <c:when test="${employee.employeeType eq 'FEMALE'}">  
 <div class="col-xl-3 col-sm-6 mb-5">  
 <div class="bg-white rounded shadow-sm py-5 px-4"><img  
 src="https://bootstrapious.com/i/snippets/sn-about/avatar-4.png" alt="" width="100"  
 class="img-fluid rounded-circle mb-3 img-thumbnail shadow-sm">  
 <h5 class="mb-0">${employee.firstName} ${employee.lastName}</h5><span  
 class="small text-uppercase text-muted">${employee.position}</span>  
 <ul class="social mb-0 list-inline mt-3">  
 <li class="list-inline-item"><a href="#" class="social-link"><i  
 class="fa fa-facebook-f"></i></a>  
 </li>  
 <li class="list-inline-item"><a href="#" class="social-link"><i class="fa fa-twitter"></i></a>  
 </li>  
 <li class="list-inline-item"><a href="#" class="social-link"><i  
 class="fa fa-instagram"></i></a>  
 </li>  
 </ul>  
 </div>  
 </div>  
 </c:when>  
 <c:when test="${employee.employeeType eq 'MALE'}">  
 <div class="col-xl-3 col-sm-6 mb-5">  
 <div class="bg-white rounded shadow-sm py-5 px-4"><img  
 src="https://bootstrapious.com/i/snippets/sn-about/avatar-3.png" alt="" width="100"  
 class="img-fluid rounded-circle mb-3 img-thumbnail shadow-sm">  
 <h5 class="mb-0">${employee.firstName} ${employee.lastName}</h5><span  
 class="small text-uppercase text-muted">${employee.position}</span>  
 <ul class="social mb-0 list-inline mt-3">  
 <li class="list-inline-item"><a href="#" class="social-link"><i  
 class="fa fa-facebook-f"></i></a>  
 </li>  
 <li class="list-inline-item"><a href="#" class="social-link"><i class="fa fa-twitter"></i></a>  
 </li>  
 <li class="list-inline-item"><a href="#" class="social-link"><i  
 class="fa fa-instagram"></i></a>  
 </li>  
 </ul>  
 </div>  
 </div>  
 </c:when>  
 <c:when test="${employee.employeeType eq 'GEEK\_MALE'}">  
 <div class="col-xl-3 col-sm-6 mb-5">  
 <div class="bg-white rounded shadow-sm py-5 px-4"><img  
 src="https://bootstrapious.com/i/snippets/sn-about/avatar-2.png" alt="" width="100"  
 class="img-fluid rounded-circle mb-3 img-thumbnail shadow-sm">  
 <h5 class="mb-0">${employee.firstName} ${employee.lastName}</h5><span  
 class="small text-uppercase text-muted">${employee.position}</span>  
 <ul class="social mb-0 list-inline mt-3">  
 <li class="list-inline-item"><a href="#" class="social-link"><i  
 class="fa fa-facebook-f"></i></a>  
 </li>  
 <li class="list-inline-item"><a href="#" class="social-link"><i class="fa fa-twitter"></i></a>  
 </li>  
 <li class="list-inline-item"><a href="#" class="social-link"><i  
 class="fa fa-instagram"></i></a>  
 </li>  
 </ul>  
 </div>  
 </div>  
 </c:when>  
 </c:choose>  
 </c:forEach>  
 </div>  
 </div>  
 </div>  
 </div>  
</tags:master>

