**Документация к проекту "Telegram-бот для записи на маникюр"**

**1. Общее описание проекта**

* **Название проекта**: Telegram-бот для записи на маникюр.
* **Описание**: Telegram-бот помогает пользователям записываться на маникюр, просматривать цены, отзывы, фото работ мастеров и управлять своими записями. Бот предназначен как для клиентов, так и для администраторов салона.
* **Цель проекта**: Автоматизировать процесс записи на услуги маникюра и упростить взаимодействие клиентов и администраторов через Telegram.
* **Основные технологии**:
  + Язык: Java 17
  + Фреймворки: Spring Boot, Spring Data JPA
  + Базы данных: MySQL
  + Взаимодействие с Telegram API через библиотеку TelegramBots
  + Docker для деплоя
  + Liquibase для управления миграциями БД

**2. Функциональные возможности**

**2.1 Функционал для всех пользователей (гостей)**

1. **Просмотр цен**: Клиенты могут видеть список доступных процедур и их стоимость.
2. **Отзывы**: Просмотр отзывов других клиентов о мастерах.
3. **Галерея работ**: Просмотр фотографий готовых работ с возможностью перейти на Instagram-страницу мастера для более подробного просмотра.
4. **Регистрация/вход**: Пользователь может создать новый аккаунт или войти в существующий.

**2.2 Функционал для зарегистрированных клиентов**

1. **Выбор мастера:** После авторизации клиент может выбрать мастера, к которому он хочет записаться.
2. **Просмотр календаря:** Возможность просматривать доступные даты на месяц вперёд для записи к выбранному мастеру.
3. **Выбор времени:** После выбора даты клиент видит доступные временные интервалы для записи.
4. **Выбор процедуры:** Клиент выбирает нужную процедуру, рядом с которой указана цена.
5. **Подтверждение записи:** После выбора процедуры клиент подтверждает запись.
6. **Напоминания:**
   * За день до процедуры бот отправляет уведомление о предстоящей записи.
   * За три часа до записи бот отправляет напоминание с указанием времени и процедуры.
7. **Уведомления для мастеров:**
   * Мастеру автоматически приходит уведомление о новой записи в момент её подтверждения клиентом.
   * Также мастеру отправляются напоминания о записи за день и за три часа до **процедуры.**
8. **Отмена/перенос записи**: Клиент может отменить или перенести свою запись.
9. **Просмотр истории записей:** Клиент может видеть свои прошлые и будущие записи.
10. **Оставить отзыв:** После завершения процедуры клиенту будет предложено оставить отзыв о мастере.

**2.3 Функционал для администратора**

1. **Открытие/закрытие дат и времени для записи**: Администратор может управлять доступными датами и временными слотами для записи клиентов.
2. **Запись клиента вручную**: Администратор может записать клиента на процедуру или отменить существующую запись.
3. **Управление доступом клиентов**: Возможность блокировать или удалять аккаунты клиентов.
4. **Отправка сообщений клиентам**: Администратор может отправлять клиентам сообщения через бота (например, напоминания или важные уведомления).
5. **Просмотр отзывов**: Администратор может просматривать и управлять отзывами клиентов.

**2.4 Дополнительные возможности для администратора**

1. **Управление мастерами**:
   * **Добавление мастера**: Администратор может добавлять новых мастеров, указав их имя, описание и другие данные (например, контактные данные, ссылки на социальные сети).
   * **Удаление мастера**: Администратор может удалять мастера из базы данных. При этом все записи и процедуры, связанные с мастером, будут также удалены или переназначены другим мастерам (по выбору администратора).
   * **Просмотр списка мастеров**: Администратор может видеть полный список мастеров с их данными.
2. **Управление процедурами**:
   * **Добавление процедуры**: Администратор может добавлять процедуры для каждого мастера отдельно, указав название, описание и цену.
   * **Удаление процедуры**: Администратор может удалять процедуры, которые больше не предоставляются.
   * **Редактирование процедуры**: Администратор может изменять существующие процедуры, обновляя их описание или цену.
   * **Привязка процедур к мастерам**: Каждый мастер имеет индивидуальный набор процедур, которые он предоставляет.

