



Технически университет - София

Разработване на уеб приложение за резервация на спортни съоръжения

1

Дипломна работа

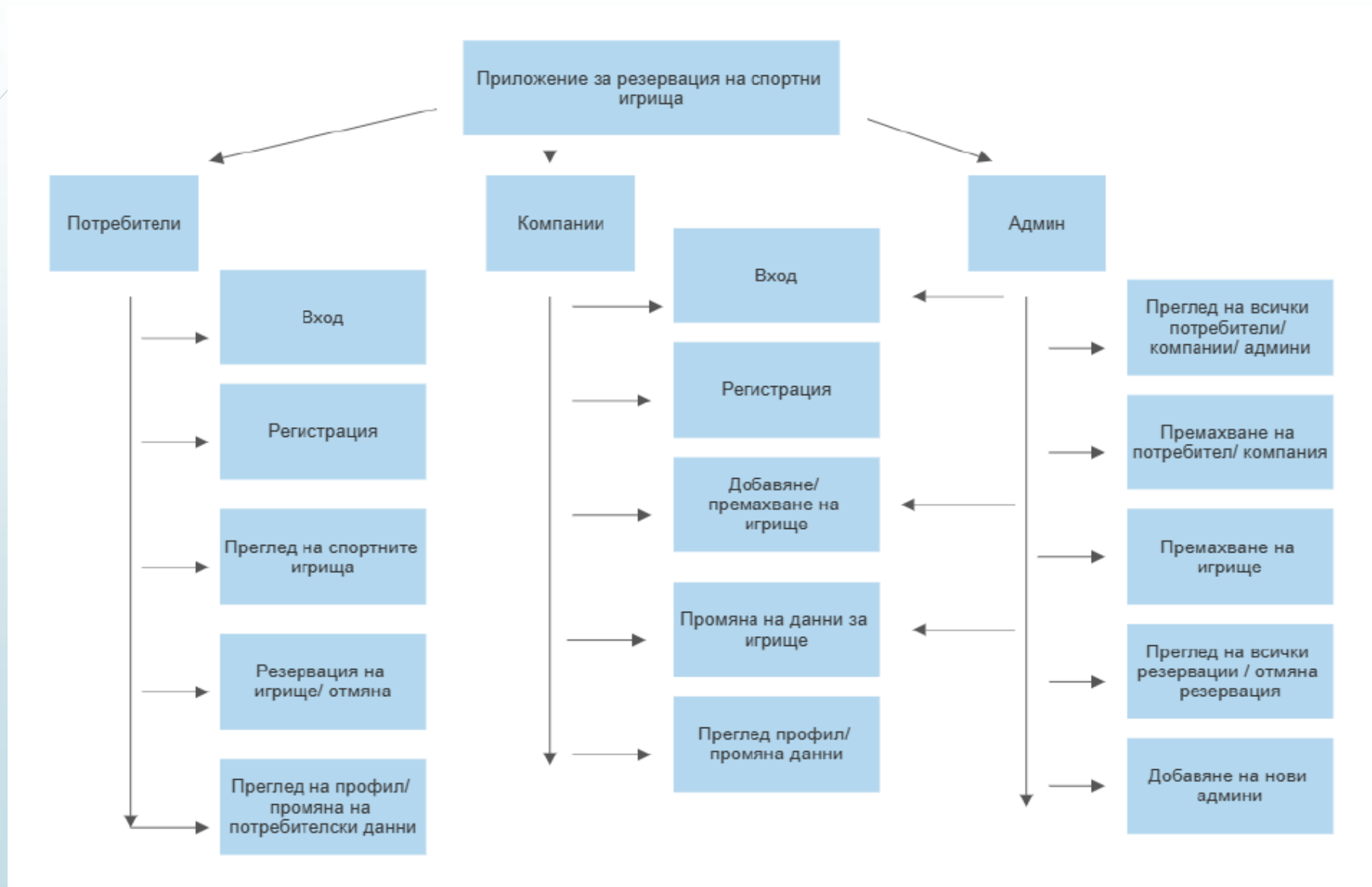
На

Изабел Христова Филипова

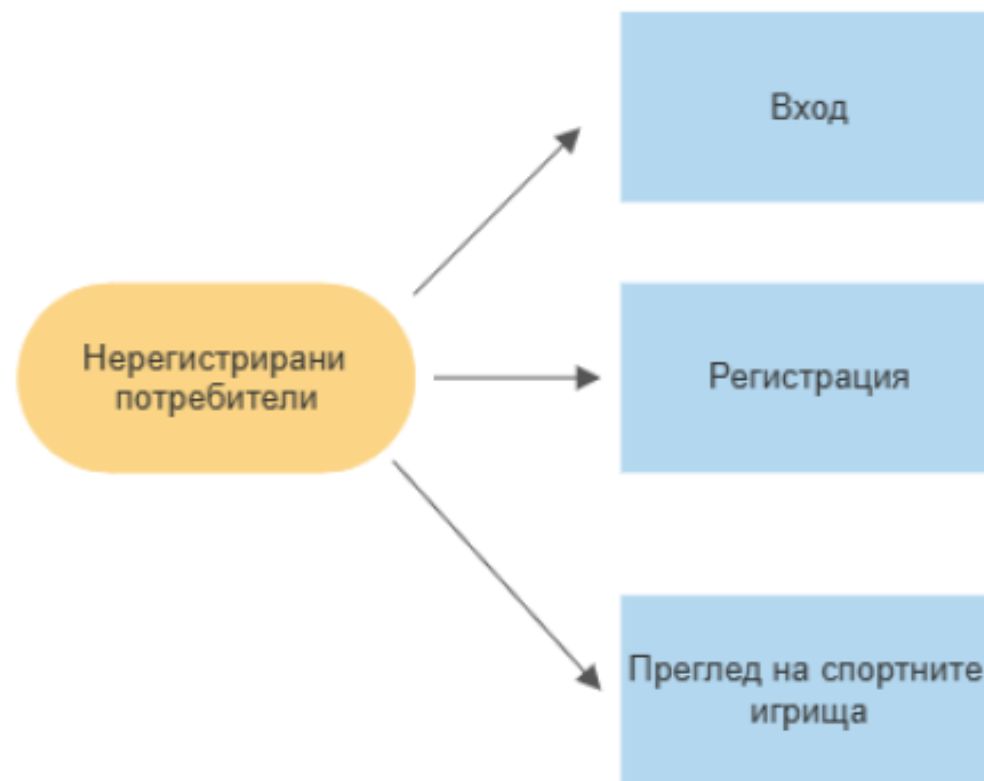
Цел и изходна постановка на дипломната работа

- Цел: Да се разработи уеб приложение за резервация на спортни съоръжения, използващо зададените в изходната постановка технологии
- Изходна постановка:
 - Java
 - Spring Boot
 - Hibernate
 - MySQL

Архитектура на разработваното приложение



Операции, извършвани от анонимен потребител



Компоненти на програмната реализация

- Обекти – Entities
- Услуги – Services
- Управление – Controllers
- Хранилища – Repositories
- Сигурност и нейната конфигурация - SecurityConfig

Обекти – „Entities“

- Обект „Потребители“ – Users Entity
- Обект „Роля“ – RoleEntity
- Обект „Спортно игрище“ - Field Entity
- Обект „Резервация“ - Reservation Entity

Хранилища – “Repositories”

- Хранилище за потребители – UserRepository
- Хранилище за роли – RoleRepository
- Хранилище за спортни игрища – FieldRepository
- Хранилище за резервации - ReservationRepository

Услуги – „Services“

Услуги за потребители – „UserService“

- Добавяне на клиент
- Добавяне на компания
- Добавяне на администратор

```
public String addRegisteredCustomer(Users user){  
    if(userRepository.findByEmail(user.getEmail()) != null) {  
        return "Вече съществува потребител с този имейл!";  
    }  
    if(userRepository.findByUsername(user.getUsername()) != null) {  
        return "Потребителското име е заето!";  
    }else {  
        Role roleCustomer = roleRepository.findByName("Customer");  
        user.addRole(roleCustomer);  
  
        user.setPassword(passwordEncoder.encode(user.getPassword()));  
        user.setActive(true);  
        userRepository.save(user);  
        return "Потребителят е добавен успешно!";  
    }  
}
```

The screenshot shows a REST client interface with the following details:

- Method:** POST
- URL:** http://localhost:8080/customer_register_customer
- Body:** A JSON object representing a user: `{ "username": "testclient2", "password": "123456", "firstName": "Izabel", "lastName": "Filipova", "email": "test@yahoo.bg", "phoneNumber": "0878175756" }`
- Status:** 200 OK
- Time:** 2.15 s
- Size:** 222 B
- Response Body:** Потребителят е добавен успешно!