Файл add-device.jsp

<%@ page contentType="text/html;charset=UTF-8" %>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>  
<%@ taglib prefix="tags" tagdir="/WEB-INF/tags" %>  
<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>  
<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form" %>  
<tags:masterLogged pageTitle="Add device">  
 <head>  
 <script src="https://code.jquery.com/jquery-1.8.3.js"></script>  
 <title><spring:message code="page.admin.add.device"/></title>  
 </head>  
 <body>  
 <c:choose>  
 <c:when test="${not empty isUpdatePage and isUpdatePage eq true}">  
 <h3>  
<spring:message code="page.admin.update.device"/>  
 </h3>  
 </c:when>  
 <c:otherwise>  
 <h3>  
 <spring:message code="page.admin.add.device"/>  
 </h3>  
 </c:otherwise>  
 </c:choose>  
 <c:if test="${not empty error}">  
 <h4>  
 <font style="color: red"><c:out value="${error}"/></font>  
 </h4>  
 </c:if>  
 <c:if test="${not empty success}">  
 <h4>  
 <font style="color:blue"><c:out value="${success}"/></font>  
 </h4>  
 </c:if>  
 <c:choose>  
 <c:when test="${empty isUpdatePage or isUpdatePage eq false}">  
 <form:form name="add-device-form"  
 action="${pageContext.request.contextPath}/admin/device-operations/add"  
 method="post" modelAttribute="phoneAddDto">  
 <div class="body-details">  
 <div class="block-left">  
 <br>  
 <h3><spring:message code="page.admin.main.info"/></h3>  
 <div class="input-group mb-3 w-25">  
 <table>  
 <tr>  
 <td>  
 <spring:message code="phone.brand"/>  
 </td>  
 <td>  
 <input type="text" name="brand" value="${recentData.brand}"/>  
 <c:if test="${not empty errors.getFieldError('brand')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td>  
 <spring:message code="model"/>  
 </td>  
 <td>  
 <input type="text" name="model" value="${recentData.model}"/>  
 <c:if test="${not empty errors.getFieldError('model')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td>  
 <spring:message code="phone.discount.percent"/>  
 </td>  
 <td>  
 <input type="text" name="discountPercent" value="${recentData.discountPercent}"/>  
 <c:if test="${not empty errors.getFieldError('discountPercent')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td>  
 <spring:message code="phone.image"/>  
 </td>  
 <td>  
 <input type="text" name="imageUrl" value="${recentData.imageUrl}"/>  
 <c:if test="${not empty errors.getFieldError('imageUrl')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td>  
 <spring:message code="price"/>  
 </td>  
 <td>  
 <input type="text" name="price" value="${recentData.price}"/>  
 <c:if test="${not empty errors.getFieldError('price')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td>  
 <spring:message code="quantity"/>  
 </td>  
 <td>  
 <input type="text" name="quantity" value="${recentData.quantity}"/>  
 <c:if test="${not empty errors.getFieldError('quantity')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if>  
 </td>  
 </tr>  
 <tr>  
 <td><spring:message code="page.admin.description"/></td>  
 <td><textarea name="description" rows="4" cols="50"  
 style="resize: none">${recentData.description}</textarea><c:if  
 test="${not empty errors.getFieldError('frontCameraMegapixels')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 </table>  
 <br>  
 <br>  
 </div>  
 </div>  
 <div class="block-right">  
 <h2><spring:message code="display"/></h2>  
 <table>  
 <tr>  
 <td><spring:message code="size"/></td>  
 <td><input type="text" name="displaySizeInches" value="${recentData.displaySizeInches}"/>  
 <c:if test="${not empty errors.getFieldError('displaySizeInches')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td><spring:message code="resolution"/></td>  
 <td><input type="text" name="displayResolution" value="${recentData.displayResolution}"/> px  
 <c:if test="${not empty errors.getFieldError('displayResolution')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td><spring:message code="technology"/></td>  
 <td><input type="text" name="displayTechnology" value="${recentData.displayTechnology}"/>  
 <c:if test="${not empty errors.getFieldError('displayTechnology')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td><spring:message code="pixelDensity"/></td>  
 <td><input type="text" name="pixelDensity" value="${recentData.pixelDensity}"/>  
 <c:if test="${not empty errors.getFieldError('pixelDensity')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 </table>  
 <br>  
 <h2><spring:message code="dimensionAndWeight"/></h2>  
 <table>  
 <tr>  
 <td><spring:message code="length"/></td>  
 <td><input type="text" name="lengthMm" value="${recentData.lengthMm}"/>  
 <c:if test="${not empty errors.getFieldError('lengthMm')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td><spring:message code="width"/></td>  
 <td><input type="text" name="widthMm" value="${recentData.widthMm}"/>  
 <c:if test="${not empty errors.getFieldError('widthMm')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td><spring:message code="color"/></td>  
 <td>  
 <input type="text" name="colors" value="${recentData.colors}"/>  
 <c:if test="${not empty errors.getFieldError('colors')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if>  
 </td>  
 </tr>  
 <tr>  
 <td><spring:message code="weight"/></td>  
 <td><input type="text" name="weightGr" value="${recentData.weightGr}"/>  
 <c:if test="${not empty errors.getFieldError('weightGr')}">  
 <font style="color:red"><spring:message code="page.admin.validation.invalid.field"/></font>  
 </c:if>  
 </tr>  
 </table>  
 <div style="position: absolute; right: 450px; bottom: 0px;">  
 <spring:message code="color.black"/>  
 <br>  
 <spring:message code="color.white"/>  
 <br>  
 <spring:message code="color.gold"/>  
 <br>  
 <spring:message code="color.blue"/>  
 <br>  
 <spring:message code="color.gray"/>  
 <br>  
 <spring:message code="color.red"/>  
 </div>  
 </div>  
 </div>  
 <div style="width: 50%; margin-top: -320px; bottom: 35px">  
 <div class="flex-container">  
 <div class="flex-child">  
 <h2><spring:message code="other"/></h2>  
 <table>  
 <tr>  
 <td><spring:message code="deviceType"/></td>  
 <td><input type="text" name="deviceType" value="${recentData.deviceType}"/>  
 <c:if test="${not empty errors.getFieldError('deviceType')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td><spring:message code="bluetooth"/></td>  
 <td><input type="text" name="bluetooth" value="${recentData.bluetooth}"/>  
 <c:if test="${not empty errors.getFieldError('bluetooth')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 </table>  
 <br>  
 <button class="btn btn-outline-primary" type="submit"><spring:message  
 code="page.admin.add.device"/></button>  
 </div>  
 <div class="flex-child">  
 <h2><spring:message code="battery"/></h2>  
 <table>  
 <tr>  
 <td><spring:message code="talkTime"/></td>  
 <td>  
 <input type="text" name="talkTimeHours" value="${recentData.talkTimeHours}"/>  
 <c:if test="${not empty errors.getFieldError('talkTimeHours')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td><spring:message code="standBy"/></td>  
 <td><input type="text" name="standByTimeHours" value="${recentData.standByTimeHours}"/>  
 <c:if test="${not empty errors.getFieldError('standByTimeHours')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td><spring:message code="batteryCapacity"/></td>  
 <td>  
 <input type="text" name="batteryCapacityMah" value="${recentData.batteryCapacityMah}"/>mAh  
 <c:if test="${not empty errors.getFieldError('battaryCapacityMah')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 </table>  
 </div>  
 <div class="flex-child">  
 <h2><spring:message code="camera"/></h2>  
 <table>  
 <tr>  
 <td><spring:message code="front"/></td>  
 <td><input type="text" name="frontCameraMegapixels"  
 value="${recentData.frontCameraMegapixels}"/>  
 megapixels  
 <c:if test="${not empty errors.getFieldError('frontCameraMegapixels')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td><spring:message code="back"/></td>  
 <td><input type="text" name="backCameraMegapixels"  
 value="${recentData.backCameraMegapixels}"/>  
 megapixels  
 <c:if test="${not empty errors.getFieldError('backCameraMegapixels')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 </table>  
 </div>  
 </div>  
 </div>  
 </form:form>  
 </c:when>  
 <c:otherwise>  
 <form:form name="update-phone-form"  
 action="${pageContext.request.contextPath}/admin/device-operations/update-delete/update/${id}"  
 method="post" modelAttribute="phoneAddDto">  
 <div class="body-details">  
 <div class="block-left">  
 <br>  
 <h3><spring:message code="page.admin.main.info"/></h3>  
 <div class="input-group mb-3 w-25">  
 <table>  
 <tr>  
 <td>  
 <spring:message code="phone.brand"/>  
 </td>  
 <td>  
 <input type="text" name="brand" value="${recentData.brand}"/>  
 <c:if test="${not empty errors.getFieldError('brand')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td>  
 <spring:message code="model"/>  
 </td>  
 <td>  
 <input type="text" name="model" value="${recentData.model}"/>  
 <c:if test="${not empty errors.getFieldError('model')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td>  
 <spring:message code="phone.discount.percent"/>  
 </td>  
 <td>  
 <input type="text" name="discountPercent" value="${recentData.discountPercent}"/>  
 <c:if test="${not empty errors.getFieldError('discountPercent')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td>  
 <spring:message code="phone.image"/>  
 </td>  
 <td>  
 <input type="text" name="imageUrl" value="${recentData.imageUrl}"/>  
 <c:if test="${not empty errors.getFieldError('imageUrl')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td>  
 <spring:message code="price"/>  
 </td>  
 <td>  
 <input type="text" name="price" value="${recentData.price}"/>  
 <c:if test="${not empty errors.getFieldError('price')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td>  
 <spring:message code="quantity"/>  
 </td>  
 <td>  
 <input type="text" name="quantity" value="${quantity.stock}"/>  
 <c:if test="${not empty errors.getFieldError('quantity')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if>  
 </td>  
 </tr>  
 <tr>  
 <td><spring:message code="page.admin.description"/></td>  
 <td><textarea name="description" rows="4" cols="50"  
 style="resize: none">${recentData.description}</textarea><c:if  
 test="${not empty errors.getFieldError('frontCameraMegapixels')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 </table>  
 <br>  
 <br>  
 </div>  
 </div>  
 <div class="block-right">  
 <h2><spring:message code="display"/></h2>  
 <table>  
 <tr>  
 <td><spring:message code="size"/></td>  
 <td><input type="text" name="displaySizeInches" value="${recentData.displaySizeInches}"/>  
 <c:if test="${not empty errors.getFieldError('displaySizeInches')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td><spring:message code="resolution"/></td>  
 <td><input type="text" name="displayResolution" value="${recentData.displayResolution}"/> px  
 <c:if test="${not empty errors.getFieldError('displayResolution')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td><spring:message code="technology"/></td>  
 <td><input type="text" name="displayTechnology" value="${recentData.displayTechnology}"/>  
 <c:if test="${not empty errors.getFieldError('displayTechnology')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td><spring:message code="pixelDensity"/></td>  
 <td><input type="text" name="pixelDensity" value="${recentData.pixelDensity}"/>  
 <c:if test="${not empty errors.getFieldError('pixelDensity')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 </table>  
 <br>  
 <h2><spring:message code="dimensionAndWeight"/></h2>  
 <table>  
 <tr>  
 <td><spring:message code="length"/></td>  
 <td><input type="text" name="lengthMm" value="${recentData.lengthMm}"/>  
 <c:if test="${not empty errors.getFieldError('lengthMm')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td><spring:message code="width"/></td>  
 <td><input type="text" name="widthMm" value="${recentData.widthMm}"/>  
 <c:if test="${not empty errors.getFieldError('widthMm')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td><spring:message code="color"/></td>  
 <td>  
 <input type="text" name="colors" value="${colors}"/>  
 <c:if test="${not empty errors.getFieldError('colors')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if>  
 </td>  
 </tr>  
 <tr>  
 <td><spring:message code="weight"/></td>  
 <td><input type="text" name="weightGr" value="${recentData.weightGr}"/>  
 <c:if test="${not empty errors.getFieldError('weightGr')}">  
 <font style="color:red"><spring:message code="page.admin.validation.invalid.field"/></font>  
 </c:if>  
 </tr>  
 </table>  
 </div>  
 </div>  
 <div style="width: 50%; margin-top: -320px; bottom: 35px">  
 <div class="flex-container">  
 <div class="flex-child">  
 <h2><spring:message code="other"/></h2>  
 <table>  
 <tr>  
 <td><spring:message code="deviceType"/></td>  
 <td><input type="text" name="deviceType" value="${recentData.deviceType}"/>  
 <c:if test="${not empty errors.getFieldError('deviceType')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td><spring:message code="bluetooth"/></td>  
 <td><input type="text" name="bluetooth" value="${recentData.bluetooth}"/>  
 <c:if test="${not empty errors.getFieldError('bluetooth')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 </table>  
 <br>  
 <button class="btn btn-outline-primary" type="submit"><spring:message  
 code="page.admin.update.device"/></button>  
 </div>  
 <div class="flex-child">  
 <h2><spring:message code="battery"/></h2>  
 <table>  
 <tr>  
 <td><spring:message code="talkTime"/></td>  
 <td>  
 <input type="text" name="talkTimeHours" value="${recentData.talkTimeHours}"/>  
 <c:if test="${not empty errors.getFieldError('talkTimeHours')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td><spring:message code="standBy"/></td>  
 <td><input type="text" name="standByTimeHours" value="${recentData.standByTimeHours}"/>  
 <c:if test="${not empty errors.getFieldError('standByTimeHours')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td><spring:message code="batteryCapacity"/></td>  
 <td>  
 <input type="text" name="batteryCapacityMah" value="${recentData.batteryCapacityMah}"/>mAh  
 <c:if test="${not empty errors.getFieldError('battaryCapacityMah')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 </table>  
 </div>  
 <div class="flex-child">  
 <h2><spring:message code="camera"/></h2>  
 <table>  
 <tr>  
 <td><spring:message code="front"/></td>  
 <td><input type="text" name="frontCameraMegapixels"  
 value="${recentData.frontCameraMegapixels}"/>  
 megapixels  
 <c:if test="${not empty errors.getFieldError('frontCameraMegapixels')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 <tr>  
 <td><spring:message code="back"/></td>  
 <td><input type="text" name="backCameraMegapixels"  
 value="${recentData.backCameraMegapixels}"/>  
 megapixels  
 <c:if test="${not empty errors.getFieldError('backCameraMegapixels')}">  
 <font style="color:red"><spring:message  
 code="page.admin.validation.invalid.field"/></font>  
 </c:if></td>  
 </tr>  
 </table>  
 </div>  
 </div>  
 </div>  
 </form:form>  
 </c:otherwise>  
 </c:choose>  
 <br>  
 <br>  
 <br>  
 </body>  
</tags:masterLogged>

Файл admin-main.jsp

<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>  
<%@ taglib prefix="tags" tagdir="/WEB-INF/tags" %>  
<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>  
<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form" %>  
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn" %>  
<tags:masterLogged pageTitle="Main Admin">  
 <div style="text-align: center; margin-top: 0">  
 <img style="width: 200px;height: 200px" src="${pageContext.servletContext.contextPath}/images/phoneshop-logo.jpg"  
 alt="Phoneshop Logo"/>  
 <h2>  
 <br>  
 <spring:message code="page.admin.greetings1"/>  
 <b><c:out value="${username}"/>.</b>  
 <br>  
 <spring:message code="page.admin.greeting2"/>  
 </h2>  
 </div>  
 <ul class="admin-list">  
 <li><a class="btn btn-outline-primary" href="${pageContext.servletContext.contextPath}/admin/orders"><spring:message  
 code="page.admin.order.management"/></a></li>  
 <li><a class="btn btn-outline-primary"  
 href="${pageContext.servletContext.contextPath}/admin/user-management"><spring:message  
 code="page.admin.users"/></a></li>  
 <li><a class="btn btn-outline-primary"  
 href="${pageContext.servletContext.contextPath}/admin/device-operations/add">  
 <spring:message code="page.admin.add.device"/></a>  
 <a class="btn btn-outline-primary"  
 href="${pageContext.servletContext.contextPath}/admin/device-operations/update-delete"><spring:message  
 code="page.admin.add.phone"/></a>  
 </li>  
 </ul>  
</tags:masterLogged>