**3. Технические требования**

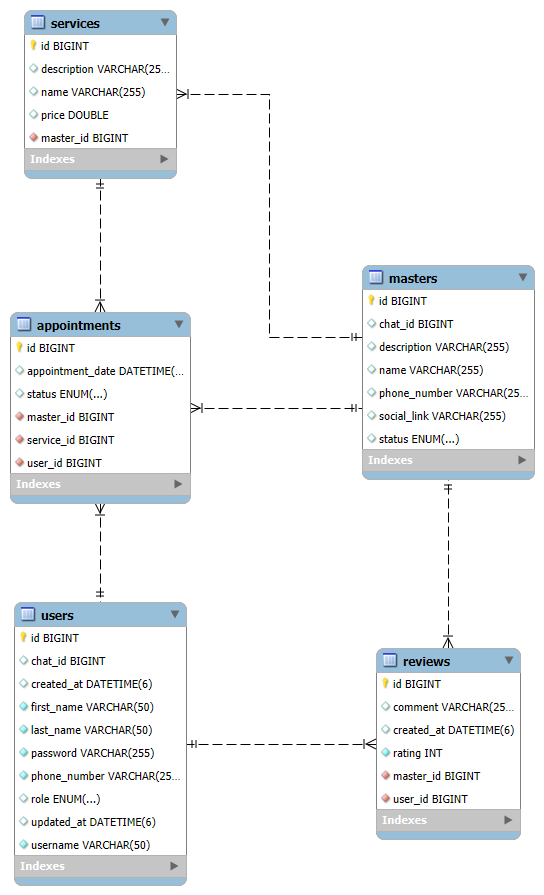
* **Telegram API**: Для взаимодействия с Telegram.
* **Java 17**: Основной язык разработки.
* **Spring Boot**: Для создания REST API и взаимодействия с клиентом через бота.
* **Spring Data JPA**: Для работы с базой данных.
* **MySQL**: Основная база данных для хранения информации о клиентах, мастерах, записях и процедурах.
* **Docker**: Для удобного деплоя и контейнеризации приложения.
* **Liquibase**: Для управления миграциями базы данных.
* **Интеграция с Instagram**: Для перенаправления на страницу мастера через Telegram.

**4. Архитектура проекта**

**4.1 Основные компоненты**

* **BotController**: Основной контроллер для обработки входящих сообщений от Telegram.
* **UserService**: Логика работы с пользователями — регистрация, вход, просмотр записей.
* **AdminService**: Логика для администратора — управление записями, блокировка пользователей.
* **AppointmentService**: Управление записями на процедуры (выбор даты, времени, процедуры).
* **NotificationService**: Система напоминаний о предстоящих записях.
* **ReviewService**: Логика для управления отзывами клиентов.
* **MasterService**: Управление мастерами — добавление, удаление и просмотр мастеров.
* **ProcedureService**: Управление процедурами — добавление, удаление и редактирование процедур.
* **Database (MySQL)**: Хранение пользователей, мастеров, расписания, процедур, записей и отзывов.

**4.2 Диаграмма базы данных (ERD)**



* **Пользователи (users)**: Таблица с информацией о клиентах и администраторах.
* **Мастера (masters)**: Таблица с информацией о мастерах.
* **Процедуры (services)**: Список процедур с указанием цен, привязанных к мастерам.
* **Записи (appointments)**: Сохранение записей клиентов к мастерам с указанием даты, времени и процедуры.
* **Отзывы (reviews)**: Хранение отзывов клиентов о мастерах.
* **4.3 Базы данных**
* **4.3.1 Таблица "Пользователи" (users)**

| **Поле** | **Тип данных** | **Описание** |
| --- | --- | --- |
| id | BIGINT | Уникальный идентификатор пользователя (первичный ключ) |
| username | VARCHAR(255) | Имя пользователя (уникальное) |
| password | VARCHAR(255) | Пароль пользователя |
| role | ENUM | Роль пользователя (CLIENT, ADMIN) |
| chat\_id | BIGINT | Идентификатор чата в Telegram |
| phone\_number | VARCHAR(15) | Номер телефона пользователя |
| created\_at | TIMESTAMP | Дата и время создания записи |
| updated\_at | TIMESTAMP | Дата и время последнего обновления записи |
| first\_name | VARCHAR(50) | Имя пользователя |
| last\_name | VARCHAR(50) | Фамилия пользователя |

