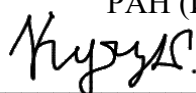


**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук
Образовательная программа бакалавриата «Программная инженерия»

СОГЛАСОВАНО

Руководитель проекта,
Приглашенный преподаватель
Базовой кафедры «Системное
программирование» Института системного
программирования им. В.П. Иванникова
РАН (ИСП РАН)



_____ А.А. Кучук
«16» мая 2021 г.

УТВЕРЖДАЮ

Академический руководитель
образовательной программы
«Программная инженерия»
профессор департамента программной
инженерии, канд. техн. наук

_____ В. В. Шилов
«__» _____ 2021 г.


**КЛИЕНТ-СЕРВЕРНОЕ ПРИЛОЖЕНИЕ ГЕОТАРГЕТИРОВАННОЙ РЕКЛАМЫ
«ГЕОФУД». СЕРВЕР**

Текст программы

ЛИСТ УТВЕРЖДЕНИЯ

RU.17701729.05.06-01 12 01-1-ЛУ

Исполнитель:

 студент группы БПИ199
_____ / Т. Р. Асатиани /
«16» мая 2021 г.

Подп. и дата	
Инв. № дубл.	
Взам. инв. №	
Подп. и дата	
Инв. № подл	RU.17701729.04.01-01 12 01-1-ЛУ

Москва 2021

УТВЕРЖДЕН
RU.17701729.04.01-01 12 01-1-ЛУ

**КЛИЕНТ-СЕРВЕРНОЕ ПРИЛОЖЕНИЕ ГЕОТАРГЕТИРОВАННОЙ РЕКЛАМЫ
«ГЕОФУД». СЕРВЕР**

Текст программы

RU.17701729.04.01-01 12 01-1

Листов 38

<i>Инв. № подл</i>	<i>Подп. и дата</i>
<i>Взам. инв. №</i>	<i>Инв. № дубл.</i>
<i>Подп. и дата</i>	<i>Подп. и дата</i>
RU.17701729.04.01-01 12 01-1-ЛУ	

Москва 2021

Содержание

1. Текст программы	4
1.1. AddShopRequest	4
1.2. AddStockRequest	4
1.3. AuthController.....	5
1.4. AuthResponse	6
1.5. AuthRequest	6
1.6. BusinessController	6
1.7. CustomUserDetails	12
1.8. CustomUserDetailsService	13
1.9. ControllerUtils	14
1.10. GeofoodApplication.....	15
1.11. GetStocksRequest	19
1.12. GetShopImgRequest.....	19
1.13. JwtProvider	20
1.14. JwtFilter	21
1.15. RegistrationRequest	22
1.16. RoleEntity.....	23
1.17. RoleRepository.....	23
1.18. SecurityConfig	23
1.19. ShopEntity	24
1.20. ShopRepository	26
1.21. ShopService	26
1.22. StockEntity	27
1.23. StockRepository	28
1.24. StockService	28
1.25. TestSecurityController.....	29
1.26. UserEntity.....	29
1.27. UserRepository.....	30
1.28. UserService.....	31
1.29. Views.....	32
1.30. VisitActionEntity	32
1.31. VisitActionRepository	33
1.32. VisitActionService	33

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-1У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

RU.17701729.04.01-01 12 01-1

1.33.	application.propeties	33
1.34.	data.sql.....	34
1.35.	pom.xml	34
2.	Список литературы.....	37
	Лист регистрации изменений.....	38

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-ЛЮ				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

1.1. AddShopRequest

```
package com.stabbers.geofood.controller.dto.business;
```

```
import lombok.Data;
```

```
import javax.validation.constraints.NotEmpty;
```

```
@Data
```

```
public class AddShopRequest {
```

```
    @NotEmpty
```

```
    private String name;
```

```
    @NotEmpty
```

```
    private double longitude;
```

```
    @NotEmpty
```

```
    private double latitude;
```

```
}
```

1.2. AddStockRequest

```
package com.stabbers.geofood.controller.dto.business;
```

```
import lombok.Data;
```

```
import javax.validation.constraints.NotEmpty;
```

```
@Data
```

```
public class AddStockRequest {
```

```
    @NotEmpty
```

```
    private String name;
```

```
    @NotEmpty
```

```
    private String promo;
```

```
    @NotEmpty
```

```
    private double oldPrice;
```

```
    @NotEmpty
```

```
    private double newPrice;
```

```
    @NotEmpty
```

```
    private String shopName;
```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-117				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

```
}
```

1.3. AuthController

```
package com.stabbers.geofood.controller;

import com.stabbers.geofood.config.jwt.JwtProvider;
import com.stabbers.geofood.controller.dto.auth.AuthRequest;
import com.stabbers.geofood.controller.dto.auth.AuthResponse;
import com.stabbers.geofood.controller.dto.register.RegistrationRequest;
import com.stabbers.geofood.entity.UserEntity;
import com.stabbers.geofood.service.ShopService;
import com.stabbers.geofood.service.UserService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.web.bind.annotation.*;

import javax.validation.Valid;

@RestController
public class AuthController {
    @Autowired
    private UserService userService;
    @Autowired
    private JwtProvider jwtProvider;

    @PostMapping("/reg/user")
    @ResponseStatus(value = HttpStatus.OK)
    public void registerUser(@RequestBody @Valid RegistrationRequest registrationRequest) {
        UserEntity user = new UserEntity();
        user.setPassword(registrationRequest.getPassword());
        user.setLogin(registrationRequest.getLogin());

        if(registrationRequest.getRole() != null && registrationRequest.getRole().equals("admin"))
            userService.saveAdmin(user);
        else
            userService.saveUser(user);
    }

    @PostMapping("/auth")
    public AuthResponse auth(@RequestBody AuthRequest request) {
        UserEntity userEntity = userService.findByLoginAndPassword(request.getLogin(), request.getPassword());
        String token = jwtProvider.generateToken(userEntity.getLogin());

        return new AuthResponse(token);
    }
}
```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-ЛП				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

1.4. AuthResponse

```
package com.stabbers.geofood.controller.dto.auth;
```

```
import lombok.AllArgsConstructor;
import lombok.Data;
```

```
@Data
@AllArgsConstructor
public class AuthResponse {
    private String token;
}
```