Файл cart.jsp

<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>  
<%@ taglib prefix="tags" tagdir="/WEB-INF/tags" %>  
<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>  
<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form" %>  
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn" %>  
<tags:master pageTitle="Cart">  
 <head>  
 <title><spring:message code="cart.page"/></title>  
 </head>  
 <body>  
 <br>  
 <form method="get" action="${pageContext.servletContext.contextPath}/product-list">  
 <button class="btn btn-outline-primary">  
 <spring:message code="button.back.to.productList"/>  
 </button>  
 </form>  
 <h2><spring:message code="cart"/></h2>  
 <div id="error-result">  
 <c:if test="${error eq true}">  
 <spring:message code="cart.update.error"/>  
 </c:if>  
 </div>  
 <div id="error-result">  
 <c:if test="${empty cart}">  
 <spring:message code="cart.empty.msg"/>  
 </c:if>  
 </div>  
 <div id="success-result">  
 <c:if test="${successUpdate eq true}">  
 <spring:message code="cart.update.success"/>  
 </c:if>  
 </div>  
 <div id="success-result">  
 <c:if test="${successDelete eq true}">  
 <spring:message code="cart.delete.success"/>  
 </c:if>  
 </div>  
 <form:form id="update-form" method="post" action="${pageContext.servletContext.contextPath}/cart"  
 modelAttribute="phoneArrayDto">  
 <table>  
 <thead>  
 <tr>  
 <td>  
 <spring:message code="phone.brand"/>  
 </td>  
 <td>  
 <spring:message code="model"/>  
 </td>  
 <td>  
 <spring:message code="color"/>  
 </td>  
 <td>  
 <spring:message code="phone.display.size"/>  
 </td>  
 <td>  
 <spring:message code="price"/>  
 </td>  
 <td>  
 <spring:message code="quantity"/>  
 </td>  
 <td>  
 <spring:message code="action"/>  
 </td>  
 </tr>  
 </thead>  
 <c:forEach var="cartItem" items="${cart.cartItems}" varStatus="status">  
 <tr>  
 <td>  
 <c:out value="${cartItem.phone.brand}"/>  
 </td>  
 <td>  
 <c:out value="${cartItem.phone.model}"/>  
 </td>  
 <td>  
 <c:choose>  
 <c:when test="${empty cartItem.phone.colors}">  
 <span class="result-error">  
 no info  
 </span>  
 </c:when>  
 <c:otherwise>  
 <c:forEach var="color" items="${cartItem.phone.colors}">  
 <span class="dot" style="background-color: ${color.code}"></span>  
 </c:forEach>  
 </c:otherwise>  
 </c:choose>  
 </td>  
 <td>  
 <c:out value="${cartItem.phone.displaySizeInches}"/>"  
 </td>  
 <td>  
 <c:out value="${cartItem.phone.actualPrice}"/> <spring:message code="currency"/>  
 </td>  
 <td>  
 <input class="quantity-input" type="text" id="quantity${cartItem.phone.id}" name="quantity"  
 value="${not empty errorsId && errorsId.contains(cartItem.phone.id) ? paramValues['cartItem.quantity'][status.index] : cartItem.quantity }"/>  
 <div class="result-error" id="result${cartItem.phone.id}"></div>  
 <c:if test="${fn:contains(errorsId, cartItem.phone.id)}">  
 <div class="result-error">  
 <spring:message code="cart.wrong.input.or.out.of.stock"/>  
 </div>  
 </c:if>  
 <input id="phoneId${cartItem.phone.id}" name="phoneId" type="hidden"  
 value="${cartItem.phone.id}"/>  
 </td>  
 <td>  
 <button class="btn btn-danger"  
 formaction="${pageContext.servletContext.contextPath}/cart/${cartItem.phone.id}">  
 <spring:message code="cart.delete.btn"/>  
 </button>  
 </td>  
 </tr>  
 </c:forEach>  
 </table>  
 </form:form>  
 <button form="update-form" class="btn btn-secondary">  
 <spring:message code="cart.update.btn"/>  
 </button>  
 <button form="goToOrder" class="btn btn-secondary">  
 <spring:message code="order.make.btn"/>  
 </button>  
  
 <form id="goToOrder" action="${pageContext.servletContext.contextPath}/order" method="get"/>  
 </body>  
</tags:master>

Файл contacts-page.jsp

<%@ page contentType="text/html;charset=UTF-8" %>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>  
<%@ taglib prefix="tags" tagdir="/WEB-INF/tags" %>  
<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>  
<%@ taglib prefix="sping" uri="http://java.sun.com/jsp/jstl/fmt" %>  
  
<tags:master pageTitle="Contacts page">  
 <body>  
 <br>  
 <br>  
 <table id="myTable">  
 <tr>  
 <td style="border: none">  
 <spring:message code="page.contacts.mobile.number.text"/>  
 </td>  
 <td style="border: none">  
 <strong><spring:message code="page.contacts.mobile.number"/></strong>  
 </td>  
 </tr>  
 <tr>  
 <td style="border: none"><spring:message code="page.contacts.stationar.number.text"/></td>  
 <td style="border: none"><strong><spring:message code="page.contacts.stationar.number"/></strong></td>  
 </tr>  
 <tr>  
 <td style="border: none"><spring:message code="page.contacts.address.text"/></td>  
 <td style="border: none"><strong><spring:message code="page.contacts.address"/></strong></td>  
 </tr>  
 <tr>  
 <td style="border: none"><spring:message code="page.contacts.email.text"/></td>  
 <td style="border: none"><strong><spring:message code="page.contacts.email"/></strong></td>  
 </tr>  
 <tr>  
 <td style="border: none"><spring:message code="page.contacts.telegram.text"/></td>  
 <td style="border: none"><strong><spring:message code="page.contacts.telegram"/></strong></td>  
 </tr>  
 </table>  
 <br>  
 </body>  
</tags:master>

Файл error.jsp

<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<%@ taglib prefix="tags" tagdir="/WEB-INF/tags" %>  
<tags:master pageTitle="404">  
<html>  
<head>  
 <title><spring:message code="error.title"/></title>  
</head>  
<body>  
<c:if test="${status eq '404'}">  
 <h3><spring:message code="page.not.found"/></h3>  
</c:if>  
<c:if test="${status eq '500'}">  
 <h3><spring:message code="internal.server.error"/></h3>  
</c:if>  
<c:if test="${status eq '403'}">  
 <h3><spring:message code="access.forbidden"/></h3>  
</c:if>  
<br>  
<spring:message code="contact.admin"/> <a href="${adminMail}"><spring:message code="admin.mail"/></a>  
</body>  
</html>  
</tags:master>

Файл hot-prices.jsp

<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ taglib prefix="tags" tagdir="/WEB-INF/tags" %>  
<%@ taglib uri="http://www.springframework.org/tags/form" prefix="form" %>  
<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>  
  
<tags:master pageTitle="Product List">  
 <head>  
 <title><spring:message code="title"/></title>  
 <script src="https://code.jquery.com/jquery-1.8.3.js"></script>  
 </head>  
 <body>  
 <p>  
 <div id="tableContainer-1">  
 <div id="tableContainer-2">  
 <tags:phone-list/>  
 </div>  
 </div>  
 </body>  
</tags:master>

Файл login.jsp

<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<%@ taglib prefix="tags" tagdir="/WEB-INF/tags" %>  
<tags:master pageTitle="404">  
 <html>  
 <head>  
 <title><spring:message code="loginPage"/></title>  
 </head>  
 <body>  
 <br>  
 <div id="error-result">  
 <c:if test="${not empty error}">  
 ${error}  
 </c:if>  
 </div>  
 <div id="success-result">  
 <c:if test="${not empty isRegistered}">  
 <spring:message code="page.registration.register.success"/>  
 </c:if>  
 </div>  
 <br>  
 <form action="${pageContext.servletContext.contextPath}/login" method="post">  
 <table>  
 <tr>  
 <td style="border:none"><spring:message code="page.login.username"/></td>  
 <td style="border:none">  
 <input type="text" name="username"/>  
 </td>  
 </tr>  
 <tr>  
 <td style="border:none"><spring:message code="page.login.password"/></td>  
 <td style="border:none">  
 <input type="password" name="password"/>  
 </td>  
 </tr>  
 </table>  
 <br>  
 <div>  
 <button class="btn btn-outline-primary" type="submit"><spring:message code="page.login"/></button>  
 <br>  
 <spring:message code="not.registered.yet"/>  
 </div>  
 </form>  
 </body>  
 </html>  
</tags:master>

Файл order.jsp

<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>  
<%@ taglib prefix="tags" tagdir="/WEB-INF/tags" %>  
<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>  
<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form" %>  
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn" %>  
  