* **4.3.2 Таблица "Мастера" (masters)**

| **Поле** | **Тип данных** | **Описание** |
| --- | --- | --- |
| id | BIGINT | Уникальный идентификатор мастера (первичный ключ) |
| name | VARCHAR(255) | Имя мастера |
| description | TEXT | Описание мастера (например, специализация или услуги) |
| social\_link | VARCHAR(255) | Ссылка на социальные сети |
| status | ENUM | Статус мастера (ACTIVE, INACTIVE) |
| chat\_id | BIGINT | Идентификатор чата в Telegram |
| phone\_number | VARCHAR(15) | Номер телефона мастера |

* **4.3.3 Таблица "Процедуры" (services)**

| **Поле** | **Тип данных** | **Описание** |
| --- | --- | --- |
| id | BIGINT | Уникальный идентификатор процедуры (первичный ключ) |
| name | VARCHAR(255) | Название процедуры |
| description | TEXT | Описание процедуры |
| price | DECIMAL(10, 2) | Цена процедуры |
| master\_id | BIGINT | Идентификатор мастера (внешний ключ) |

* **4.3.4 Таблица "Записи" (appointments)**

| **Поле** | **Тип данных** | **Описание** |
| --- | --- | --- |
| id | BIGINT | Уникальный идентификатор записи (первичный ключ) |
| user\_id | BIGINT | Идентификатор пользователя (внешний ключ) |
| master\_id | BIGINT | Идентификатор мастера (внешний ключ) |
| service\_id | BIGINT | Идентификатор процедуры (внешний ключ) |
| appointment\_date | TIMESTAMP | Дата и время записи |
| status | ENUM | Статус записи (CONFIRMED, CANCELLED, COMPLETED) |

* **4.3.5 Таблица "Отзывы" (reviews)**

| **Поле** | **Тип данных** | **Описание** |
| --- | --- | --- |
| id | BIGINT | Уникальный идентификатор отзыва (первичный ключ) |
| user\_id | BIGINT | Идентификатор пользователя (внешний ключ) |
| master\_id | BIGINT | Идентификатор мастера (внешний ключ) |
| rating | INT | Оценка отзыва (от 1 до 5) |
| comment | TEXT | Комментарий отзыва |
| created\_at | TIMESTAMP | Дата и время создания отзыва |

**5. Примеры взаимодействия администратора с мастерами и процедурами**

1. **Добавление мастера**: Администратор через интерфейс бота или панель управления может добавить нового мастера.
2. **Удаление мастера**: Администратор удаляет мастера, и его процедуры либо удаляются, либо переназначаются другим мастерам.
3. **Добавление процедуры**: Для каждого мастера можно добавить индивидуальный набор процедур с указанием цены и описания.
4. **Редактирование процедуры**: Администратор может редактировать название, описание и цену существующих процедур.

**6. Использование @BotFather**

Для создания и настройки бота в Telegram используется @BotFather:

1. Найти @BotFather в Telegram и создать нового бота через команду /newbot.
2. Придумать имя для бота и получить токен для доступа к API.
3. Настроить токен в Spring Boot проекте для интеграции с Telegram API.

**7. Сценарии использования**

1. **Запись клиента на процедуру**: Клиент выбирает мастера, дату, время и процедуру, затем подтверждает запись.
2. **Отмена записи**: Клиент может отменить свою запись через соответствующую команду.
3. **Напоминания**: Бот автоматически отправляет клиенту уведомление за день и за три часа до процедуры.
4. **Оставление отзыва**: После завершения процедуры клиент получает запрос на оставление отзыва.
5. **Управление мастерами**: Администратор добавляет, удаляет или обновляет мастеров через бот или интерфейс.
6. **Управление процедурами**: Администратор добавляет, удаляет или редактирует процедуры для каждого мастера.

**8. План разработки**

1. **Неделя 1**: Создание бота через @BotFather, интеграция с Telegram API.
2. **Неделя 2**: Разработка функционала для клиентов: регистрация, запись на процедуры.
3. **Неделя 3**: Разработка административной части: управление мастерами, процедурами, записями и отзывами.
4. **Неделя 4**: Реализация системы напоминаний и добавление возможности редактировать процедуры и мастеров.
5. **Неделя 5**: Тестирование, отладка проекта и деплой на сервер.