1.5. AuthRequest

```
package com.stabbers.geofood.controller.dto.auth;
```

```
import lombok.Data;
```

```
import javax.validation.constraints.NotEmpty;
```

```
@Data
public class AuthRequest {
    @NotEmpty
    private String login;

    @NotEmpty
    private String password;
}
```

1.6. BusinessController

```
package com.stabbers.geofood.controller;
```

```
import com.fasterxml.jackson.annotation.JsonView;
import com.stabbers.geofood.controller.dto.business.*;
import com.stabbers.geofood.config.jwt.JwtProvider;
import com.stabbers.geofood.entity.ShopEntity;
import com.stabbers.geofood.entity.StockEntity;
import com.stabbers.geofood.entity.UserEntity;
import com.stabbers.geofood.entity.VisitActionEntity;
import com.stabbers.geofood.entity.json.Views;
import com.stabbers.geofood.service.ShopService;
import com.stabbers.geofood.service.StockService;
import com.stabbers.geofood.service.UserService;
import com.stabbers.geofood.service.VisitActionService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;
```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-ЛД				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

```
import java.sql.Timestamp;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.*;
```

```
@RestController
```

```
public class BusinessController {
```

```
    @Autowired
```

```
    private UserService userService;
```

```
    @Autowired
```

```
    private ShopService shopService;
```

```
    @Autowired
```

```
    private StockService stockService;
```

```
    @Autowired
```

```
    private VisitActionService visitActionService;
```

```
    @Autowired
```

```
    private JwtProvider jwtProvider;
```

```
private UserEntity getUserByToken(String bearer ){
```

```
    String token = ControllerUtils.getTokenFromHeader(bearer);
```

```
    UserEntity user = null;
```

```
    if (token != null && jwtProvider.validateToken(token)) {
```

```
        String userLogin = jwtProvider.getLoginFromToken(token);
```

```
        user = userService.findByLogin(userLogin);
```

```
    }
```

```
    return user;
```

```
}
```

```
// ADD SHOP.
```

```
@PostMapping("/admin/shop/add")
```

```
public HttpStatus addShops(@RequestHeader("Authorization") String bearer, @RequestBody AddShopRequest request) {
```

```
    String token = ControllerUtils.getTokenFromHeader(bearer);
```

```
    UserEntity admin = null;
```

```
    if (token != null && jwtProvider.validateToken(token)) {
```

```
        String userLogin = jwtProvider.getLoginFromToken(token);
```

```
        admin = userService.findByLogin(userLogin);
```

```
    }
```

```
// TODO: Прокнуть SecurityException если токен не подходит
```

```
    if (admin == null)
```

```
        return HttpStatus.BAD_REQUEST;
```

```
    ShopEntity newShop = ControllerUtils.createShop(request);
```

```
    newShop.setHolder(admin);
```

```
    admin.addShop(newShop);
```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-1У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата


```

shopService.saveShop(newShop);
return HttpStatus.OK;

}

// GET SHOPS.
@GetMapping("/admin/shop/get")
public ResponseEntity<Iterable<ShopEntity>> getShops(@RequestHeader("Authorization") String bearer) {
    String token = ControllerUtils.getTokenFromHeader(bearer);

    UserEntity user = null;
    if (token != null && jwtProvider.validateToken(token)) {
        String userLogin = jwtProvider.getLoginFromToken(token);
        user = userService.findByLogin(userLogin);
    }

    if (user == null)
        return null;

    List<ShopEntity> shops = user.getShops();
    return new ResponseEntity<>(shops, HttpStatus.OK);
}

// ADD STOCK.
@PostMapping("/admin/stock/add")
public HttpStatus addStock(@RequestHeader("Authorization") String bearer, @RequestBody AddStockRequest request) {
    String token = ControllerUtils.getTokenFromHeader(bearer);

    UserEntity admin = null;
    if (token != null && jwtProvider.validateToken(token)) {
        String userLogin = jwtProvider.getLoginFromToken(token);
        admin = userService.findByLogin(userLogin);
    }

    // TODO: Прокнуть SecurityException если токен не подходит

    if (admin == null)
        return HttpStatus.BAD_REQUEST;

    List<ShopEntity> shops = admin.getShops();
    ShopEntity curShop = null;
    for (ShopEntity shop : shops) {
        if (shop.getName().equals(request.getShopName()))
            curShop = shop;
    }

    if (curShop == null)
        return HttpStatus.BAD_REQUEST;

    StockEntity newStock = ControllerUtils.createStock(request);

```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-Л1У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

```

newStock.setShop(curShop);
curShop.addStock(newStock);
stockService.saveStock(newStock);

return HttpStatus.OK;
}

// GET STOCKS.
@GetMapping("/admin/stock/get")
public ResponseEntity<Iterable<StockEntity>> getStocks(@RequestHeader("Authorization") String bearer) {
    String token = ControllerUtils.getTokenFromHeader(bearer);

    UserEntity admin = null;
    if (token != null && jwtProvider.validateToken(token)) {
        String userLogin = jwtProvider.getLoginFromToken(token);
        admin = userService.findByLogin(userLogin);
    }

    if (admin == null)
        return null;

    ArrayList<StockEntity> stocks = new ArrayList<>();
    for (ShopEntity shop : admin.getShops()) {
        stocks.addAll(shop.getStocks());
    }

    return new ResponseEntity<>(stocks, HttpStatus.OK);
}

// USER.
// GET NEAR SHOPS.
@JsonView(Views.forList.class)
@PostMapping("/user/shops")
public List<ShopEntity> getNearShops(@RequestHeader("Authorization") String bearer,
                                     @RequestBody GetNearShopsRequest request) {

    UserEntity user = null;
    user = getUserByToken(bearer);

    if (user == null)
        return null;

    ArrayList<ShopEntity> shops = (ArrayList<ShopEntity>) shopService.getAllShops();
    ArrayList<ShopEntity> validShops = new ArrayList<>();

    for (ShopEntity shop : shops) {
        if (ControllerUtils.shopInAarea(request.getLatitude(), request.getLongitude(), request.getRadius(), shop))
            validShops.add(shop);
    }

    return validShops;
}