- Преглед на всички компании, клиенти, админи
- Премахване на потребител
- Промяна на профилната информация

```
public String viewAllCompanies() {  
    if (userRepository.findAll().isEmpty()) {  
        return "Няма регистрирани потребители!";  
    } else {  
        Role roleCompany = roleRepository.findByName("Company");  
        Set<Role> searchRoles = new HashSet<>();  
        searchRoles.add(roleCompany);  
        Iterable<Users> companies = userRepository.findByRolesIn(searchRoles);  
        String companiesToString = "";  
        for (Users users : companies) {  
            companiesToString = companiesToString + users.toString() + " ";  
        }  
        if (companiesToString == "") {  
            companiesToString = "Няма регистрирани компании!";  
        }  
        return companiesToString;  
    }  
}
```

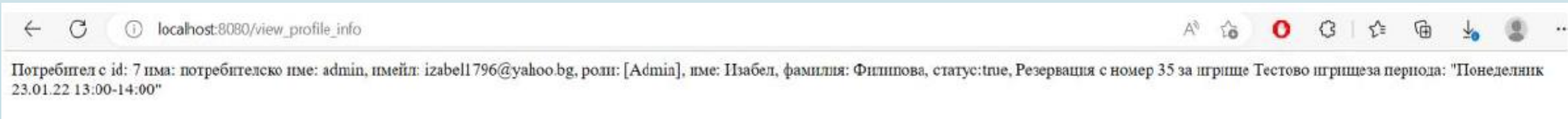
← ↻ ⓘ localhost:8080/admin/view_all_companies

Потребител с id: 15 има: потребителско име: test_company, имейл: test_company@yahoo.bg, роли: [Company], име: Georgi, фамилия: Georgiev, статус:true

➡ Преглед на профилна информация и резервации

```
public String viewProfileInfo(String email){
    Users currUser = userRepository.findByEmail(email);
    if(!(currUser == null)) {
        String userData = currUser.toString();
        String reservationsByUser = reservationService.getReservationHistory("{ " + currUser.getUsername() + " }");

        return userData + ", " + reservationsByUser;
    }
    else {
        return "Няма намерен потребител!";
    }
}
```



Услуга за Роли – „RoleService“

- Добавяне на роли, преглед на всички роли

```
public void addRole(Role role) {  
    if((roleRepository.findAll().contains(role))) {  
        throw new IllegalArgumentException("Тази роля вече съществува!");  
    }else {  
        roleRepository.save(role);  
    }  
}  
  
public Iterable<Role> getAllRoles(){  
    if(roleRepository.findAll() == null){  
        System.out.println("Няма добавени роли.");  
        return null;  
    }  
    return roleRepository.findAll();  
}
```

Услуга за спортни игрища – „FieldService“

```
public List<String> getAllFieldsByCity(String city){
    List<String> fieldsForCity = new ArrayList<>();

    if(fieldRepository.findAll() == null){
        fieldsForCity.add("Няма добавени игрища.");
    }else {
        List<Field> allFieldsIterable = fieldRepository.findAll();
        System.out.print(city);
        for (Field field : allFieldsIterable) {
            String adress = field.getLocation();
            if(adress.contains(city)) {
                fieldsForCity.add(field.toString());
                fieldsForCity.add(System.lineSeparator());
            }
        }
        if(fieldsForCity.isEmpty()) {
            fieldsForCity.add("Няма добавени игрища за този град!");
        }
    }
    return fieldsForCity;
}
```

- Добавяне на игрище
- Изтриване на игрище
- Преглед всички игрища
- Игрища по град
- Промяна състоянието на игрище

➤ Резервация, Промяна на информацията, Преглед на игрища по тип

13

```
public String reserve(String madeBy, int fieldId, String duration) {  
    Field fieldToReserve = fieldRepository.findByFieldId(fieldId);  
    Reservation reservation = new Reservation();  
  
    if(fieldToReserve.getState().contains(duration)){  
        return "Игрището вече е резервирано за този период. Моля изберете друг.";  
    }else {  
        reservation.setFieldName(fieldToReserve.getFieldName());  
        reservation.setMadeBy(madeBy);  
        reservation.setReservationDuration(duration);  
        reservationRepository.save(reservation);  
        long reservationId = reservationRepository  
            .findByFieldNameAndReservationDuration(fieldToReserve.getFieldName(), duration).getId();  
  
        String newState = String.format("Резервирано за " + duration);  
        changeFieldState(fieldId, newState);  
  
        return String.format("Игрището %s е резервирано от %s за периода %s. Вашият номер на резервацията е %d.",  
            fieldToReserve.getFieldName(), madeBy, duration, reservationId);  
    }  
}
```

PUT http://localhost:8080/reserve_field/{admin}/17 Send

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings Cookies

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** Beautify

1 "Понеделник 23.01.22 13:00-14:00"

Body Cookies (1) Headers (5) Test Results 🌐 Status: 200 OK Time: 2.78 s Size: 374 B Save Response

Pretty Raw Preview Visualize Text 🔍

1 Игрището Тестово игрище е резервирано от {admin} за периода "Понеделник 23.01.22 13:00-14:00". Вашият номер на резервацията е 34.

