Health Appointment System: Application Overview and Workflow

Application Overview

The Health Appointment System is a comprehensive web-based platform that connects patients with healthcare providers. Built with Flask, this application streamlines the healthcare appointment booking process while providing robust management tools for patients, doctors, and administrators.

Core User Roles

---

1. Patients

End users seeking medical care who can search for doctors, book appointments, and manage their health information.

---

2. Doctors

Healthcare providers who can manage their availability, view appointments, and maintain their professional profiles.

---

3. Administrators

System managers who verify doctors, manage user accounts, and oversee the platform's operation.

Health Appointment System: Sitemap

Based on my analysis of the routes and templates, here's a comprehensive sitemap of the Health Appointment System:

Public Area

Home

- **URL**: `/`
- **Purpose: Landing page with service overview**
- **Features: Hero section, feature highlights, how it works, testimonials**

About

- **URL**: `/about`
- **Purpose: Information about the service**
- **Features: Mission statement, team information, service description**

Contact

- **URL: /contact**
- **Purpose: Contact form for inquiries**
- **Features: Contact form, contact information**

Authentication

Registration Choice

- **URL: /auth/register**
- **Purpose: Select account type**
- **Features: Patient or doctor registration options**

Patient Registration

- **URL: /auth/register/patient**
- **Purpose: Create patient account**
- **Features: Registration form with personal and basic health information**

Doctor Registration

- **URL: /auth/register/doctor**
- **Purpose: Create doctor account**
- **Features: Registration form with professional credentials and document upload**

Email Verification

- **URL: /auth/verify-email/<token>**
- **Purpose: Verify user email address**
- **Features: Verification confirmation, next steps**

Phone Verification

- **URL: /auth/verify-phone/<user_id>**
- **Purpose: Verify user phone number**
- **Features: Verification code entry, resend option**

Pending Verification

- **URL: /auth/pending-verification/<user_id>**
- **Purpose: Waiting screen for verification**
- **Features: Status information, next steps**

Login

- **URL: /auth/login**
- **Purpose: User authentication**
- **Features: Login form, forgot password link**

Forgot Password

- **URL: /auth/forgot-password**
- **Purpose: Password reset request**
- **Features: Email entry form**

Reset Password

- **URL: /auth/reset-password/<token>**
- **Purpose: Set new password**
- **Features: New password form**

Logout

- **URL: /auth/logout**
- **Purpose: End user session**
- **Features: Redirect to home**

Patient Area

Patient Dashboard

- **URL: /patient/dashboard**
- **Purpose: Patient control center**
- **Features: Appointment statistics, upcoming appointments, profile information**

Patient Appointments

- **URL: /patient/appointments**
- **Purpose: View and manage appointments**
- **Features: Appointment listings with filters, action buttons**

Cancel Appointment

- **URL: /patient/appointments/<appointment_id>/cancel**
- **Purpose: Cancel a booked appointment**
- **Features: Cancellation form, reason entry**

Doctor Area

Doctor Dashboard

- **URL: /doctor/dashboard**
- **Purpose: Doctor control center**
- **Features: Appointment statistics, today's schedule, upcoming appointments**

Doctor Profile Management

- **URL: /doctor/profile**
- **Purpose: Edit professional profile**
- **Features: Profile form, document upload**

Doctor Availability Management

- **URL: /doctor/availability**
- **Purpose: Set working hours**
- **Features: Weekly schedule, time slot management**

Doctor Appointments

- **URL: /doctor/appointments**
- **Purpose: View and manage appointments**
- **Features: Calendar view, appointment listings**

Complete Appointment

- **URL: /doctor/appointments/<appointment_id>/complete**
- **Purpose: Mark appointment as completed**
- **Features: Completion form, notes entry**

Admin Area

Admin Dashboard

- **URL: /admin/dashboard**
- **Purpose: Admin control center**

- **Features: System statistics, recent users, pending verifications**

Doctor Verification

- **URL: /admin/doctor-verification**
- **Purpose: Verify doctor credentials**
- **Features: Verification listings, document review**

Verify Doctor

- **URL: /admin/verify-doctor/<doctor_id>/<action>**
- **Purpose: Approve or reject verification**
- **Features: Verification action, notes entry**

Shared Functionality

Find Doctors

- **URL: /main/find-doctors**
- **Purpose: Search for doctors**
- **Features: Search filters, doctor listings**

Doctor Profile

- **URL: /main/doctor-profile/<doctor_id>**
- **Purpose: View doctor information**
- **Features: Doctor profile, credentials, reviews**

Book Appointment

- **URL: /main/book-appointment/<doctor_id>**
- **Purpose: Schedule appointment**
- **Features: Date selection, time slot selection, reason entry**

Notifications

- **URL: /main/notifications**
- **Purpose: View system notifications**
- **Features: Notification listings, mark as read**