```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-1У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

```

}

// GET ALL SHOP STOCKS
@JsonView(Views.forList.class)
@PostMapping("/user/stocks")
public List<StockEntity> getStocks(@RequestHeader("Authorization") String bearer,
                                   @RequestBody GetStocksRequest request) {

    String token = ControllerUtils.getTokenFromHeader(bearer);

    UserEntity user = null;
    if (token != null && jwtProvider.validateToken(token)) {
        String userLogin = jwtProvider.getLoginFromToken(token);
        user = userService.findByLogin(userLogin);
    }

    if (user == null)
        return null;

    ShopEntity shop = shopService.findById(request.getId());
    if(shop == null)
        return null;

    //ArrayList<StockEntity> validStocks = (ArrayList<StockEntity>) shop.getStocks();
    ArrayList<StockEntity> validStocks = new ArrayList<>(shop.getStocks());

    boolean isSpecialVisitor = false;
    Map<Integer, VisitActionEntity> visitActions = new HashMap<>();
    for(VisitActionEntity action: user.getVisitActions()){
        if(action.getUser().getId().equals(user.getId())){
            visitActions.put(action.getShop().getId(), action);
        }
    }

    if(visitActions.get(shop.getId()) != null && visitActions.get(shop.getId()).getCount() >= 10)
        isSpecialVisitor = true;

    if(!isSpecialVisitor)
        validStocks.removeIf(StockEntity::isSpecial);

    return validStocks;
}

// MOVEMENT CONTROLLER.
@JsonView(Views.forList.class)
@PostMapping("/movement")
public HttpStatus movement(@RequestHeader("Authorization") String bearer,
                           @RequestBody MovementRequest request) throws ParseException {

    UserEntity user;
    user = getUserByToken(bearer);

```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-Л/У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

```

if (user == null)
    return null;

ArrayList<ShopEntity> shops = (ArrayList<ShopEntity>) shopService.getAllShops();
ArrayList<ShopEntity> validShops = new ArrayList<>();

for (ShopEntity shop : shops) {
    if (ControllerUtils.shopInAarea(request.getLatitude(), request.getLongitude(), 100, shop))
        validShops.add(shop);
}

Map<Integer, VisitActionEntity> visitActions = new HashMap<>();
for(VisitActionEntity action: user.getVisitActions()){
    if(action.getUser().getId().equals(user.getId())){
        visitActions.put(action.getShop().getId(), action);
    }
}

for(ShopEntity shop : validShops) {
    Integer shopId = shop.getId();
    VisitActionEntity visit = visitActions.get(shopId);
    SimpleDateFormat formatter = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
    Date requestDate = formatter.parse(request.getDate());

    if(visit != null){
        Date lastDate = new Date(visit.getLastVisit().getTime());

        if(requestDate.getTime() - lastDate.getTime() >= 1000){
            visit.setCount(visit.getCount() + 1);
            visit.setLastVisit(new Timestamp(requestDate.getTime()));
            visitActionService.saveVisitAction(visit);
        }
    }
    else {
        VisitActionEntity newVisit = new VisitActionEntity();
        newVisit.setLastVisit(new Timestamp(requestDate.getTime()));
        newVisit.setShop(shopService.findById(shopId));
        newVisit.setCount(1);
        newVisit.setUser(user);

        user.addVisit(newVisit);
        visitActionService.saveVisitAction(newVisit);
    }
}

return HttpStatus.OK;
}

@JsonView(Views.fullMessage.class)

```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-1/У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

```

@PostMapping("/shop/img")
public byte[] getImg(@RequestBody GetShopImgRequest request) {

    ShopEntity shop = shopService.findById(request.getShopId());
    return shop.getImg();
}

@JsonView(Views.fullMessage.class)
@PostMapping("/stock/img")
public byte[] getImg(@RequestBody GetStockImgRequest request) {

    StockEntity stock = stockService.findById(request.getStockId());
    return stock.getImg();
}
}

```

1.7. CustomUserDetails

```

package com.stabbers.geofood.config;

import com.stabbers.geofood.entity.UserEntity;
import org.springframework.security.core.GrantedAuthority;
import org.springframework.security.core.authority.SimpleGrantedAuthority;
import org.springframework.security.core.userdetails.UserDetails;

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

public class CustomUserDetails implements UserDetails {

    private String login;
    private String password;
    private Collection<? extends GrantedAuthority> grantedAuthorities;

    public static CustomUserDetails fromUserEntityToCustomUserDetails(UserEntity userEntity) {
        CustomUserDetails c = new CustomUserDetails();
        c.login = userEntity.getLogin();
        c.password = userEntity.getPassword();
        c.grantedAuthorities = Collections.singletonList(new SimpleGrantedAuthority(userEntity.getRole().getName()));
        //c.grantedAuthorities = Collections.singletonList(new SimpleGrantedAuthority("ROLE_USER"));
        return c;
    }

    @Override
    public Collection<? extends GrantedAuthority> getAuthorities() {
        return grantedAuthorities;
    }

    @Override

```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-Л1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

```

public String getPassword() {
    return password;
}

@Override
public String getUsername() {
    return login;
}

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

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

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

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

```

1.8. CustomUserService

```
package com.stabbers.geofood.config;
```

```

import com.stabbers.geofood.entity.UserEntity;
import com.stabbers.geofood.service.UserService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.core.userdetails.UserDetailsService;
import org.springframework.security.core.userdetails.UsernameNotFoundException;
import org.springframework.stereotype.Component;

```

```

@Component
public class CustomUserService implements UserDetailsService {
    @Autowired
    private UserService userService;

```

```

@Override
public CustomUserDetails loadUserByUsername(String username) throws UsernameNotFoundException {
    UserEntity userEntity = userService.findByLogin(username);
    return CustomUserDetails.fromUserEntityToCustomUserDetails(userEntity);
}

```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-Л1У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

```

}
}