<tags:master pageTitle="Order confirmation">  
 <head>  
 <title><spring:message code="page.order.confirmation"/></title>  
 </head>  
 <body>  
 <br>  
 <h2>  
 <spring:message code="order.page"/>  
 </h2>  
 <form method="get" action="${pageContext.servletContext.contextPath}/cart">  
 <button class="btn btn-outline-primary">  
 <spring:message code="button.back.to.cartPage"/>  
 </button>  
 </form>  
 <div id="error-result">  
 <c:if test="${error eq true}">  
 <spring:message code="order.error"/>  
 </c:if>  
 </div>  
 <div id="error-result">  
 <c:if test="${cart.cartItems.size() eq 0 }">  
 <spring:message code="cart.empty.msg"/>  
 </c:if>  
 </div>  
 <table class="table-bordered w-50">  
 <tr>  
 <td><spring:message code="phone.brand"/></td>  
 <td><spring:message code="model"/></td>  
 <td><spring:message code="color"/></td>  
 <td><spring:message code="phone.display.size"/></td>  
 <td><spring:message code="quantity"/></td>  
 <td><spring:message code="price"/></td>  
 </tr>  
 <c:forEach var="cartItem" items="${cart.cartItems}">  
 <tr>  
 <td><c:out value="${cartItem.phone.brand}"/></td>  
 <td><c:out value="${cartItem.phone.model}"/></td>  
 <td>  
 <c:choose>  
 <c:when test="${empty cartItem.phone.colors}">  
 <span class="result-error">  
 no info  
 </span>  
 </c:when>  
 <c:otherwise>  
 <c:forEach var="color" items="${cartItem.phone.colors}">  
 <span class="dot" style="background-color: ${color.code}"></span>  
 </c:forEach>  
 </c:otherwise>  
 </c:choose>  
 </td>  
 <td><c:out value="${cartItem.phone.displaySizeInches}"/>"</td>  
 <td><c:out value="${cartItem.quantity}"/></td>  
 <td><c:out value="${cartItem.price}"/> <spring:message code="currency"/> \* ${cartItem.quantity}</td>  
 </tr>  
 </c:forEach>  
 <tr>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td><spring:message code="order.subtotal"/></td>  
 <td><c:out value="${cart.totalCost}"/> <spring:message code="currency"/></td>  
 </tr>  
 <tr>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td><spring:message code="order.delivery"/></td>  
 <td><c:out value="${deliveryPrice}"/> <spring:message code="currency"/></td>  
 </tr>  
 <tr>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td><spring:message code="order.total"/></td>  
 <td><c:out value="${totalCost}"/> <spring:message code="currency"/></td>  
 </tr>  
 </table>  
 <br>  
 <br>  
 <form:form id="place-order" method="post" action="${pageContext.servletContext.contextPath}/order"  
 modelAttribute="orderDataDto">  
 <table class="table table-bordered table-borderless w-25">  
 <tr>  
 <td>  
 <spring:message code="order.data.firstName"/>  
 </td>  
 <td>  
 <input type="text" name="firstName"/>  
 <tags:error name="firstName" errors="${errors}"/>  
 </td>  
 </tr>  
 <tr>  
 <td>  
 <spring:message code="order.data.lastName"/>  
 </td>  
 <td>  
 <input type="text" name="lastName"/>  
 <tags:error name="lastName" errors="${errors}"/>  
 </td>  
 </tr>  
 <tr>  
 <td>  
 <spring:message code="order.data.address"/>  
 </td>  
 <td>  
 <input type="text" name="address"/>  
 <tags:error name="address" errors="${errors}"/>  
 </td>  
 </tr>  
 <tr>  
 <td>  
 <spring:message code="order.data.phone"/>  
 </td>  
 <td>  
 <input type="text" name="phone"/>  
 <tags:error name="phone" errors="${errors}"/>  
 </td>  
 </tr>  
 <tr>  
 <td>  
 <spring:message code="order.data.additionalInfo"/>  
 </td>  
 <td>  
 <textarea placeholder="Additional information" aria-label="Additional information"  
 name="additionalInfo"></textarea>  
 </td>  
 </tr>  
 </table>  
 </form:form>  
 <button form="place-order" class="btn btn-outline-primary">  
 <spring:message code="order.make.btn"/>  
 </button>  
 </body>  
</tags:master>

Файл order-overview.jsp

<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>  
<%@ taglib prefix="tags" tagdir="/WEB-INF/tags" %>  
<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>  
<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form" %>  
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn" %>  
  
<tags:master pageTitle="Order overview">  
 <head>  
 <title><spring:message code="page.order.overview"/></title>  
 </head>  
 <body>  
 <br>  
 <h2>  
 <spring:message code="order.page"/> ${id} <spring:message code="admin.status"/>:  
 <c:if test="${order.status.name() eq 'NEW'}">  
 <spring:message code="order.new"/>  
 </c:if>  
 <c:if test="${order.status.name() eq 'DELIVERED'}">  
 <spring:message code="admin.button.delivered"/>  
 </c:if>  
 <c:if test="${order.status.name() eq 'IN\_DELIVERY'}">  
 <spring:message code="admin.button.in.delivery"/>  
 </c:if>  
 <c:if test="${order.status.name() eq 'REJECTED'}">  
 <spring:message code="admin.button.rejected"/>  
 </c:if>  
 </h2>  
 <table class="table-bordered w-50">  
 <tr>  
 <td><spring:message code="phone.brand"/></td>  
 <td><spring:message code="model"/></td>  
 <td><spring:message code="color"/></td>  
 <td><spring:message code="phone.display.size"/></td>  
 <td><spring:message code="quantity"/></td>  
 <td><spring:message code="price"/></td>  
 </tr>  
 <c:forEach var="orderItem" items="${order.orderItems}">  
 <tr>  
 <td><c:out value="${orderItem.phone.brand}"/></td>  
 <td><c:out value="${orderItem.phone.model}"/></td>  
 <td>  
 <c:choose>  
 <c:when test="${empty orderItem.phone.colors}">  
 <span class="result-error">  
 no info  
 </span>  
 </c:when>  
 <c:otherwise>  
 <c:forEach var="color" items="${orderItem.phone.colors}">  
 <span class="dot" style="background-color: ${color.code}"></span>  
 </c:forEach>  
 </c:otherwise>  
 </c:choose>  
 </td>  
 <td><c:out value="${orderItem.phone.displaySizeInches}"/>"</td>  
 <td><c:out value="${orderItem.quantity}"/></td>  
 <td><c:out value="${orderItem.phone.actualPrice.multiply(orderItem.quantity)}"/> <spring:message code="currency"/></td>  
 </tr>  
 </c:forEach>  
 <tr>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td><spring:message code="order.subtotal"/></td>  
 <td><c:out value="${order.subtotal}"/> <spring:message code="currency"/></td>  
 </tr>  
 <tr>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td><spring:message code="order.delivery"/></td>  
 <td><c:out value="${order.deliveryPrice}"/> <spring:message code="currency"/></td>  
 </tr>  
 <tr>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td><spring:message code="order.total"/></td>  
 <td><c:out value="${order.totalPrice}"/> <spring:message code="currency"/></td>  
 </tr>  
 </table>  
 <br>  
 <br>  
 <table class="table table-bordered table-borderless w-25">  
 <tr>  
 <td>  
 <spring:message code="order.data.firstName"/>  
 </td>  
 <td>  
 <c:out value="${order.firstName}"/>  
 </td>  
 </tr>  
 <tr>  
 <td>  
 <spring:message code="order.data.lastName"/>  
 </td>  
 <td>  
 <c:out value="${order.lastName}"/>  
 </td>  
 </tr>  
 <tr>  
 <td>  
 <spring:message code="order.data.address"/>  
 </td>  
 <td>  
 <c:out value="${order.deliveryAddress}"/>  
 </td>  
 </tr>  
 <tr>  
 <td>  
 <spring:message code="order.data.phone"/>  
 </td>  
 <td>  
 <c:out value="${order.contactPhoneNo}"/>  
 </td>  
 </tr>  
 <tr>  
 <td>  
 <spring:message code="order.data.additionalInfo"/>  
 </td>  
 <td>  
 <c:out value="${order.additionalInfo}"/>  
 </td>  
 </tr>  
 </table>  
 <form action="${pageContext.servletContext.contextPath}/product-list">  
 <button class="btn btn-outline-primary">  
 <spring:message code="button.back.to.productList"/>  
 </button>  
 </form>  
 </body>  
</tags:master>

Файл order-overview-admin.jsp

<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>  
<%@ taglib prefix="tags" tagdir="/WEB-INF/tags" %>  
<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>  
<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form" %>  
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn" %>  
  