Услуга за резервации – „ReservationService“

```
public String reserveField(String madeBy, int id, String duration) {  
    return fieldService.reserve(madeBy, id, duration);  
}
```

➔ Методи:

```
public String cancelReservation(long reservationId, int fieldId) {  
    Reservation toCancel = reservationRepository.findById(reservationId);  
    if(toCancel == null) {  
        return "Няма резервация с този номер!";  
    }else {  
        String reservationPeriod = toCancel.getReservationDuration();  
        String stateToRemove = "Резервирано за " + reservationPeriod;  
        fieldService.changeFieldState(fieldId, stateToRemove);  
        reservationRepository.delete(toCancel);  
        return "Резервацията е успешно отменена!";  
    }  
}
```

- резервация
- Отмяна на резервация
- История на резервациите
- Преглед на всички резервации

Управление (Controllers)

Основно управление – „MainController“

Потребителското управление – „UserController“

```
@RequestMapping(value = "/view_profile_info", method = RequestMethod.GET)
@ResponseBody
public String customerProfile(Principal principal){
    String email = principal.getName();
    System.out.print(email);
    return userService.viewProfileInfo(email);
}

@PutMapping("/change_user_information/{email}")
public String changeUserInformation(@RequestBody Users newCompanyData, @PathVariable String email){
    return userService.changePersonalInformation(newCompanyData, email);
}

@DeleteMapping("/delete_field/{fieldId}")
public String removeField(@PathVariable int fieldId){
    fieldService.deleteField(fieldId);
    return String.format("Игрище с идентификационен номер %d е изтрито!", fieldId);
}

@PutMapping("/reserve_field/{madeBy}/{fieldId}")
public String reserveField(@PathVariable String madeBy, @PathVariable int fieldId, @RequestBody String duration){
    return reservationService.reserveField(madeBy, fieldId, duration);
}

@DeleteMapping("/cancel_reservation/{reservationId}/{fieldId}")
public String cancelReservation(@PathVariable long reservationId, @PathVariable int fieldId){
    return reservationService.cancelReservation(reservationId, fieldId);
}
```

Клас Конфигурация на сигурността – „SecurityConfig“

16

```
@Override
public void configure(WebSecurity web) throws Exception {
    web.ignoring()
        .antMatchers("/home")
        .antMatchers("/view_all_fields")
        .antMatchers("/view_all_fields_for_city/{city}")
        .antMatchers("/view_all_fields_for_type/{type}")
        .antMatchers("/company_add_company")
        .antMatchers("/customer_register_customer");
}

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

```
@Configuration
@EnableWebSecurity
public class SecurityConfig extends WebSecurityConfigurerAdapter {

    @Bean
    public UserDetailsService usersDetailsService() {
        return new CustomUserDetailsService();
    }