```

1.9. ControllerUtils

```

package com.stabbers.geofood.controller;

import com.stabbers.geofood.controller.dto.business.AddShopRequest;
import com.stabbers.geofood.controller.dto.business.AddStockRequest;
import com.stabbers.geofood.entity.ShopEntity;
import com.stabbers.geofood.entity.StockEntity;

import static org.springframework.util.StringUtils.hasText;

public class ControllerUtils {

    public static String getTokenFromHeader(String bearer) {
        if (hasText(bearer) && bearer.startsWith("Bearer ")) {
            return bearer.substring(7);
        }
        return null;
    }

    public static ShopEntity createShop(AddShopRequest request){
        ShopEntity newShop = new ShopEntity();
        newShop.setLatitude(request.getLatitude());
        newShop.setLongitude(request.getLongitude());
        newShop.setName(request.getName());
        return newShop;
    }

    public static StockEntity createStock(AddStockRequest request){
        StockEntity newStock = new StockEntity();
        newStock.setName(request.getName());
        newStock.setPromo(request.getPromo());
        newStock.setOldPrice(request.getOldPrice());
        newStock.setNewPrice(request.getNewPrice());
        return newStock;
    }

    public static boolean shopInAarea(double x, double y, double radius, ShopEntity shop){
        double stockX = shop.getLatitude();
        double stockY = shop.getLongitude();
        double deltaX = 0.00005;
        double deltaY = 0.00005;
        // double h = Math.sqrt((stockX - x) * (stockX - x) + (stockY - y) * (stockY - y));
        // if(h <= radius)
        //     return true;
        // else
        //     return false;
    }

```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-ЛЮ				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

```

    return Math.abs(stockX - x) < deltaX * radius && Math.abs(stockY - y) < deltaY * radius;
}
}

```

1.10. GeofoodApplication

```

package com.stabbers.geofood;

import com.stabbers.geofood.entity.ShopEntity;
import com.stabbers.geofood.entity.StockEntity;
import com.stabbers.geofood.entity.UserEntity;
import com.stabbers.geofood.entity.VisitActionEntity;
import com.stabbers.geofood.service.ShopService;
import com.stabbers.geofood.service.StockService;
import com.stabbers.geofood.service.UserService;
import com.stabbers.geofood.service.VisitActionService;
import lombok.AllArgsConstructor;
import org.apache.commons.io.IOUtils;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.boot.context.event.ApplicationReadyEvent;
import org.springframework.context.event.EventListener;
import java.io.*;

import java.sql.Timestamp;
import java.util.Arrays;
import java.util.Date;
import java.util.List;

@SpringBootApplication
public class GeofoodApplication {
    @Autowired
    private UserService userService;
    @Autowired
    private ShopService shopService;
    @Autowired
    private StockService stockService;
    @Autowired
    private VisitActionService visitActionService;

    public static void main(String[] args) {
        SpringApplication.run(GeofoodApplication.class, args);
    }
}

```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-Л1У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата


```
String path = "achek/uploads/";
```

```
private String generatePromo(){
    String ans = "";
    for(int i = 0; i < 5; ++i){
        String srcChar = "qwertyuiopasdfghjklzxcvbnm1234567890QWERTYUIOPASDFGHJKLZXCVBNM";
        ans += srcChar.toCharArray()[((int) (Math.random() * (srcChar.length())))];
    }
    return ans;
}
```

```
@AllArgsConstructor
```

```
private class Shop {
    double longitude;
    double latitude;
    String name;
    int type;
}
```

```
private final List<Shop> shops = Arrays.asList(new Shop(55.74776520767051, 37.612974838004156, "Cofix", 0),
    new Shop(55.751741766356055, 37.61115492031277, "Cofix", 0),
    new Shop(55.755537194261436, 37.61409890481368, "Cofix", 0),
    new Shop(55.758067274364045, 37.6186486990423, "Cofix", 0),
    new Shop(55.75945272487036, 37.6241619791078, "Cofix", 0),
    new Shop(55.757314291488534, 37.62823003041814, "Cofix", 0),
    new Shop(55.755567315228, 37.63251218969216, "McDonald's", 1),
    new Shop(55.75336842353714, 37.63331509455606, "McDonald's", 1),
    new Shop(55.74987403927311, 37.63144164987365, "McDonald's", 1),
    new Shop(55.74924140176145, 37.624750776007915, "McDonald's", 1),
    new Shop(55.74927152758989, 37.616721727369075, "McDonald's", 1)
);
```

```
@EventListener(ApplicationReadyEvent.class)
```

```
public void createUsersWithContent() throws IOException {
    UserEntity admin = new UserEntity();
    admin.setLogin("admin");
    admin.setPassword("impsstabb");
    userService.saveAdmin(admin);
```

```
UserEntity user = new UserEntity();
user.setLogin("egor");
user.setPassword("1234567890");
userService.saveUser(user);
```

```
generateShops(admin, user);
}
```

```
private void generateShops(UserEntity holder, UserEntity user) throws IOException {
    int cnt = 0;
    for(Shop point : shops){
```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-1У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

```

ShopEntity newShop = new ShopEntity();
newShop.setHolder(holder);
String lorem = "lorem ipsum";
newShop.setLocation(lorem);
newShop.setLatitude(point.longitude);
newShop.setLongitude(point.latitude);
newShop.setName(point.name + "_" + cnt);
newShop.setType(point.type);
if(newShop.getType() == 1){
    final File initialFile = new File(path + "Mac_Logo.png");
    final InputStream targetStream = new DataInputStream(new FileInputStream(initialFile));
    newShop.setImg(IOUtils.toByteArray(targetStream));
}
else {
    final File initialFile = new File(path + "Cofix_Logo.png");
    final InputStream targetStream = new DataInputStream(new FileInputStream(initialFile));
    newShop.setImg(IOUtils.toByteArray(targetStream));
}

holder.addShop(newShop);
shopService.saveShop(newShop);

// SPECIAL
if(cnt == 0){
    createSpecialStock(newShop);
    createVisit(user, newShop, 9);
}
else if(cnt < 6){
    createSpecialStock(newShop);
    createVisit(user, newShop, 8);
}
// SPECIAL

if(newShop.getType() == 0)
    generateStocks(newShop, false);
else
    generateStocks(newShop, true);
cnt++;
}
}

private void generateStocks(ShopEntity holder, boolean isMac) throws IOException{
    for(int i = 0; i < 3; ++i){
        StockEntity newStock = new StockEntity();
        newStock.setPromo(generatePromo());
        newStock.setShop(holder);
        newStock.setName(holder.getName() + "_stock_" + i);
        newStock.setNewPrice(100);
        newStock.setOldPrice(200);
    }
}

```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-Л1У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