<tags:masterLogged pageTitle="Order overview">  
 <head>  
 <title><spring:message code="page.order.overview"/></title>  
 </head>  
 <br>  
 <h2>  
 <spring:message code="order.page"/> ${id}, <spring:message code="admin.status"/>:  
 <c:if test="${order.status.name() eq 'NEW'}">  
 <spring:message code="order.new"/>  
 </c:if>  
 <c:if test="${order.status.name() eq 'DELIVERED'}">  
 <spring:message code="admin.button.delivered"/>  
 </c:if>  
 <c:if test="${order.status.name() eq 'IN\_DELIVERY'}">  
 <spring:message code="admin.button.in.delivery"/>  
 </c:if>  
 <c:if test="${order.status.name() eq 'REJECTED'}">  
 <spring:message code="admin.button.rejected"/>  
 </c:if>  
 </h2>  
 <table class="table-bordered w-50">  
 <tr>  
 <td><spring:message code="phone.brand"/></td>  
 <td><spring:message code="model"/></td>  
 <td><spring:message code="color"/></td>  
 <td><spring:message code="phone.display.size"/></td>  
 <td><spring:message code="quantity"/></td>  
 <td><spring:message code="price"/></td>  
 </tr>  
 <c:forEach var="orderItem" items="${order.orderItems}">  
 <tr>  
 <td><c:out value="${orderItem.phone.brand}"/></td>  
 <td><c:out value="${orderItem.phone.model}"/></td>  
 <td>  
 <c:choose>  
 <c:when test="${empty orderItem.phone.colors}">  
 <span class="result-error">  
 no info  
 </span>  
 </c:when>  
 <c:otherwise>  
 <c:forEach var="color" items="${orderItem.phone.colors}">  
 <c:out value="${color.code},"/>  
 </c:forEach>  
 </c:otherwise>  
 </c:choose>  
 </td>  
 <td><c:out value="${orderItem.phone.displaySizeInches}"/>"</td>  
 <td><c:out value="${orderItem.quantity}"/></td>  
 <td><c:out value="${orderItem.phone.price.multiply(orderItem.quantity)}"/> <spring:message  
 code="currency"/></td>  
 </tr>  
 </c:forEach>  
 <tr>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td><spring:message code="order.subtotal"/></td>  
 <td><c:out value="${order.subtotal} $"/></td>  
 </tr>  
 <tr>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td><spring:message code="order.delivery"/></td>  
 <td><c:out value="${order.deliveryPrice} $"/></td>  
 </tr>  
 <tr>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td style="border: none!important;"></td>  
 <td><spring:message code="order.total"/></td>  
 <td><c:out value="${order.totalPrice} $"/></td>  
 </tr>  
 </table>  
 <br>  
 <br>  
 <table class="table table-bordered table-borderless w-25">  
 <tr>  
 <td>  
 <spring:message code="order.data.firstName"/>  
 </td>  
 <td>  
 <c:out value="${order.firstName}"/>  
 </td>  
 </tr>  
 <tr>  
 <td>  
 <spring:message code="order.data.lastName"/>  
 </td>  
 <td>  
 <c:out value="${order.lastName}"/>  
 </td>  
 </tr>  
 <tr>  
 <td>  
 <spring:message code="order.data.address"/>  
 </td>  
 <td>  
 <c:out value="${order.deliveryAddress}"/>  
 </td>  
 </tr>  
 <tr>  
 <td>  
 <spring:message code="order.data.phone"/>  
 </td>  
 <td>  
 <c:out value="${order.contactPhoneNo}"/>  
 </td>  
 </tr>  
 <tr>  
 <td>  
 <spring:message code="order.data.additionalInfo"/>  
 </td>  
 <td>  
 <c:out value="${order.additionalInfo}"/>  
 </td>  
 </tr>  
 </table>  
 <form id="change-status-delivered" action="${pageContext.servletContext.contextPath}/admin/orders/${order.id}">  
 <input name="orderId" type="hidden" value="${order.id}">  
 <input type="hidden" name="orderStatus" value="DELIVERED">  
 </form>  
  
 <form id="change-status-rejected" action="${pageContext.servletContext.contextPath}/admin/orders/${order.id}">  
 <input name="orderId" type="hidden" value="${order.id}">  
 <input type="hidden" name="orderStatus" value="REJECTED">  
 </form>  
  
 <form id="change-status-in-delivery"  
 action="${pageContext.servletContext.contextPath}/admin/orders/${order.id}">  
 <input name="orderId" type="hidden" value="${order.id}">  
 <input type="hidden" name="orderStatus" value="IN\_DELIVERY">  
 </form>  
 <c:choose>  
 <c:when test="${order.status.name().equals('NEW')}">  
 <spring:message code="page.admin.change.order.status"/>  
 <button form="change-status-rejected" formmethod="post" class="btn btn-outline-danger">  
 <spring:message code="admin.button.rejected"/>  
 </button>  
 <button form="change-status-in-delivery" formmethod="post" class="btn btn-outline-primary">  
 <spring:message code="admin.button.in.delivery"/>  
 </button>  
 <button disabled form="change-status-delivered" formmethod="post" class="btn btn-outline-success">  
 <spring:message code="admin.button.delivered"/>  
 </button>  
 </c:when>  
  
 <c:when test="${order.status.name().equals('DELIVERED')}">  
 <button disabled form="change-status-rejected" formmethod="post" class="btn btn-outline-danger">  
 <spring:message code="admin.button.rejected"/>  
 </button>  
 <button disabled form="change-status-in-delivery" formmethod="post" class="btn btn btn-outline-primary">  
 <spring:message code="admin.button.in.delivery"/>  
 </button>  
 <button disabled form="change-status-delivered" formmethod="post" class="btn btn-success">  
 <spring:message code="admin.button.delivered"/>  
 </button>  
 </c:when>  
  
 <c:when test="${order.status.name().equals('REJECTED')}">  
 <button disabled form="change-status-rejected" formmethod="post" class="btn btn btn-danger">  
 <spring:message code="admin.button.rejected"/>  
 </button>  
 <button disabled form="change-status-in-delivery" formmethod="post" class="btn btn btn-outline-primary">  
 <spring:message code="admin.button.in.delivery"/>  
 </button>  
 <button disabled form="change-status-delivered" formmethod="post" class="btn btn-outline-success">  
 <spring:message code="admin.button.delivered"/>  
 </button>  
 </c:when>  
  
 <c:when test="${order.status.name().equals('IN\_DELIVERY')}">  
 <spring:message code="page.admin.change.order.status"/>  
 <button disabled form="change-status-rejected" formmethod="post" class="btn btn btn-outline-danger">  
 <spring:message code="admin.button.rejected"/>  
 </button>  
 <button disabled form="change-status-in-delivery" formmethod="post" class="btn btn btn-primary">  
 <spring:message code="admin.button.in.delivery"/>  
 </button>  
 <button form="change-status-delivered" formmethod="post" class="btn btn-outline-success">  
 <spring:message code="admin.button.delivered"/>  
 </button>  
 </c:when>  
 </c:choose>  
 <br>  
 <form id="back-form"></form>  
 <button form="back-form" formmethod="get" formaction="${pageContext.servletContext.contextPath}/admin/orders"  
 class="btn btn-secondary">  
 <spring:message code="admin.button.back"/>  
 </button>  
 <br>  
</tags:masterLogged>

Файл order.jsp

<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<%@ taglib prefix="tags" tagdir="/WEB-INF/tags" %>  
<tags:masterLogged pageTitle="Orders">  
 <html>  
 <head>  
 <title><spring:message code="admin.ordersPage"/></title>  
 </head>  
 <body>  
 <h2>  
 <spring:message code="admin.ordersPage"/>:  
 </h2>  
 <table>  
 <tr>  
 <td>  
 <spring:message code="admin.orderNumber"/>  
 </td>  
 <td>  
 <spring:message code="admin.customer"/>  
 </td>  
 <td>  
 <spring:message code="admin.phoneNum"/>  
 </td>  
 <td>  
 <spring:message code="admin.address"/>  
 </td>  
 <td>  
 <spring:message code="admin.date"/>  
 </td>  
 <td>  
 <spring:message code="admin.totalPrice"/>  
 </td>  
 <td>  
 <spring:message code="admin.status"/>  
 </td>  
 </tr>  
 <c:forEach var="order" items="${orderList}">  
 <tr>  
 <td>  
 <a href="${pageContext.servletContext.contextPath}/admin/orders/${order.id}">  
 <c:out value="${order.id}"/>  
 </a>  
 </td>  
 <td>  
 <c:out value="${order.firstName}"/>  
 <c:out value="${order.lastName}"/>  
 </td>  
 <td>  
 <c:out value="${order.contactPhoneNo}"/>  
 </td>  
 <td>  
 <c:out value="${order.deliveryAddress}"/>  
 </td>  
 <td>  
 <c:out value="${order.date.toString()}"/>  
 </td>  
 <td>  
 <c:out value="${order.totalPrice}"/>  
 </td>  
 <td>  
 <c:if test="${order.status.name() eq 'NEW'}">  
 <spring:message code="order.new"/>  
 </c:if>  
 <c:if test="${order.status.name() eq 'DELIVERED'}">  
 <spring:message code="admin.button.delivered"/>  
 </c:if>  
 <c:if test="${order.status.name() eq 'IN\_DELIVERY'}">  
 <spring:message code="admin.button.in.delivery"/>  
 </c:if>  
 <c:if test="${order.status.name() eq 'REJECTED'}">  
 <spring:message code="admin.button.rejected"/>  
 </c:if>  
 </td>  
 </tr>  
 </c:forEach>  
 </table>  
 </body>  
 </html>  
</tags:masterLogged>

Файл product-details.jsp

<%@ page contentType="text/html;charset=UTF-8" %>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>  
<%@ taglib prefix="tags" tagdir="/WEB-INF/tags" %>  
<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>  
  
<tags:master pageTitle="Product Details">  
 <head>  
 <script src="https://code.jquery.com/jquery-1.8.3.js"></script>  
 <title><spring:message code="titleDetails"/></title>  
 </head>  
 <body>  
 <div class="body-details">  
 <div class="block-left">  
 <br>  
 <form action="${pageContext.servletContext.contextPath}/product-list" method="get">  
 <button class="btn btn-outline-primary"><spring:message code="button.back.to.productList"/></button>  
 </form>  
 <div id="success-result">  
 </div>  
 <div id="error-result">  
 </div>  
 <div id="ajax-errors">  
 </div>  
 <h3>  
 <strong>${phone.model}</strong>  
 </h3>  
 <img class="img" src="${phone.imageUrl}">  
 <br>  
 <br>  
 <textarea disabled readonly rows="4" cols="50" translate="yes">${phone.description}</textarea>  
 <br>  
 <br>  
 <h3>  
 <spring:message code="price"/>:<c:out value="${phone.price}"/>  
 <c:if test="${phone.discountPercent != 0}">  
 - <c:out value="${phone.discountPercent}"/>%  
 <br>  
  