Unread Notifications Count

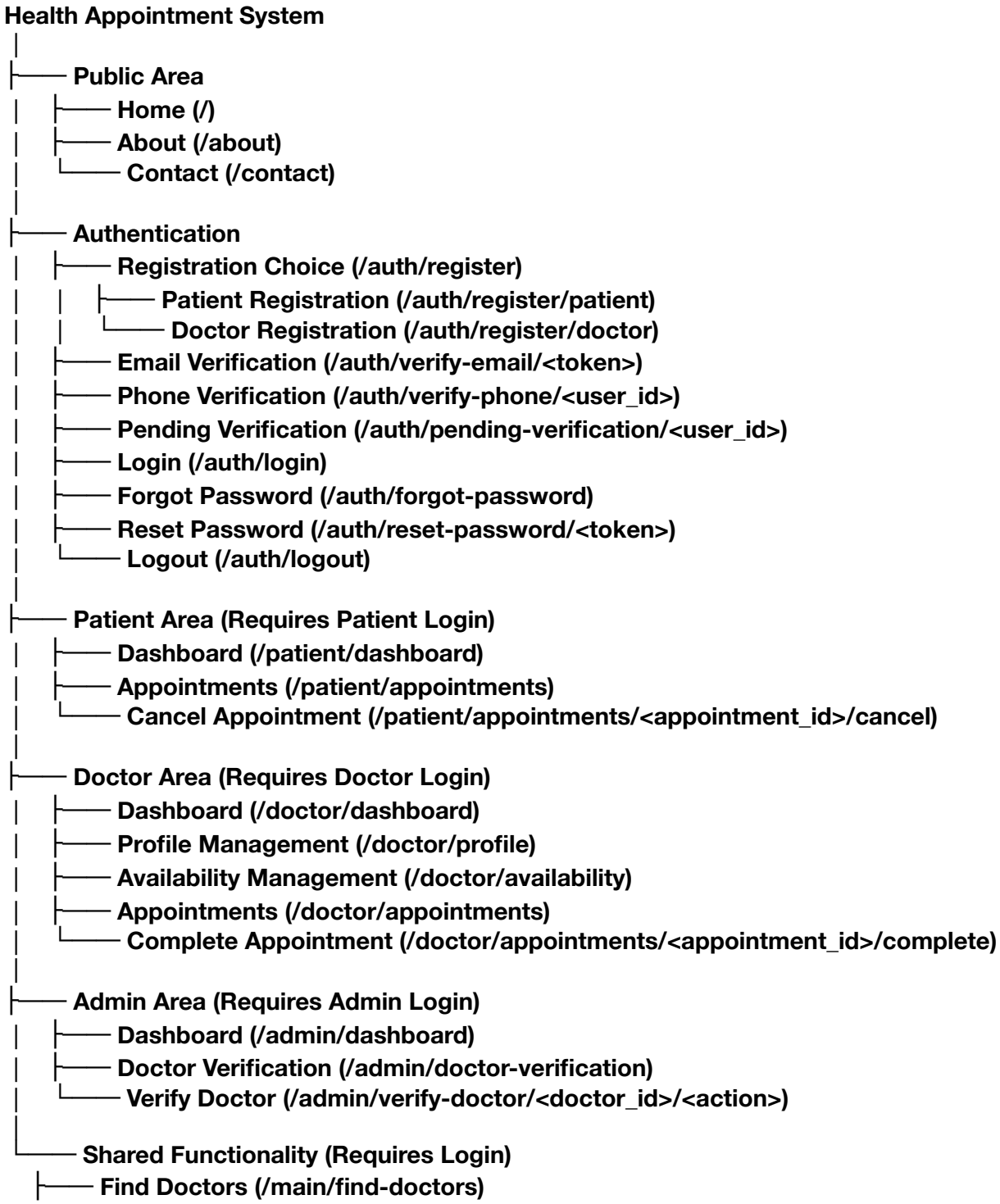
- **URL: /main/unread-notifications-count**
- **Purpose: API endpoint for notification count**
- **Features: JSON response with count**

API Endpoints

Available Slots

- **URL: /main/get-available-slots/<doctor_id>**
- **Purpose: Get doctor availability**
- **Features: JSON response with available time slots**

Hierarchical Structure



├── Doctor Profile (/main/doctor-profile/<doctor_id>)
├── Book Appointment (/main/book-appointment/<doctor_id>)
└── Notifications (/main/notifications)

Health Appointment System: Detailed Data Structure

Based on my analysis of the models.py file, here's a comprehensive breakdown of the data structure used in the Health Appointment System:

Enumerations

1. UserType
   - **PATIENT**: Regular users seeking medical care
   - **DOCTOR**: Healthcare providers
   - **ADMIN**: System administrators
2. VerificationStatus
   - **PENDING**: Initial state for doctor verification
   - **VERIFIED**: Doctor credentials confirmed
   - **REJECTED**: Doctor credentials rejected
3. AppointmentStatus
   - **PENDING**: Initial state for new appointments
   - **CONFIRMED**: Appointment confirmed by doctor
   - **CANCELLED**: Appointment cancelled by either party
   - **COMPLETED**: Appointment successfully completed