RU.17701729.04.01-01 12 01-1

```

if(isMac){
    final File initialFile = new File(path + "Mac_Stock.png");
    final InputStream targetStream = new DataInputStream(new FileInputStream(initialFile));
    newStock.setImg(IOUtils.toByteArray(targetStream));
}
else {
    final File initialFile = new File(path + "Cofix_Stock.png");
    final InputStream targetStream = new DataInputStream(new FileInputStream(initialFile));
    newStock.setImg(IOUtils.toByteArray(targetStream));
}

stockService.saveStock(newStock);

holder.addStock(newStock);
}
}

private void createSpecialStock(ShopEntity holder) throws IOException{
    StockEntity newStock = new StockEntity();
    newStock.setPromo(generatePromo());
    newStock.setShop(holder);
    newStock.setName(holder.getName() + "_SPECIAL");
    newStock.setNewPrice(50);
    newStock.setOldPrice(200);
    newStock.setSpecial(true);

    final File initialFile = new File(path + "Special.png");
    final InputStream targetStream = new DataInputStream(new FileInputStream(initialFile));
    newStock.setImg(IOUtils.toByteArray(targetStream));

    stockService.saveStock(newStock);

    holder.addStock(newStock);
}

public void createVisit(UserEntity user, ShopEntity shop, int cnt){
    VisitActionEntity newVisit = new VisitActionEntity();
    newVisit.setUser(user);
    newVisit.setCount(cnt);
    newVisit.setShop(shop);
    user.addVisit(newVisit);
    Date date = new Date(System.currentTimeMillis());
    newVisit.setLastVisit(new Timestamp(date.getTime()));
    visitActionService.saveVisitAction(newVisit);
}
}
}

```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-Л/У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

1.11.GetStocksRequest

```
package com.stabbers.geofood.controller.dto.business;

import lombok.Data;

import javax.validation.constraints.NotEmpty;

@Data
public class GetStocksRequest {
    @NotEmpty
    private int id;
}
```

1.11.1. GetStockImgRequest

```
package com.stabbers.geofood.controller.dto.business;

import lombok.Data;

import javax.validation.constraints.NotEmpty;

@Data
public class GetStockImgRequest {

    @NotEmpty
    int stockId;
}
```

1.12.GetShopImgRequest

```
package com.stabbers.geofood.controller.dto.business;

import lombok.Data;

import javax.validation.constraints.NotEmpty;

@Data
public class GetShopImgRequest {

    @NotEmpty
    int shopId;
}
```

1.12.1. GetNearShopsRequest

```
package com.stabbers.geofood.controller.dto.business;

import lombok.Data;
```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-1/У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

```
import javax.validation.constraints.NotEmpty;
```

```
@Data
public class GetNearShopsRequest {
```

```
    @NotEmpty
    private double longitude;
```

```
    @NotEmpty
    private double latitude;
```

```
    @NotEmpty
    private double radius;
```

```
}
```

1.13.JwtProvider

```
package com.stabbers.geofood.config.jwt;
```

```
import io.jsonwebtoken.Claims;
import io.jsonwebtoken.Jwts;
import io.jsonwebtoken.SignatureAlgorithm;
import lombok.extern.java.Log;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.stereotype.Component;
```

```
import java.time.LocalDate;
import java.time.ZonedDateTime;
import java.util.Date;
```

```
@Component
```

```
@Log
```

```
public class JwtProvider {
```

```
    @Value("${jwt.secret}")
    private String jwtSecret;
```

```
    public String generateToken(String login) {
        Date date = Date.from(LocalDate.now().plusDays(15).atStartOfDay(ZonedDateTime.systemDefault()).toInstant());
        return Jwts.builder()
            .setSubject(login)
            .setExpiration(date)
            .signWith(SignatureAlgorithm.HS512, jwtSecret)
            .compact();
    }
```

```
    public boolean validateToken(String token) {
        try {
            Jwts.parser().setSigningKey(jwtSecret).parseClaimsJws(token);
```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-1У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

```

        return true;
    } catch (Exception e) {
        log.severe("invalid token");
    }
    return false;
}

public String getLoginFromToken(String token) {
    Claims claims = Jwts.parser().setSigningKey(jwtSecret).parseClaimsJws(token).getBody();
    return claims.getSubject();
}
}

```

1.14.JwtFilter

```
package com.stabbers.geofood.config.jwt;
```

```

import com.stabbers.geofood.config.CustomUserDetails;
import com.stabbers.geofood.config.CustomUserDetailsService;
import lombok.extern.java.Log;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.authentication.UsernamePasswordAuthenticationToken;
import org.springframework.security.core.context.SecurityContextHolder;
import org.springframework.stereotype.Component;
import org.springframework.web.filter.GenericFilterBean;

```

```

import javax.servlet.FilterChain;
import javax.servlet.ServletException;
import javax.servlet.ServletRequest;
import javax.servlet.ServletResponse;
import javax.servlet.http.HttpServletRequest;
import java.io.IOException;

```

```
import static org.springframework.util.StringUtils.hasText;
```

```
@Component
```

```
@Log
```

```
public class JwtFilter extends GenericFilterBean {
```

```
    public static final String AUTHORIZATION = "Authorization";
```

```
@Autowired
```

```
    private JwtProvider jwtProvider;
```

```
@Autowired
```

```
    private CustomUserDetailsService customUserDetailsService;
```

```
@Override
```

```
    public void doFilter(ServletRequest servletRequest, ServletResponse servletResponse, FilterChain filterChain) throws
        IOException, ServletException {
```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-ЛП				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

```

logger.info("do filter...");
String token = getTokenFromRequest((HttpServletRequest) servletRequest);

if (token != null && jwtProvider.validateToken(token)) {
    String userLogin = jwtProvider.getLoginFromToken(token);
    CustomUserDetails customUserDetails = customUserDetailsService.loadUserByUsername(userLogin);
    UsernamePasswordAuthenticationToken auth = new UsernamePasswordAuthenticationToken(customUserDetails,
null, customUserDetails.getAuthorities());
    SecurityContextHolder.getContext().setAuthentication(auth);
}

filterChain.doFilter(servletRequest, servletResponse);
}

private String getTokenFromRequest(HttpServletRequest request) {
    String bearer = request.getHeader(AUTHORIZATION);
    if (hasText(bearer) && bearer.startsWith("Bearer ")) {
        return bearer.substring(7);
    }
    return null;
}
}

```

1.14.1. MovementRequest

```

package com.stabbers.geofood.controller.dto.business;

import lombok.Data;

import javax.validation.constraints.NotEmpty;
import java.util.Date;

@Data
public class MovementRequest {
    @NotEmpty
    private double longitude;

    @NotEmpty
    private double latitude;

    @NotEmpty
    private String date;
}

