

Университет ИТМО

Мегафакультет компьютерных технологий и управления

Факультет программной инженерии и компьютерной техники

ОТЧЕТ ПО ПРОЕКТИРОВАНИЮ И РАЗРАБОТКЕ ДИЗАЙН ПРОЕКТА

Курса «Рефакторинг баз данных и приложений»

Санкт-Петербург 2022

В данном спринте было проведено следующее:

1. Подключила информирование о назначениях и событиях на электронную почту с помощью протокола smtp.
2. Создание скрипта для создания БД.

Изменения в коде.

Создан скрипт для создания необходимых таблиц БД:

```
CREATE TABLE treatment(id SERIAL PRIMARY KEY, name TEXT, type TEXT);

CREATE TABLE user(id SERIAL PRIMARY KEY, name TEXT, password TEXT, token TEXT, position TEXT);

CREATE TABLE doctor(id SERIAL PRIMARY KEY, FOREIGN KEY(user_id) REFERENCES user(id));

CREATE TABLE patient(id SERIAL PRIMARY KEY, name TEXT, diagnosis TEXT, insurance INT, status TEXT, email TEXT);

CREATE table patient_doctor(FOREIGN KEY(patient_id) REFERENCES patient(id), FOREIGN KEY(doctor_id) REFERENCES doctor(id));

CREATE TABLE appointments(id SERIAL PRIMARY KEY, FOREIGN KEY(patient_id) REFERENCES patient(id), FOREIGN KEY(treatment_id) REFERENCES treatment(id), time_pattern TEXT, dose DOUBLE, start_date DATE, end_date DATE);


CREATE TABLE events(id SERIAL PRIMARY KEY, date DATE, status TEXT, reason TEXT, FOREIGN KEY(appointment_id) REFERENCES appointment(id));
```

Подключение смс-информирования.

1. Было необходимо отключить двухфазную аутентификацию и назначить приложение, которое будет иметь доступ к аккаунту по паролю.

← Пароли приложений

Пароли приложений позволяют входить в аккаунт Google на устройствах, которые не поддерживают двухэтапную аутентификацию. Такой пароль достаточно ввести один раз. [Подробнее...](#)

Ваши пароли приложений		
Название	Дата создания	Дата последнего использования
JavaSchoolBackend2Applicatid0:42	—	

2. Затем я установила необходимые параметры для подключения с помощью протокола smtp в файле application.properties.

```
spring.mail.host=smtp.gmail.com
spring.mail.port=587
spring.mail.username=marinakrayukh@gmail.com
spring.mail.password=
spring.mail.properties.mail.smtp.auth=true
spring.mail.properties.mail.smtp.starttls.enable=true
spring.mail.properties.mail.smtp.ssl.trust=smtp.gmail.com
```

3. Были созданы необходимые классы для работы приложения.
Класс EmailValidator (для проверки валидности введенного email).
Сделано это с помощью регулярного выражения.

```
public class EmailValidator {

    private Pattern pattern;
    private Matcher matcher;

    private static final String EMAIL_PATTERN =
        "^[_A-Za-z0-9-\\+](\\.[_A-Za-z0-9-]+)*@" +
        "[A-Za-z0-9-](\\.[A-Za-z0-9-]+)*(\\.[A-Za-z]{2,})$";

    public EmailValidator() {
        pattern = Pattern.compile(EMAIL_PATTERN);
    }

    public boolean validate(String email) {
        if(email != null) {
            matcher = pattern.matcher(email);
            return matcher.matches();
        }
        return false;
    }
}
```

Класс EmailMessage. Описывает тело сообщений для информирования.

```
public class EmailMessage {

    public String getMessageForAppointmentCreation(Appointment appointment) {
        return "Dear " + appointment.getPatient().getName() + ",\n" +
            "We've created appointment for you.\n" + getAppointmentInfo(appointment);
    }

    private String getAppointmentInfo(Appointment appointment) {
        return "Treatment: " + appointment.getTreatment().getName() + ",\n" +
            "Dose: " + (appointment.getTreatment().getType().equals("procedure") ? "-" :
            appointment.getDose()) + ",\n" +
            "Time Pattern: " + appointment.getTimePattern() + ",\n" +
            "Period: " + appointment.getStartDate() + " - " + appointment.getEndDate() + ".";
    }

    public String getMessageForEditAppointment(Appointment appointment) {
        return "Dear " + appointment.getPatient().getName() + ",\n" +
            "We've changed your appointment" + ",\n" + getAppointmentInfo(appointment);
    }
}
```

Класс EmailService. Содержит метод, который отправляет сообщение по заданному адресу.

```
@Service
public class EmailService {

    private JavaMailSender mailSender;

    @Autowired
    public EmailService(JavaMailSender mailSender) { this.mailSender = mailSender; }

    private static final Logger LOGGER = LogManager.getLogger(EmailService.class);

    void sendMessage(String email, String subject, String body) {
        SimpleMailMessage message = new SimpleMailMessage();
        message.setFrom("marinakrayukh@gmail.com");
        message.setTo(email);
        message.setText(body);
        message.setSubject(subject);
        mailSender.send(message);

        LOGGER.info("message sent");
    }
}
```