 <div style="color:green">  
 <spring:message code="currency"/> <c:out value="${phone.actualPrice}"/>  
 </div>  
 </c:if>  
 </h3>  
 <div class="form-inline my-2 my-lg-0 mr-auto">  
 <input type="text" id="quantity${phone.id}" name="quantity" class="form-control mr-sm-2"  
 placeholder="quantity" aria-label="quantity" aria-describedby="basic-addon2"  
 value="1">  
 <input id="phoneId${phone.id}" name="phoneId" type="hidden" value="${phone.id}"/>  
 <div class="input-group-append">  
 <button onclick="addToCart(${phone.id})" class="btn btn-outline-primary"><spring:message  
 code="addToCart"/></button>  
 </div>  
 <div class="result-error" id="result${phone.id}"></div>  
 </div>  
 </div>  
 <div class="block-right">  
 <h2><spring:message code="display"/></h2>  
 <table>  
 <tr>  
 <td><spring:message code="size"/></td>  
 <td>${phone.displaySizeInches}</td>  
 </tr>  
 <tr>  
 <td><spring:message code="resolution"/></td>  
 <td>${phone.displayResolution}</td>  
 </tr>  
 <tr>  
 <td><spring:message code="technology"/></td>  
 <td>${phone.displayTechnology}</td>  
 </tr>  
 <tr>  
 <td><spring:message code="pixelDensity"/></td>  
 <td>${phone.pixelDensity}</td>  
 </tr>  
 </table>  
 <h2><spring:message code="dimensionAndWeight"/></h2>  
 <table>  
 <tr>  
 <td><spring:message code="length"/></td>  
 <td>${phone.lengthMm}mm</td>  
 </tr>  
 <tr>  
 <td><spring:message code="width"/></td>  
 <td>${phone.widthMm}mm</td>  
 </tr>  
 <tr>  
 <td><spring:message code="color"/></td>  
 <td>  
 <c:choose>  
 <c:when test="${empty phone.colors}">  
 <span class="result-error">  
 no info  
 </span>  
 </c:when>  
 <c:otherwise>  
 <c:forEach var="color" items="${phone.colors}">  
 <span class="dot" style="background-color: ${color.code}"></span>  
 </c:forEach>  
 </c:otherwise>  
 </c:choose>  
 </td>  
 </tr>  
 <tr>  
 <td><spring:message code="weight"/></td>  
 <td>${phone.weightGr}</td>  
 </tr>  
 </table>  
 <h2><spring:message code="camera"/></h2>  
 <table>  
 <tr>  
 <td><spring:message code="front"/></td>  
 <td>${phone.frontCameraMegapixels} megapixels</td>  
 </tr>  
 <tr>  
 <td><spring:message code="back"/></td>  
 <td>${phone.backCameraMegapixels} megapixels</td>  
 </tr>  
 </table>  
 <h2><spring:message code="battery"/></h2>  
 <table>  
 <tr>  
 <td><spring:message code="talkTime"/></td>  
 <td>  
 <c:choose>  
 <c:when test="${empty phone.talkTimeHours}">  
 <span class="result-error">  
 no info  
 </span>  
 </c:when>  
 <c:otherwise>  
 ${phone.talkTimeHours} часов  
 </c:otherwise>  
 </c:choose>  
 </td>  
 </tr>  
 <tr>  
 <td><spring:message code="standBy"/></td>  
 <td>  
 <c:choose>  
 <c:when test="${empty phone.standByTimeHours}">  
 <span class="result-error">  
 no info  
 </span>  
 </c:when>  
 <c:otherwise>  
 ${phone.standByTimeHours} часов  
 </c:otherwise>  
 </c:choose>  
 </td>  
 </tr>  
 <tr>  
 <td><spring:message code="batteryCapacity"/></td>  
 <td>  
 <c:choose>  
 <c:when test="${empty phone.batteryCapacityMah}">  
 <span class="result-error">  
 no info  
 </span>  
 </c:when>  
 <c:otherwise>  
 ${phone.batteryCapacityMah}mAh  
 </c:otherwise>  
 </c:choose>  
 </td>  
 </tr>  
 </table>  
 <h2><spring:message code="other"/></h2>  
 <table>  
 <tr>  
 <td><spring:message code="deviceType"/></td>  
 <td>${phone.deviceType}</td>  
 </tr>  
 <tr>  
 <td><spring:message code="bluetooth"/></td>  
 <td>${phone.bluetooth}</td>  
 </tr>  
 </table>  
 </div>  
 </div>  
 </body>  
</tags:master>

Файл product-list.jsp

<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ taglib prefix="tags" tagdir="/WEB-INF/tags" %>  
<%@ taglib uri="http://www.springframework.org/tags/form" prefix="form" %>  
<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>  
  
<tags:master pageTitle="Product List">  
 <head>  
 <title><spring:message code="title"/></title>  
 <script src="https://code.jquery.com/jquery-1.8.3.js"></script>  
 </head>  
 <body>  
 <p>  
 <div class="abc">  
 <h2>  
 <spring:message code="found"/>  
 <c:out value="${phoneQuantity}"/> <spring:message code="phones"/>  
 <div style="">  
 <a href="${pageContext.request.contextPath}/product-list?field=${not empty param.field ? param.field : null}&order=${not empty param.order ? param.order : null}&search=${not empty param.search ? param.search : null}&page=${empty param.page ? 1 : (param.page > 1 ? param.page - 1 : 1)}"><img  
 src="${pageContext.servletContext.contextPath}/images/previous.png" alt="Prev Page" height="30"  
 width="30"/></a>  
 <a href="${pageContext.request.contextPath}/product-list?field=${not empty param.field ? param.field : null}&order=${not empty param.order ? param.order : null}&search=${not empty param.search ? param.search : null}&page=${empty param.page ? (pages eq 1 ? 1 : 2) : (param.page < pages ? param.page + 1 : pages)}"><img  
 src="${pageContext.servletContext.contextPath}/images/next.png" alt="Next Page" height="30" width="30"/></a>  
 </div>  
 <div id="success-result">  
 </div>  
 <div id="error-result">  
 </div>  
 <div id="ajax-errors">  
 </div>  
 </h2>  
 </div>  
 <tags:phone-list/>  
 </body>  
</tags:master>

Файл quick-order.jsp

<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>  
<%@ taglib prefix="tags" tagdir="/WEB-INF/tags" %>  
<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>  
<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form" %>  
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn" %>  
<tags:master pageTitle="Quick Order">  
 <html>  
 <head>  
 <title><spring:message code="page.fastOrder"/></title>  
 </head>  
 <br>  
 <h2>  
 <spring:message code="page.fastOrder"/>  
 </h2>  
 <div id="success-result">  
 <c:if test="${not empty success}">  
 <spring:message code="cart.update.success"/>  
 </c:if>  
 <c:if test="${not empty successfulPhones}">  
 <c:forEach var="phone" items="${successfulPhones}">  
 <c:out value="${phone.model} added successfuly"/>  
 <br>  
 </c:forEach>  
 </c:if>  
 </div>  
 <div class="result-error">  
 <c:if test="${not empty errors}">  
 <spring:message code="cart.update.error"/>  
 </c:if>  
 </div>  
 <table>  
 <tr>  
 <td>  
 <spring:message code="model"/>  
 </td>  
 <td>  
 <spring:message code="quantity"/>  
 </td>  
 </tr>  
 <form:form id="add-form" action="${pageContext.servletContext.contextPath}/quick-order" method="post"  
 modelAttribute="quickOrderElementsDto">  
 <c:forEach begin="0" end="10" varStatus="status">  
 <c:set var="index" value="${status.index}"/>  
 <tr>  
 <td>  
 <form:input path="quickOrderElements[${index}].model"/>  
 <div class="result-error">  
 <form:errors path="quickOrderElements[${index}].model"/>  
 </div>  
 </td>  
 <td>  
 <form:input path="quickOrderElements[${index}].quantity"/>  
 <div class="result-error">  
 <form:errors path="quickOrderElements[${index}].quantity"/>  
 </div>  
 </td>  
 </tr>  
 </c:forEach>  
 </form:form>  
 </table>  
 <br>  
 <button form="add-form" class="btn btn-outline-primary" type="submit">  
 <spring:message code="addToCart"/>  
 </button>  
 </html>  
</tags:master>

Файл regitstration-page.jsp

<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<%@ taglib prefix="tags" tagdir="/WEB-INF/tags" %>  
<tags:master pageTitle="404">  
 <html>  
 <head>  
 <title><spring:message code="page.registration.register"/></title>  
 </head>  
 <body>  
 <br>  
 <div id="error-result">  
 <c:if test="${not empty error}">  
 <spring:message code="page.registration.error.while.registration"/>  
 </c:if>  
 <c:if test="${not empty errorEmpty}">  
 <spring:message code="page.registration.error.while.registration.empty"/>  
 </c:if>  
 </div>  
 <form action="${pageContext.servletContext.contextPath}/register" method="post" onchange="checkPassword(this)">  
 <div style="color:red" id="password-mismatch"></div>  
 <table>  
 <tr>  
 <td style="border:none"><spring:message code="page.login.username"/></td>  
 <td style="border:none">  
 <input type="text" name="username"/>  
 </td>  
 </tr>  
 <tr>  
 <td style="border:none"><spring:message code="page.login.password"/></td>  
 <td style="border:none">  
 <input name="password1" type="password"/>  
 <div style="color:red" id="password1-err"></div>  
 </td>  
 </tr>  
 <tr>  
 <td style="border:none"><spring:message code="page.registration.repeat.password"/></td>  
 <td style="border:none">  
 <input name="password2" type="password"/>  
 <div style="color:red" id="password2-err"></div>  
 </td>  
 </tr>  
 </table>  
 <br>  
 <div>  
 <button class="btn btn-outline-primary" type="submit"><spring:message  
 code="page.registration.register"/></button>  
 </div>  
 </form>  
 </body>  
 </html>  
</tags:master>