```

1.15.RegistrationRequest

```

package com.stabbers.geofood.controller.dto.register;

import lombok.Data;

```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-Л/У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

```
import javax.validation.constraints.NotEmpty;
```

```
@Data
public class RegistrationRequest {
```

```
    @NotEmpty
    private String login;
```

```
    @NotEmpty
    private String password;
```

```
    private String role;
```

```
}
```

1.16.RoleEntity

```
package com.stabbers.geofood.entity;
```

```
import lombok.Data;
```

```
import javax.persistence.*;
```

```
@Data
@Entity
@Table(name = "role")
public class RoleEntity {
```

```
    @Id
    @GeneratedValue(strategy=GenerationType.IDENTITY)
    @Column(name = "id", updatable = false, nullable = false)
    private Integer id;
```

```
    @Column
    private String name;
```

```
}
```

1.17.RoleRepository

```
package com.stabbers.geofood.repository;
```

```
import com.stabbers.geofood.entity.RoleEntity;
```

```
import org.springframework.data.jpa.repository.JpaRepository;
```

```
public interface RoleRepository extends JpaRepository<RoleEntity, Integer> {
```

```
    RoleEntity findByName(String name);
```

```
}
```

1.18.SecurityConfig

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-Л1У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата


```
package com.stabbers.geofood.config;
```

```
import com.stabbers.geofood.config.jwt.JwtFilter;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
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 org.springframework.security.config.http.SessionCreationPolicy;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.security.web.authentication.UsernamePasswordAuthenticationFilter;
```

```
@Configuration
```

```
@EnableWebSecurity
```

```
public class SecurityConfig extends WebSecurityConfigurerAdapter {
```

```
    @Autowired
```

```
    private JwtFilter jwtFilter;
```

```
    @Override
```

```
    protected void configure(HttpSecurity http) throws Exception {
```

```
        http
```

```
            .httpBasic().disable()
```

```
            .csrf().disable()
```

```
            .sessionManagement().sessionCreationPolicy(SessionCreationPolicy.STATELESS)
```

```
            .and()
```

```
            .authorizeRequests()
```

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

```
            .antMatchers("/user/*").hasRole("USER")
```

```
            .antMatchers("/reg/user", "/auth").permitAll()
```

```
            .and()
```

```
            .addFilterBefore(jwtFilter, UsernamePasswordAuthenticationFilter.class);
```

```
    }
```

```
    @Bean
```

```
    public PasswordEncoder passwordEncoder() {
```

```
        return new BCryptPasswordEncoder();
```

```
    }
```

```
}
```

1.19.ShopEntity

```
package com.stabbers.geofood.entity;
```

```
import com.fasterxml.jackson.annotation.JsonBackReference;
import com.fasterxml.jackson.annotation.JsonIgnore;
import com.fasterxml.jackson.annotation.JsonManagedReference;
import com.fasterxml.jackson.annotation.JsonView;
```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-1/1				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

```
import lombok.*;
import com.stabbers.geofood.entity.json.Views;
import org.hibernate.validator.constraints.UniqueElements;
```

```
import javax.persistence.*;
import java.util.ArrayList;
import java.util.List;
```

```
@Data
@Entity
@Table(name = "shop")
public class ShopEntity {
    @JsonView({Views.forList.class})
    @Id
    @GeneratedValue(strategy=GenerationType.IDENTITY)
    @Column(name = "id", updatable = false, nullable = false)
    private Integer id;

    @JsonView({Views.forList.class})
    @Column
    private String name;

    @JsonView({Views.forList.class})
    @Column
    private double longitude;

    @JsonView({Views.forList.class})
    @Column
    private double latitude;

    @JsonView({Views.forList.class})
    @Column
    private String location;

    @JsonView({Views.forList.class})
    @Column
    private int type;

    @JsonView({Views.forList.class})
    @JsonBackReference
    @ManyToOne
    @JoinColumn (name="user_id")
    private UserEntity holder;

    @JsonView({Views.fullMessage.class})
    @Lob
    @Column(columnDefinition = "BLOB")
    private byte[] img;
```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-Л1У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

```

@JsonView({Views.fullMessage.class})
@JsonIgnore
@JsonManagedReference
@OneToMany(targetEntity = StockEntity.class, mappedBy = "shop", cascade = CascadeType.ALL, orphanRemoval = true)
private List<StockEntity> stocks = new ArrayList<>();

public void addStock(StockEntity newStock){
    stocks.add(newStock);
}
}

```

1.20.ShopRepository

```

package com.stabbers.geofood.repository;

import com.stabbers.geofood.entity.ShopEntity;
import org.springframework.data.jpa.repository.JpaRepository;

public interface ShopRepository extends JpaRepository<ShopEntity, Integer> {
    ShopEntity findByName(String name);
}

```

1.21.ShopService

```

package com.stabbers.geofood.service;

import com.stabbers.geofood.entity.ShopEntity;
import com.stabbers.geofood.entity.StockEntity;
import com.stabbers.geofood.repository.ShopRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
public class ShopService {

    @Autowired
    private ShopRepository shopRepository;

    public ShopEntity saveShop(ShopEntity shop) {
        final String lorem = "Lorem ipsum";
        shop.setLocation(lorem);
        return shopRepository.save(shop);
    }

    public List<ShopEntity> getAllShops() {
        return shopRepository.findAll();
    }
}

```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-1/У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

```

public ShopEntity findById(int id){
    return shopRepository.findById(id).orElse(new ShopEntity());
}

}

```

1.22.StockEntity

```

package com.stabbers.geofood.entity;

import com.fasterxml.jackson.annotation.JsonBackReference;
import com.fasterxml.jackson.annotation.JsonView;
import com.stabbers.geofood.entity.json.Views;
import lombok.AllArgsConstructor;
import lombok.Builder;
import lombok.Data;
import lombok.NoArgsConstructor;
import org.hibernate.validator.constraints.UniqueElements;

import javax.persistence.*;

@Data
@Entity
@Table(name = "stock")
public class StockEntity {
    @JsonView({Views.forList.class})
    @Id
    @GeneratedValue(strategy=GenerationType.IDENTITY)
    @Column(name = "id", updatable = false, nullable = false)
    private Integer id;

    @JsonView({Views.forList.class})
    @Column(columnDefinition = "VARCHAR(128)")
    private String name;

    @JsonView({Views.forList.class})
    @Column(columnDefinition = "VARCHAR(128)")
    private String promo;

    @JsonView({Views.forList.class})
    @Column
    private double oldPrice;

    @JsonView({Views.forList.class})
    @Column
    private double newPrice;

    @JsonView({Views.forList.class})
    @Column