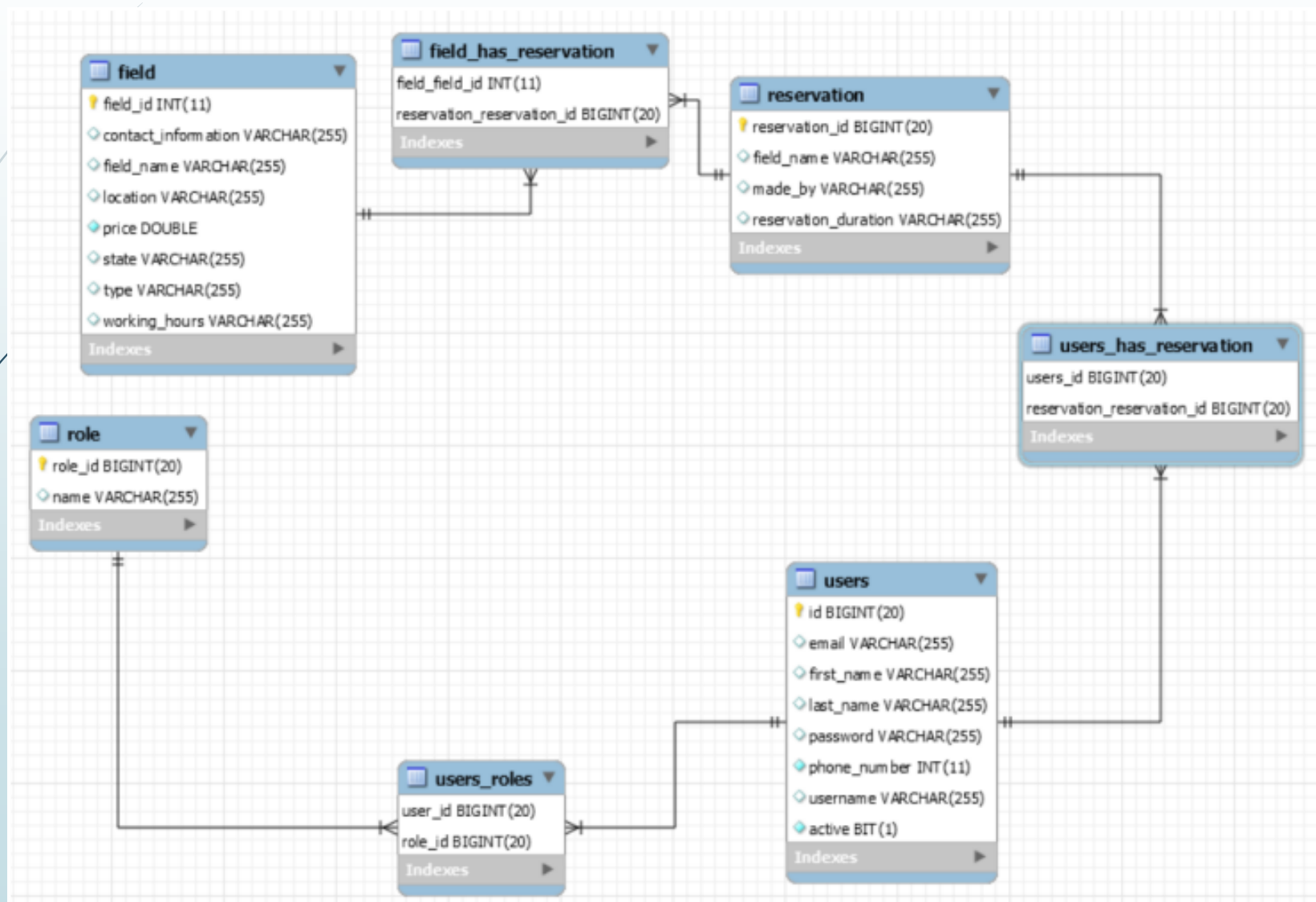
    @Bean
    public PasswordEncoder passwordEncoder() {
        return new BCryptPasswordEncoder();
    }

    @Bean
    public DaoAuthenticationProvider authenticationProvider() {
        DaoAuthenticationProvider authenticationProvider = new DaoAuthenticationProvider();
        authenticationProvider.setUserDetailsService(usersDetailsService());
        authenticationProvider.setPasswordEncoder(passwordEncoder());
        return authenticationProvider;
    }
}
```

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

    http.authorizeRequests()
        .antMatchers("/home").permitAll()
        .antMatchers("/view_profile_info", "/change_user_information/{email}").hasAnyAuthority("Admin", "Customer")
        .antMatchers("/reserve_field/{madeBy}/{fieldId}", "/cancel_reservation/{reservationId}/{fieldId}")
        .hasAnyAuthority("Admin", "Customer")
        .antMatchers("/delete_field/{fieldId}").hasAnyAuthority("Admin", "Company")
        .antMatchers("/admin/**", "add_role", "get_all_roles").hasAuthority("Admin")
        .antMatchers("/customer/**").hasAuthority("Customer")
        .antMatchers("/company/**").hasAuthority("Company")
        .anyRequest().authenticated()
        .and()
        .formLogin().permitAll()
        .and()
        .logout().permitAll();
}
```


Диаграма на връзките между обектите (ER – Entity Relationship diagram)



Изводи и бъдеща оптимизация

18

- **Зададените технологии са използвани за програмната реализация**
- **Подобренията на дипломният проект са спазени**
 - Свободна резервация/отмяна
 - Обхват на игрища в цялата страна
 - Свободен преглед на спортни игрища
 - Достъп до история на резервациите
- **Варианти за оптимизация**
 - Фронт-енд интерфейс с модул за график
 - Интеграция на „Google maps“
 - Чат платформа

Благодаря за вниманието!