Файл update-delete.jsp

<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>  
<%@ taglib prefix="tags" tagdir="/WEB-INF/tags" %>  
<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>  
<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form" %>  
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn" %>  
<tags:masterLogged pageTitle="Update/Delete">  
 <br>  
 <div id="success-result">  
 </div>  
 <div id="error-result">  
 </div>  
 <br>  
 <form method="get" class="form-inline my-2 my-lg-0 mr-auto"  
 action="${pageContext.servletContext.contextPath}/admin/device-operations/update-delete">  
 <input name="search" class="form-control mr-sm-2" type="search" placeholder="Search" aria-label="search"  
 aria-describedby="basic-addon2"  
 value="${not empty param.search ? param.search : ''}">  
 <button class="btn btn-outline-primary my-2 my-sm-0"><spring:message code="search"/></button>  
 </form>  
 <br>  
 <table id="myTable">  
 <thead style="border-top: none">  
 <tr>  
 <td><spring:message code="phone.image"/></td>  
 <td>  
 <spring:message code="phone.brand"/>  
 </td>  
 <td>  
 <spring:message code="model"/>  
 </td>  
 <td><spring:message code="color"/></td>  
 <td>  
 <spring:message code="phone.display.size"/>  
 </td>  
 <td>  
 <spring:message code="price"/>  
 </td>  
 <td><spring:message code="action"/></td>  
 </tr>  
 </thead>  
 <c:forEach var="phone" items="${phones}">  
 <tr>  
 <td>  
 <div class="d-flex align-items-center">  
 <img src="https://raw.githubusercontent.com/andrewosipenko/phoneshop-ext-images/master/${phone.imageUrl}"  
 alt="logo" style="width: 85px; height: 85px" class="rounded-circle">  
 </div>  
 </td>  
 <td><c:out value="${phone.brand}"/></td>  
 <td>  
 <a href="${pageContext.servletContext.contextPath}/update/${phone.id}">  
 <c:out value="${phone.model}"/>  
 </a>  
 </td>  
 <td>  
 <c:forEach var="color" items="${phone.colors}">  
 <c:out value="${color.code}"/><br>  
 </c:forEach>  
 </td>  
 <td><c:out value="${phone.displaySizeInches}"/>"</td>  
 <td>$ <c:out value="${phone.price}"/>  
 <c:if test="${phone.discountPercent != 0}">  
 - <c:out value="${phone.discountPercent}"/>%  
 <br>  
  
 <div style="color:green">  
 $ <c:out value="${phone.actualPrice}"/>  
 </div>  
 </td>  
 </c:if>  
 <td>  
 <input id="phoneId${phone.id}" name="phoneId" type="hidden" value="${phone.id}"/>  
 <div class="result-error" id="result${phone.id}"></div>  
 <button class="btn btn-outline-danger" onclick="deleteDevice(${phone.id},this)">  
 <spring:message code="page.admin.delete.device"/>  
 </button>  
 <a class="btn btn-outline-primary" href="${pageContext.servletContext.contextPath}/admin/device-operations/update-delete/update/${phone.id}">  
 <spring:message code="page.admin.update.device"/>  
 </a>  
 </td>  
 </tr>  
 </c:forEach>  
 </table>  
 <div style="text-align:center; position: center;">  
 <a href="${pageContext.request.contextPath}/admin/device-operations/update-delete?search=${not empty param.search ? param.search : null}&page=${empty param.page ? 1 : (param.page > 1 ? param.page - 1 : 1)}"><img  
 src="${pageContext.servletContext.contextPath}/images/previous.png" alt="Prev Page" height="30" width="30"/></a>  
 <a href="${pageContext.request.contextPath}/admin/device-operations/update-delete?search=${not empty param.search ? param.search : null}&page=${empty param.page ? (pages eq 1 ? 1 : 2) : (param.page <= pages ? param.page + 1 : pages)}"><img  
 src="${pageContext.servletContext.contextPath}/images/next.png" alt="Next Page" height="30" width="30"/></a>  
 </div>  
</tags:masterLogged>

Файл user-management.jsp

<%@ page contentType="text/html;charset=UTF-8" language="java" %>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>  
<%@ taglib prefix="tags" tagdir="/WEB-INF/tags" %>  
<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>  
<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form" %>  
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn" %>  
<tags:masterLogged pageTitle="Update/Delete">  
 <br>  
 <table style="position: absolute; left: 40%; top: 10%">  
 <tr>  
 <td><spring:message code="page.login.username"/></td>  
 <td><spring:message code="page.admin.user.role"/></td>  
 <td><spring:message code="action"/></td>  
 </tr>  
 <c:forEach var="user" items="${users}">  
 <tr>  
 <td>  
 <c:out value="${user.userName}"/>  
 </td>  
 <td>  
 <c:out value="${user.role}"/>  
 </td>  
 <td>  
 <c:if test="${user.isAccountNonLocked eq 1}">  
 <div style="color:blue" id="success-blocking${user.userName}"></div>  
 <button id="button${user.userName}" class="btn btn-outline-danger"  
 onclick="updateStatus('${user.userName}',${user.isAccountNonLocked},this)">  
 <spring:message code="page.admin.block.user"/>  
 </button>  
 </c:if>  
 <c:if test="${user.isAccountNonLocked eq 0}">  
 <div style="color:blue" id="success-blocking${user.userName}"></div>  
 <button id="button${user.userName}" class="btn btn-outline-primary"  
 onclick="updateStatus('${user.userName}',${user.isAccountNonLocked},this)">  
 <spring:message code="page.admin.unblock.user"/>  
 </button>  
 </c:if>  
 </td>  
 </tr>  
 </c:forEach>  
 </table>  
 <div style="text-align:center; position: center;">  
 <a href="${pageContext.request.contextPath}/admin/user-management?page=${empty param.page ? 1 : (param.page > 1 ? param.page - 1 : 1)}"><img  
 src="${pageContext.servletContext.contextPath}/images/previous.png" alt="Prev Page" height="30" width="30"/></a>  
 <a href="${pageContext.request.contextPath}/admin/user-management?page=${empty param.page ? (pages eq 1 ? 1 : 2) : (param.page <= pages ? param.page + 1 : pages)}"><img  
 src="${pageContext.servletContext.contextPath}/images/next.png" alt="Next Page" height="30" width="30"/></a>  
 </div>  
</tags:masterLogged>

Файл error.tag

<%@ tag trimDirectiveWhitespaces="true" %>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>  
<%@ attribute name="name" required="true" %>  
<%@ attribute name="errors" required="true" type="org.springframework.validation.BindingResult" %>  
  
<div class="result-error">  
 <c:if test="${not empty errors.getFieldErrors(name)}">  
 <c:if test="${errors.getFieldErrors(name)[0].code eq 'emptyField'}">  
 <spring:message code="order.error.noValue"/>  
 </c:if>  
 <c:if test="${errors.getFieldErrors(name)[0].code ne 'emptyField'}">  
 <spring:message code="order.error.wrongInput"/>  
 </c:if>  
 </c:if>  
</div>

Файл master.tag

<%@ tag trimDirectiveWhitespaces="true" pageEncoding="UTF-8" %>  
<%@ attribute name="pageTitle" required="true" %>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>  
<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>  
<%@ taglib prefix="sec" uri="http://www.springframework.org/security/tags" %>  
  
<html>  
<head>  
 <link href='http://fonts.googleapis.com/css?family=Lobster+Two' rel='stylesheet' type='text/css'>  
 <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">  
 <link rel="stylesheet" href="${pageContext.servletContext.contextPath}/styles/main.css">  
 <script src="https://code.jquery.com/jquery-1.8.3.js"></script>  
 <c:if test="${pageContext.response.locale eq 'en'}">  
 <script type="text/javascript">  
 <%@include file="/js/main.js"%>  
 </script>  
 </c:if>  
 <c:if test="${pageContext.response.locale eq 'ru'}">  
 <script type="text/javascript" charset="UTF-8">  
 <%@include file="/js/main\_ru.js"%>  
 </script>  
 </c:if>  
 <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"  
 integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">  
 <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"  
 integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"  
 crossorigin="anonymous"></script>  
</head>  
<body>  
<nav id="navbar" class="navbar navbar-expand-lg navbar-light nav-borderb sticky-top"  
 style="background-color:deepskyblue">  
 <a href="${pageContext.servletContext.contextPath}/product-list" class="navbar-brand"><spring:message  
 code="page.phoneshop.logo"/>  
 </a>  
 <div class="collapse navbar-collapse">  
 <ul class="navbar-nav mr-auto">  
 <li class="nav-item">  
 <a href="${pageContext.servletContext.contextPath}/product-list" class="nav-link"><spring:message  
 code="page.homepage"/> </a>  
 </li>  
 <li class="nav-item">  
 <a href="${pageContext.servletContext.contextPath}/hot-prices" class="nav-link"><spring:message  
 code="page.hot.prices"/></a>  
 </li>  
 <li class="nav-item">  
 <a href="${pageContext.servletContext.contextPath}/contacts" class="nav-link"><spring:message  
 code="page.contacts"/></a>  
 </li>  
 <li class="nav-item">  
 <a href="${pageContext.servletContext.contextPath}/about-us" class="nav-link"><spring:message  
 code="page.about.us.page.name"/> </a>  
 </li>  
 <sec:authorize access="hasRole('ADMIN')">  
 <li class="nav-item">  
 <a class="nav-link" href="${pageContext.servletContext.contextPath}/admin/main"><spring:message code="page.admin.go.to.admin.console"/></a>  
 </li>  
 </sec:authorize>  
 </ul>  
 <form method="get" class="form-inline my-2 my-lg-0 mr-auto"  
 action="${pageContext.servletContext.contextPath}/product-list">  
 <input name="search" class="form-control mr-sm-2" type="search" placeholder="Search" aria-label="search"  
 aria-describedby="basic-addon2"  
 value="${not empty param.search ? param.search : ''}">  
 <button class="btn btn-outline-light my-2 my-sm-0"><spring:message code="search"/></button>  
 </form>  
 <div class="mr-lg-auto">  
 <a href="?lang=en"><img src="${pageContext.servletContext.contextPath}/images/english.svg" alt="English"  
 width="55" height="40"></a>  
 <a href="?lang=ru"><img src="${pageContext.servletContext.contextPath}/images/russian.svg" alt="English"  
 width="55" height="40"></a>  
 </div>  
 <span class="navbar-text">  
 <sec:authorize access="hasRole('ADMIN')">  
 <a href="${pageContext.servletContext.contextPath}/admin/orders">  
 <c:out value="${pageContext.request.userPrincipal.name}"/>  
 </a>,  
 <a href="${pageContext.servletContext.contextPath}/logout"><spring:message code="page.logout"/></a>  
 </sec:authorize>  
  