```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-ЛД				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

```

private boolean special;

@JsonView({Views.fullMessage.class})
@Lob
@Column(columnDefinition = "BLOB")
private byte[] img;

@JsonView({Views.forList.class})
@JsonBackReference
@ManyToOne
@JoinColumn (name="shop_id")
private ShopEntity shop;
}

```

1.23.StockRepository

```

package com.stabbers.geofood.repository;

import com.stabbers.geofood.entity.StockEntity;
import com.stabbers.geofood.entity.UserEntity;
import org.springframework.data.jpa.repository.JpaRepository;

public interface StockRepository extends JpaRepository<StockEntity, Integer> {
    StockEntity findByName(String name);
}

```

1.24.StockService

```

package com.stabbers.geofood.service;

import com.stabbers.geofood.entity.ShopEntity;
import com.stabbers.geofood.entity.StockEntity;
import com.stabbers.geofood.repository.ShopRepository;
import com.stabbers.geofood.repository.StockRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
public class StockService {

    @Autowired
    private StockRepository stockRepository;

    public StockEntity saveStock(StockEntity stock) {
        return stockRepository.save(stock);
    }
}

```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-Л1У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

```

public StockEntity findById(int id){
    return stockRepository.findById(id).orElse(new StockEntity());
}
}

```

1.25. TestSecurityController

```

package com.stabbers.geofood.controller;

import com.stabbers.geofood.config.jwt.JwtProvider;
import com.stabbers.geofood.entity.UserEntity;
import com.stabbers.geofood.service.UserService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestHeader;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class TestSecurityController {

    @Autowired
    private UserService userService;

    @Autowired
    private JwtProvider jwtProvider;

    @GetMapping("/admin/get")
    public ResponseEntity<UserEntity> getAdmin(@RequestHeader("Authorization") String bearer) {
        UserEntity admin =
        userService.findByLogin(jwtProvider.getLoginFromToken(ControllerUtils.getTokenFromHeader(bearer)));
        return new ResponseEntity<>(admin, HttpStatus.OK);
    }

    @GetMapping("/user/get")
    public ResponseEntity<UserEntity> getUser(@RequestHeader("Authorization") String bearer) {
        UserEntity user =
        userService.findByLogin(jwtProvider.getLoginFromToken(ControllerUtils.getTokenFromHeader(bearer)));
        return new ResponseEntity<>(user, HttpStatus.OK);
    }
}

```

1.26. UserEntity

```

package com.stabbers.geofood.entity;

import com.fasterxml.jackson.annotation.JsonManagedReference;
import lombok.*;
import org.hibernate.validator.constraints.UniqueElements;

```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-Л1У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

```

import javax.persistence.*;
import java.util.ArrayList;
import java.util.List;
import java.util.Set;

@Data
@Entity
@Table(name = "user")
public class UserEntity {
    @Id
    @GeneratedValue(strategy=GenerationType.IDENTITY)
    @Column(name = "id", updatable = false, nullable = false)
    private Integer id;

    @Column(unique = true, columnDefinition = "VARCHAR(128)")
    private String login;

    @Column(columnDefinition = "VARCHAR(128)")
    private String password;

    @ManyToOne
    @JoinColumn(name = "role_id")
    private RoleEntity role;

    @JsonManagedReference
    @OneToMany(targetEntity = VisitActionEntity.class, mappedBy = "user", cascade = CascadeType.ALL, orphanRemoval = true)
    private List<VisitActionEntity> visitActions = new ArrayList<>();

    public void addVisit(VisitActionEntity newVisit){
        visitActions.add(newVisit);
    }

    @JsonManagedReference
    @OneToMany(targetEntity = ShopEntity.class, mappedBy = "holder", cascade = CascadeType.ALL, orphanRemoval = true)
    private List<ShopEntity> shops = new ArrayList<>();

    public void addShop(ShopEntity newShop){
        shops.add(newShop);
    }
}

```

1.27. UserRepository

```

package com.stabbers.geofood.repository;

import com.stabbers.geofood.entity.ShopEntity;
import com.stabbers.geofood.entity.UserEntity;
import org.springframework.data.jpa.repository.JpaRepository;

```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-Л1У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

```
public interface UserRepository extends JpaRepository<UserEntity, Integer> {
    UserEntity findByLogin(String login);
}
```

1.28.UserService

```
package com.stabbers.geofood.service;
```

```
import com.stabbers.geofood.entity.RoleEntity;
import com.stabbers.geofood.entity.ShopEntity;
import com.stabbers.geofood.entity.UserEntity;
import com.stabbers.geofood.repository.RoleRepository;
import com.stabbers.geofood.repository.UserRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.stereotype.Service;
```

```
import java.util.List;
```

```
@Service
```

```
public class UserService {
```

```
    @Autowired
```

```
    private UserRepository userEntityRepository;
```

```
    @Autowired
```

```
    private RoleRepository roleRepository;
```

```
    @Autowired
```

```
    private PasswordEncoder passwordEncoder;
```

```
    public UserEntity saveUser(UserEntity user) {
```

```
        RoleEntity userRole = roleRepository.findByName("ROLE_USER");
```

```
        user.setRole(userRole);
```

```
        user.setPassword(passwordEncoder.encode(user.getPassword()));
```

```
        return userEntityRepository.save(user);
```

```
    }
```

```
    public UserEntity saveAdmin(UserEntity user) {
```

```
        RoleEntity userRole = roleRepository.findByName("ROLE_ADMIN");
```

```
        user.setRole(userRole);
```

```
        user.setPassword(passwordEncoder.encode(user.getPassword()));
```

```
        return userEntityRepository.save(user);
```

```
    }
```

```
    public UserEntity findByLogin(String login) {
```

```
        return userEntityRepository.findByLogin(login);
```

```
    }
```

```
    public UserEntity findByLoginAndPassword(String login, String password) {
```

```
        UserEntity user = findByLogin(login);
```

```
        if (user != null) {
```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-1У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата


```

    if (passwordEncoder.matches(password, user.getPassword())) {
        return user;
    }
}
return null;
}
}

