# Health Connect REST API Documentation

## Introduction

[Introduction to the Health Connect API system, its purpose, and overall architecture]

## Technologies Used

[List of technologies, frameworks, languages, and tools used]

## Authentication Flow

The API implements token-based authentication with OTP verification for both users and partners.

### Authentication Process

1. Registration or login request (phone number provided)
2. OTP sent to user/partner phone
3. OTP verification
4. Token issuance (access + refresh tokens)
5. Token usage for API access

## API Routes

### 1. User Management

#### 1.1 Register User

* **Endpoint:** POST /auth/register/user/
* **Description:** Registers a new user in the system
* **Request Body:**

{

"phone\_number": "9876543210",

"email": "user@example.com",

"full\_name": "John Doe"

}

* **Response:** Returns verification ID for OTP verification

#### 1.2 Verify User OTP

* **Endpoint:** POST /auth/verify/user/
* **Description:** Verifies the OTP sent during registration
* **Request Body:**

json

Copy

{

"phone\_number": "9876543210",

"verification\_id": "123456",

"otp": "123456"

}

* **Response:** Returns authentication tokens upon successful verification

#### 1.3 User Login (Request OTP)

* **Endpoint:** POST /auth/login/user/
* **Description:** Initiates the login process by requesting an OTP
* **Request Body:**

json

Copy

{

"phone\_number": "9876543210"

}

* **Response:** Returns verification ID for OTP verification

#### 1.4 Verify User Login OTP

* **Endpoint:** POST /auth/verify/login/user/
* **Description:** Verifies the OTP sent during login
* **Request Body:**

json

Copy

{

"phone\_number": "9876543210",

"verification\_id": "123456",

"otp": "123456"

}

* **Response:** Returns authentication tokens upon successful verification

#### 1.5 User Home

* **Endpoint:** GET /user/home/
* **Description:** Retrieves user home screen data
* **Headers:** Authorization: Bearer {access\_token}
* **Response:** Returns user profile information and relevant data

### 2. Partner Management

#### 2.1 Register Partner

* **Endpoint:** POST /auth/register/partner/
* **Description:** Registers a new partner in the system
* **Request Body:**

json

Copy

{

"phone\_number": "9876543211",

"email": "partner@example.com",

"full\_name": "Dr. Smith",

"education": "MBBS",

"experience": "10"

}

* **Response:** Returns verification ID for OTP verification

#### 2.2 Register Partner (with Medical Certificate)

* **Endpoint:** POST /auth/register/partner/
* **Description:** Registers a new partner with medical certificate upload
* **Request Body:** Form-data with fields:
  + phone\_number: "9876543211"
  + email: "[partner@example.com](mailto:partner@example.com)"
  + full\_name: "Dr. Smith"
  + education: "MBBS"
  + medical\_certificate: [file upload]
  + experience: "10"
* **Response:** Returns verification ID for OTP verification

[Additional partner management endpoints are documented in the complete version]

### 3. Service Management

#### 3.1 Get Service Types

* **Endpoint:** GET /user/services/
* **Description:** Retrieves available service types
* **Headers:** Authorization: Bearer {access\_token}
* **Response:** Returns list of available service types

### 4. Booking Management

#### 4.1 Create Booking

* **Endpoint:** POST /user/bookings/create/
* **Description:** Creates a new booking request
* **Headers:** Authorization: Bearer {access\_token}
* **Request Body:**

json

Copy

{

"service\_type": 1,

"partner\_type": "trained",

"is\_instant": true,

"hours": 4,

"user\_location": "123 Main Street, Anytown",

"hospital\_location": "General Hospital, Anytown"

}

* **Response:** Returns booking details and status

[Additional booking management endpoints are documented in the complete version]

### 5. Session Management

#### 5.1 Token Refresh

* **Endpoint:** POST /auth/token/refresh/
* **Description:** Refreshes an expired access token
* **Request Body:**

json

Copy

{

"refresh": "{refresh\_token}"

}

* **Response:** Returns a new access token

#### 5.2 Token Verify

* **Endpoint:** POST /auth/token/verify/
* **Description:** Verifies if a token is valid
* **Request Body:**

json

Copy

{

"token": "{access\_token}"

}

* **Response:** Returns 200 OK if token is valid

#### 5.3 Logout

* **Endpoint:** POST /auth/logout/
* **Description:** Invalidates the refresh token
* **Headers:** Authorization: Bearer {access\_token}
* **Request Body:**

json

Copy

{

"refresh": "{refresh\_token}"

}

* **Response:** Returns confirmation of logout success

## Data Models

[Key data structures and their relationships]

## Error Handling

[Standard error responses and codes]

## Rate Limiting

[Information about API rate limits]

## Versioning

[API versioning information]