 <sec:authorize access="isAuthenticated() eq false">  
 <a href="${pageContext.servletContext.contextPath}/login"  
 class="btn btn-light btn-sm"><spring:message  
 code="page.login"/></a>  
 </sec:authorize>  
  
 <sec:authorize access="hasRole('CUSTOMER')">  
 <a style="font-weight: bold"><c:out value="${pageContext.request.userPrincipal.name}"/>,</a>  
 <a class="btn btn-light btn-sm" href="${pageContext.servletContext.contextPath}/logout"><spring:message  
 code="page.logout"/></a>  
 <a href="${pageContext.servletContext.contextPath}/cart">  
 <spring:message code="cart"/>  
 <div id="cart-quantity" class="l">  
 <c:out value="${cart.totalQuantity}"/>  
 </div>  
 <spring:message code="items"/>,  
 <div id="cart-totalCost" class="l">  
 <c:out value="${cart.totalCost}"/>  
 </div>  
 <spring:message code="currency"/>  
 </a>  
 </sec:authorize>  
 </span>  
 </div>  
</nav>  
<main>  
 <jsp:doBody/>  
</main>  
<br>  
<br>  
</body>  
</html>

Файл masterLogged.tag

<%@ tag trimDirectiveWhitespaces="true" %>  
<%@ attribute name="pageTitle" required="true" %>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>  
<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>  
<%@ taglib prefix="sec" uri="http://www.springframework.org/security/tags" %>  
<html>  
<head>  
 <link href='http://fonts.googleapis.com/css?family=Lobster+Two' rel='stylesheet' type='text/css'>  
 <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">  
 <link rel="stylesheet" href="${pageContext.servletContext.contextPath}/styles/main.css">  
 <script src="https://code.jquery.com/jquery-1.8.3.js"></script>  
 <c:if test="${pageContext.response.locale eq 'en'}">  
 <script type="text/javascript">  
 <%@include file="/js/main.js"%>  
 </script>  
 </c:if>  
 <c:if test="${pageContext.response.locale eq 'ru'}">  
 <script type="text/javascript" charset="UTF-8">  
 <%@include file="/js/main\_ru.js"%>  
 </script>  
 </c:if>  
 <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"  
 integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">  
 <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"  
 integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"  
 crossorigin="anonymous"></script>  
</head>  
<body>  
<nav class="navbar navbar-expand-lg navbar-light bg-light">  
 <a href="${pageContext.servletContext.contextPath}/admin/main" class="navbar-brand"><spring:message code="page.admin.console"/>  
 </a>  
 <div class="collapse navbar-collapse">  
 <ul class="navbar-nav mr-auto">  
 <li class="nav-item">  
 <a href="${pageContext.servletContext.contextPath}/admin/orders" class="nav-link"><spring:message code="page.admin.order.management"/></a>  
 </li>  
 <li class="nav-item">  
 <a href="${pageContext.servletContext.contextPath}/admin/device-operations/add" class="nav-link"><spring:message code="page.admin.add.device"/></a>  
 </li>  
 <li class="nav-item">  
 <a href="${pageContext.servletContext.contextPath}/admin/device-operations/update-delete" class="nav-link"><spring:message code="page.admin.add.phone"/></a>  
 </li>  
 <li class="nav-item">  
 <a href="${pageContext.servletContext.contextPath}/admin/user-management" class="nav-link"><spring:message code="page.admin.users"/></a>  
 </li>  
 <li class="nav-item">  
 <a href="${pageContext.servletContext.contextPath}/product-list" class="nav-link"><spring:message code="page.admin.back.to.user.part"/></a>  
 </li>  
 </ul>  
 <div class="mr-lg-auto">  
 <a href="?lang=en"><img src="${pageContext.servletContext.contextPath}/images/english.svg" alt="English"  
 width="55" height="40"></a>  
 <a href="?lang=ru"><img src="${pageContext.servletContext.contextPath}/images/russian.svg" alt="English"  
 width="55" height="40"></a>  
 </div>  
 <span class="navbar-text">  
 <sec:authorize access="hasRole('ADMIN')">  
 <a href="${pageContext.servletContext.contextPath}/admin/main">  
 <c:out value="${pageContext.request.userPrincipal.name}"/>  
 </a>,  
 <a href="${pageContext.servletContext.contextPath}/logout"><spring:message code="page.logout"/></a>  
 </sec:authorize>  
 <sec:authorize access="hasRole('ADMIN') eq false">  
 <a href="${pageContext.servletContext.contextPath}/login"  
 class="btn btn-outline-primary btn-sm"><spring:message  
 code="page.login"/></a>  
 </sec:authorize>  
 </span>  
 </div>  
</nav>  
<main>  
 <jsp:doBody/>  
</main>  
</body>  
</html>

Файл phone-list.tag

<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ taglib prefix="tags" tagdir="/WEB-INF/tags" %>  
<%@ taglib uri="http://www.springframework.org/tags/form" prefix="form" %>  
<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>  
  
<html>  
<table id="myTable">  
 <thead style="border-top: none">  
 <tr>  
 <td><spring:message code="phone.image"/></td>  
 <td>  
 <spring:message code="phone.brand"/>  
 <tags:sortLink field="brand" order="asc"/>  
 <tags:sortLink field="brand" order="desc"/>  
 </td>  
 <td>  
 <spring:message code="model"/>  
 <tags:sortLink field="model" order="asc"/>  
 <tags:sortLink field="model" order="desc"/>  
 </td>  
 <td><spring:message code="color"/></td>  
 <td>  
 <spring:message code="phone.display.size"/>  
 <tags:sortLink field="displaySizeInches" order="asc"/>  
 <tags:sortLink field="displaySizeInches" order="desc"/>  
 </td>  
 <td>  
 <spring:message code="price"/>  
 <tags:sortLink field="price" order="asc"/>  
 <tags:sortLink field="price" order="desc"/>  
 </td>  
 <td><spring:message code="quantity"/></td>  
 <td><spring:message code="action"/></td>  
 </tr>  
 </thead>  
 <c:forEach var="phone" items="${phones}">  
 <tr>  
 <td>  
 <div class="d-flex align-items-center">  
 <img src="${phone.imageUrl}" alt="logo" style="width: 100px; height: 95px">  
 </div>  
 </td>  
 <td><c:out value="${phone.brand}"/></td>  
 <td>  
 <a href="${pageContext.servletContext.contextPath}/product-details/${phone.id}">  
 <c:out value="${phone.model}"/>  
 </a>  
 </td>  
 <td>  
 <c:if test="${not empty phone.colors}">  
 <c:forEach var="color" items="${phone.colors}">  
 <span class="dot" style="background-color: ${color.code}"></span>  
 </c:forEach>  
 </c:if>  
 </td>  
 <td><c:out value="${phone.displaySizeInches}"/>"</td>  
 <td><spring:message code="currency"/> <c:out value="${phone.price}"/>  
 <c:if test="${phone.discountPercent != 0}">  
 - <c:out value="${phone.discountPercent}"/>%  
 <br>  
  
 <div style="color:green">  
 <spring:message code="currency"/> <c:out value="${phone.actualPrice}"/>  
 </div>  
 </td>  
 </c:if>  
 <td>  
 <input class="quantity-input" type="number" id="quantity${phone.id}" name="quantity" value="1"/>  
 <div class="result-error" id="result${phone.id}"></div>  
 <input id="phoneId${phone.id}" name="phoneId" type="hidden" value="${phone.id}"/>  
 </td>  
 <td>  
 <button onclick="addToCart(${phone.id})" class="btn btn-outline-primary">  
 <spring:message code="addToCart"/>  
 </button>  
 </td>  
 </tr>  
 </c:forEach>  
</table>  
</html>

Файл sort-link.tag

<%@ tag trimDirectiveWhitespaces="true" pageEncoding="UTF-8" %>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ attribute name="field" required="true" %>  
<%@ attribute name="order" required="true" %>  
  
<a href="?field=${field}&order=${order}&search=${param.search}"  
 style="${field eq param.field and order eq param.order  
? 'font-weight: bold' : ''}">  
 <c:if test="${order eq 'asc'}">  
 ${'🠗'}  
 </c:if>  
 <c:if test="${order eq 'desc'}">  
 ${'🠕'}  
 </c:if>  
</a>