```

1.29.Views

```

package com.stabbers.geofood.entity.json;

public class Views {
    public interface forList { }
    public interface fullMessage extends forList { }
}

```

1.30.VisitActionEntity

```

package com.stabbers.geofood.entity;

import com.fasterxml.jackson.annotation.JsonBackReference;
import com.fasterxml.jackson.annotation.JsonView;
import com.stabbers.geofood.entity.json.Views;
import lombok.Data;

import javax.persistence.*;

@Data
@Entity
@Table(name = "visit")
public class VisitActionEntity {
    @Id
    @GeneratedValue(strategy= GenerationType.IDENTITY)
    @Column(name = "id", updatable = false, nullable = false)
    private Integer id;

    @JsonView({Views.forList.class})
    @JsonBackReference
    @ManyToOne
    @JoinColumn (name="user_id")
    private UserEntity user;

    @JsonView({Views.forList.class})
    @JsonBackReference
    @ManyToOne
    @JoinColumn (name="shop_id")
    private ShopEntity shop;

    @JsonView({Views.forList.class})

```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-1/У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

```

@Column
private Integer count;

@JsonView({Views.forList.class})
@Column
private java.sql.Timestamp lastVisit;
}

```

1.31. VisitActionRepository

```

package com.stabbers.geofood.repository;

import com.stabbers.geofood.entity.VisitActionEntity;
import org.springframework.data.jpa.repository.JpaRepository;

public interface VisitActionRepository extends JpaRepository<VisitActionEntity, Integer> {
    VisitActionEntity findById(int id);
}

```

1.32. VisitActionService

```

package com.stabbers.geofood.service;

import com.stabbers.geofood.entity.VisitActionEntity;
import com.stabbers.geofood.repository.VisitActionRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service
public class VisitActionService {

    @Autowired
    private VisitActionRepository visitActionRepository;

    public VisitActionEntity saveVisitAction(VisitActionEntity newVisitAction) {
        return visitActionRepository.save(newVisitAction);
    }
}

```

1.33. application.properties

```

## Spring DATASOURCE (DataSourceAutoConfiguration & DataSourceProperties)
spring.datasource.url = jdbc:mysql://localhost:3306/geofood?serverTimezone=UTC&useLegacyDatetimeCode=false
spring.datasource.username = root
spring.datasource.password = root
spring.jpa.properties.hibernate.jdbc.time_zone=UTC

## Hibernate Properties
# The SQL dialect makes Hibernate generate better SQL for the chosen database
spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL5InnoDBDialect

```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-Л1У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

```
# spring.jpa.open-in-view = false
```

```
#db initialization and data loading
```

```
# Hibernate ddl auto (create, create-drop, validate, update)
```

```
spring.jpa.hibernate.ddl-auto = create-drop
```

```
spring.datasource.initialization-mode=always
```

```
jwt.secret = stabbers
```

```
server.port = 8080
```

```
upload.path = /Users/egor/Desktop/uploads
```

1.34.data.sql

```
INSERT INTO role(name) VALUES ('ROLE_ADMIN');
```

```
INSERT INTO role(name) VALUES ('ROLE_USER');
```

1.35.pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
```

```
  <modelVersion>4.0.0</modelVersion>
```

```
  <parent>
```

```
    <groupId>org.springframework.boot</groupId>
```

```
    <artifactId>spring-boot-starter-parent</artifactId>
```

```
    <version>2.4.2</version>
```

```
    <relativePath/> <!-- lookup parent from repository -->
```

```
  </parent>
```

```
  <groupId>com.stabbers</groupId>
```

```
  <artifactId>geofood</artifactId>
```

```
  <version>0.0.1-SNAPSHOT</version>
```

```
  <name>geofood</name>
```

```
  <description>geofood</description>
```

```
  <properties>
```

```
    <java.version>11</java.version>
```

```
  </properties>
```

```
  <dependencies>
```

```
    <dependency>
```

```
      <groupId>org.springframework.boot</groupId>
```

```
      <artifactId>spring-boot-starter-data-jpa</artifactId>
```

```
    </dependency>
```

```
    <dependency>
```

```
      <groupId>org.springframework.boot</groupId>
```

```
      <artifactId>spring-boot-starter-security</artifactId>
```

```
    </dependency>
```

```
    <dependency>
```

```
      <groupId>io.jsonwebtoken</groupId>
```

```
      <artifactId>jjwt</artifactId>
```

```
      <version>0.9.1</version>
```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-Л1У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

```

</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-validation</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-web</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-devtools</artifactId>
  <scope>runtime</scope>
  <optional>true</optional>
</dependency>
<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
  <scope>runtime</scope>
</dependency>
<dependency>
  <groupId>org.projectlombok</groupId>
  <artifactId>lombok</artifactId>
  <optional>true</optional>
</dependency>
<dependency>
  <groupId>org.springframework.security</groupId>
  <artifactId>spring-security-test</artifactId>
  <scope>test</scope>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-test</artifactId>
  <scope>test</scope>
</dependency>
<dependency>
  <groupId>commons-io</groupId>
  <artifactId>commons-io</artifactId>
  <version>2.8.0</version>
</dependency>
</dependencies>

```

```

<build>
  <plugins>
    <plugin>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-maven-plugin</artifactId>
      <configuration>
        <excludes>
          <exclude>

```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-Л1У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

RU.17701729.04.01-01 12 01-1

```

    <groupId>org.projectlombok</groupId>
    <artifactId>lombok</artifactId>
    </exclude>
  </excludes>
</configuration>
</plugin>
<plugin>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-maven-plugin</artifactId>
  <version>2.4.0</version>
</plugin>
</plugins>
</build>

</project>

```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-Л1У				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

2. СПИСОК ЛИТЕРАТУРЫ

1. ГОСТ 19.401-78. ЕСПД. Текст программы. Требования к содержанию и оформлению. – М.: ИПК Издательство стандартов, 2001.
2. Документация по Spring Framework [Электронный ресурс]// URL: <https://docs.spring.io/spring-framework/docs/current/reference/html/> (Дата обращения: 5.05.2021, режим доступа свободный).

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-ЛЮ				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

[illegible]

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.01-01 12 01-1-ЛП				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата