**Documentation for the Project "Telegram Bot for Manicure Appointments"**

**1. General Project Description**

**• Project Name: Telegram Bot for Manicure Appointments.  
Description: This document describes the Telegram bot project for manicure appointments. The bot automates the booking process, record management, and interaction with masters and administrators.  
Project Goal: The project's goal is to create a user-friendly Telegram bot that allows users to book manicure services, manage appointments, receive notifications, and communicate with masters and administrators.**

**• Key Technologies:**

* **Language: Java 17**
* **Frameworks: Spring Boot, Spring Data JPA**
* **Databases: MySQL**
* **Interaction with Telegram API: Through the TelegramBots library**
* **Deployment: Docker**
* **Database Migrations: Liquibase**

**2. Functional Features**

**2.1. For Clients:**

* **Registration in the bot**
* **Viewing the list of services**
* **Booking a service**
* **Selecting a master, date, and time**
* **Managing active, completed, and canceled appointments**
* **Leaving reviews about masters**
* **Receiving notifications one day and three hours before the appointment**

**2.2. For Masters:**

* **Viewing their appointments**
* **Rescheduling or canceling appointments**
* **Managing schedules**
* **Communicating with clients**

**2.3. For Administrators:**

* **Adding/removing masters**
* **Adding/removing services**
* **Managing client appointments**
* **Sending notifications to clients and masters**

**3. Technical Requirements**

* **Telegram API: For interaction with Telegram**
* **Java 17: Main development language**
* **Spring Boot: For creating REST APIs and interacting with clients via the bot**
* **Spring Data JPA: For database management**
* **MySQL: Main database for storing client, master, appointment, and service data**
* **Docker: For convenient deployment and containerization**
* **Liquibase: For managing database migrations**
* **Instagram Integration: For redirecting to master pages through Telegram**

**4. Project Architecture**

**4.1 Entity Classes (Models)**

**User:**

* **Represents a bot user (client)**
* **Attributes: id, firstName, lastName, phone, language**

**Master:**

* **Represents a master providing services**
* **Attributes: id, name, specialization, availability**

**Services:**

* **Represents services offered to clients**
* **Attributes: id, name, description, price, duration**

**Review:**

* **Represents client reviews of masters and services**
* **Attributes: id, rating, comment, userId, masterId**

**Appointment:**

* **Represents a service appointment**
* **Attributes: id, userId, masterId, serviceId, appointmentDate, status (e.g., active, completed, canceled)**

**4.2 Repositories**

* **UserRepository: Interface for interacting with the User entity in the database**
* **MasterRepository: Interface for managing master data**
* **ServiceRepository: Interface for managing service data**
* **ReviewRepository: Interface for managing reviews**
* **AppointmentRepository: Interface for managing appointments**

**4.3 Controllers**

* **UserController: Manages users (creation, retrieval, deletion)**
* **MasterController: Manages masters (creation, updates, deletion)**
* **ServiceController: Manages services (creation, updates, deletion)**
* **ReviewController: Manages reviews (creation, retrieval, deletion)**
* **AppointmentController: Manages appointments (creation, retrieval, deletion)**
* **AdminController: Manages administrative tasks, restricted to ROLE\_ADMIN**

**4.4 Database Tables**

**4.3.1 "Users" Table (users)**

| **Field** | **Data Type** | **Description** |
| --- | --- | --- |
| **id** | **BIGINT** | **Unique user identifier (primary key)** |
| **language** | **VARCHAR(10)** | **User's preferred language** |
| **role** | **ENUM** | **User role (CLIENT, ADMIN)** |
| **chat\_id** | **BIGINT** | **Telegram chat ID** |
| **phone\_number** | **VARCHAR(15)** | **User's phone number** |
| **created\_at** | **TIMESTAMP** | **Record creation timestamp** |
| **updated\_at** | **TIMESTAMP** | **Last record update timestamp** |
| **first\_name** | **VARCHAR(50)** | **User's first name** |
| **last\_name** | **VARCHAR(50)** | **User's last name** |

**4.3.2 "Technicians" Table (masters)**

| **Field** | **Data Type** | **Description** |
| --- | --- | --- |
| **id** | **BIGINT** | **Unique technician identifier (PK)** |
| **name** | **VARCHAR(255)** | **Technician's name** |
| **description** | **TEXT** | **Technician's description/services** |
| **social\_link** | **VARCHAR(255)** | **Social media link** |
| **status** | **ENUM** | **Technician status (ACTIVE, INACTIVE)** |
| **chat\_id** | **BIGINT** | **Telegram chat ID** |
| **phone\_number** | **VARCHAR(15)** | **Technician's phone number** |

**4.3.3 "Services" Table (services)**

| **Field** | **Data Type** | **Description** |
| --- | --- | --- |
| **id** | **BIGINT** | **Unique service identifier (PK)** |
| **name\_ru** | **VARCHAR(255)** | **Service name in Russian** |
| **name\_uk** | **VARCHAR(255)** | **Service name in Ukrainian** |
| **name\_en** | **VARCHAR(255)** | **Service name in English** |
| **description\_ru** | **TEXT** | **Service description in Russian** |
| **description\_uk** | **TEXT** | **Service description in Ukrainian** |
| **description\_en** | **TEXT** | **Service description in English** |
| **price** | **DOUBLE** | **Service price** |
| **master\_id** | **BIGINT** | **Technician ID (FK)** |

**4.3.4 "Appointments" Table (appointments)**

| **Field** | **Data Type** | **Description** |
| --- | --- | --- |
| **id** | **BIGINT** | **Unique appointment ID (PK)** |
| **chat\_id** | **BIGINT** | **User's Telegram chat ID** |
| **user\_id** | **BIGINT** | **User ID (FK)** |
| **master\_id** | **BIGINT** | **Technician ID (FK)** |
| **service\_id** | **BIGINT** | **Service ID (FK)** |
| **appointment\_date** | **TIMESTAMP** | **Appointment date and time** |
| **status** | **ENUM** | **Appointment status (CONFIRMED, CANCELLED, COMPLETED)** |

**4.3.5 "Reviews" Table (reviews)**

| **Field** | **Data Type** | **Description** |
| --- | --- | --- |
| **id** | **BIGINT** | **Unique review ID (PK)** |
| **user\_id** | **BIGINT** | **User ID (FK)** |
| **master\_id** | **BIGINT** | **Technician ID (FK)** |
| **rating** | **INT** | **Rating (1 to 5)** |
| **comment** | **TEXT** | **Review comment** |
| **created\_at** | **TIMESTAMP** | **Review creation timestamp** |

**4.3.6 "Help Requests" Table (help\_requests)**

| **Field** | **Data Type** | **Description** |
| --- | --- | --- |
| **id** | **BIGINT** | **Unique request ID (PK)** |
| **user\_id** | **BIGINT** | **Requesting user ID** |
| **admin\_id** | **BIGINT** | **Administrator responsible (nullable)** |
| **help\_question** | **TEXT** | **User's question** |
| **admin\_response** | **TEXT** | **Admin's response** |
| **created\_at** | **TIMESTAMP** | **Request creation timestamp** |
| **closed\_at** | **TIMESTAMP** | **Request closure timestamp** |
| **status** | **VARCHAR(10)** | **Request status (pending, open, closed)** |

**4.3.7 "Time Slots" Table (time\_slots)**

| **Field** | **Data Type** | **Description** |
| --- | --- | --- |
| **id** | **BIGINT** | **Unique time slot ID (PK)** |
| **available\_date\_id** | **BIGINT** | **Linked available date ID (FK)** |
| **time** | **TIME** | **Appointment time** |
| **is\_booked** | **BOOLEAN** | **Whether the slot is booked** |
| **master\_id** | **BIGINT** | **Technician ID linked to the slot** |

**4.3.8 "Available Dates" Table (available\_dates)**

| **Field** | **Data Type** | **Description** |
| --- | --- | --- |
| **id** | **BIGINT** | **Unique date ID (PK)** |
| **date** | **DATE** | **Available appointment date** |
| **master\_id** | **BIGINT** | **Technician ID linked to the date** |

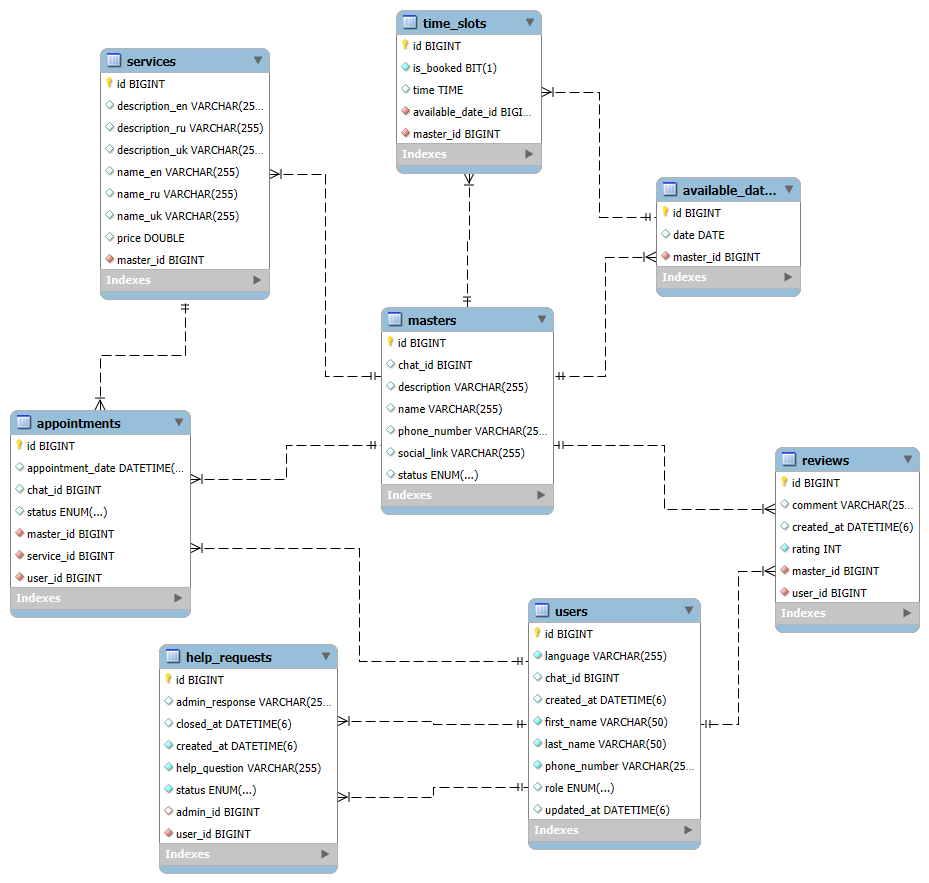
**4.5 Configurations**

* **SecurityConfig: Configures project security, authentication, and role-based access**
* **BotConfig: Configures Telegram bot settings, including bot token and name**

**4.6 Service Layer**

* **UserService: Manages user-related logic (registration, profile updates)**
* **MasterService: Manages master-related logic (adding/removing, updating availability)**
* **ServiceService: Manages service-related logic (adding/removing/updating services)**
* **ReviewService: Manages review-related logic (adding and retrieving reviews)**
* **AppointmentService: Manages appointment logic (creating, canceling, rescheduling)**
* **NotificationService: Handles sending notifications to users and masters**

**4.7 Database Diagram (ERD)**



* **Users Table (users): Stores client and administrator information**
* **Masters Table (masters): Stores master details**
* **Services Table (services): Stores procedures with prices linked to masters**
* **Appointments Table (appointments): Stores client appointments with masters**
* **Reviews Table (reviews): Stores client reviews about masters**
* **Help Requests Table (help\_requests): Handles client queries with administrator assignment**
* **Time Slots Table (time\_slots): Stores available time slots for appointments**
* **Available Dates Table (available\_dates): Stores dates available for bookings**

**5. Administrator and Master Interactions**

1. **Adding a Master: Administrators can add new masters through the bot or admin panel.**
2. **Removing a Master: Administrators can remove masters, reassign services, or delete them.**
3. **Adding a Service: Administrators can add individual services for each master.**
4. **Editing a Service: Administrators can edit service details such as name, description, and price.**

**6. API Endpoints**

* **POST /api/appointments: Create an appointment**
* **GET /api/appointments: View all appointments**
* **DELETE /api/appointments/{id}: Delete an appointment**
* **POST /api/users: Register a user**
* **GET /api/users: Get user information**

**7. Use Cases**

1. **Client Booking: Clients choose masters, dates, and services, then confirm the appointment.**
2. **Appointment Cancellation: Clients can cancel their appointments through a specific command.**
3. **Reminders: The bot sends notifications one day and three hours before appointments.**
4. **Leaving a Review: After the service, clients receive a request to leave a review.**
5. **Master Management: Administrators can add, remove, or update masters.**
6. **Service Management: Administrators can manage services for each master.**

**8. Development Plan**

1. **Week 1: Create the bot via @BotFather and integrate it with the Telegram API**
2. **Week 2: Develop client functionalities (registration, service booking)**
3. **Week 3: Develop administrative functionalities (managing masters, services, appointments, and reviews)**
4. **Week 4: Implement notifications, service editing, and master updates**
5. **Week 5: Test, debug, and deploy the project to the server**