Database Models

1. User

Purpose: Base user model for authentication and common user data **Table**: `users` **Fields**:

- `id`: Integer, Primary Key
- `email`: String(120), Unique, Not Null
- `phone`: String(20), Unique, Not Null
- `password_hash`: String(128), Not Null
- `first_name`: String(50), Not Null
- `last_name`: String(50), Not Null
- `user_type`: Enum(UserType), Not Null
- `is_active`: Boolean, Default=False
- `created_at`: DateTime, Default=Current time
- `updated_at`: DateTime, Auto-updates
- `phone_verified`: Boolean, Default=False
- phone_verification_code: String(6)
- phone_verification_sent_at: DateTime
- `email_verified`: Boolean, Default=False
- email_verification_token: String(100)
- email_verification_sent_at: DateTime

Relationships:

- One-to-one with Patient
- One-to-one with Doctor
- One-to-many with Notification

2. Patient

Purpose: Extended profile for patient users **Table**: `patients` **Fields**:

- `id`: Integer, Primary Key
- `user_id`: Integer, Foreign Key to users.id, Not Null
- date_of_birth: Date
- `gender`: String(10)
- blood_type: String(5)
- medical_history: Text

Relationships:

- Many-to-one with User
- One-to-many with Appointment

3. Doctor

Purpose: Extended profile for healthcare providers **Table**: `doctors` **Fields**:

- `id`: Integer, Primary Key
- `user_id`: Integer, Foreign Key to users.id, Not Null
- `specialty`: String(100)
- `license_number`: String(50), Unique, Not Null
- years_of_experience: Integer
- education: Text
- `bio`: Text
- `verification_status`: Enum(VerificationStatus), Default=PENDING
- verification_notes: Text
- `location`: String(255)
- `languages`: String(255) (Comma-separated list)
- consultation_fee: Float
- `profile_picture`: String(255) (File path)
- `license_document_path`: String(255) (File path)
- `certificate_path`: String(255) (File path)

Relationships:

- Many-to-one with User
- One-to-many with DoctorAvailability
- One-to-many with Appointment
- One-to-many with VerificationDocument

4. DoctorAvailability

Purpose: Doctor's available consultation times **Table**: `doctor_availability` **Fields**:

- `id`: Integer, Primary Key
- `doctor_id`: Integer, Foreign Key to doctors.id, Not Null
- `day_of_week`: Integer, Not Null (0=Monday, 6=Sunday)
- `start_time`: Time, Not Null
- `end_time`: Time, Not Null
- `is_available`: Boolean, Default=True

Relationships:

- Many-to-one with Doctor

5. VerificationDocument

Purpose: Documents uploaded by doctors for verification **Table**: `verification_documents` **Fields**:

- `id`: Integer, Primary Key
- `doctor_id`: Integer, Foreign Key to doctors.id, Not Null
- `document_type`: String(50), Not Null
- `file_path`: String(255), Not Null
- `uploaded_at`: DateTime, Default=Current time

Relationships:

- Many-to-one with Doctor

6. Appointment

Purpose: Appointment booking between patients and doctors **Table**: `appointments` **Fields**:

- `id`: Integer, Primary Key
- `patient_id`: Integer, Foreign Key to patients.id, Not Null
- `doctor_id`: Integer, Foreign Key to doctors.id, Not Null
- appointment_date: Date, Not Null
- `start_time`: Time, Not Null
- `end_time`: Time, Not Null
- `status`: Enum(AppointmentStatus), Default=PENDING
- reason: Text

- notes: Text
- `created_at`: DateTime, Default=Current time
- `updated_at`: DateTime, Auto-updates

Relationships:

- Many-to-one with Patient
- Many-to-one with Doctor

7. Notification

Purpose: System notifications for users **Table**: `notifications` **Fields**:

- `id`: Integer, Primary Key
- `user_id`: Integer, Foreign Key to users.id, Not Null
- `title`: String(100), Not Null
- `message`: Text, Not Null
- `is_read`: Boolean, Default=False
- `created_at`: DateTime, Default=Current time

Relationships:

- Many-to-one with User

Entity Relationships

1. User-Patient/Doctor Relationship:
   - One-to-one relationship between User and Patient/Doctor
   - User serves as the authentication entity
   - Patient/Doctor contains role-specific information
2. Doctor-Availability Relationship:
   - One-to-many relationship
   - A doctor can have multiple availability slots
3. Doctor-Verification Relationship:
   - One-to-many relationship
   - A doctor can upload multiple verification documents
4. Patient-Doctor-Appointment Relationship:
   - Many-to-many relationship implemented through the Appointment table
   - A patient can book appointments with multiple doctors
   - A doctor can have appointments with multiple patients
5. User-Notification Relationship:
   - One-to-many relationship
   - A user can have multiple notifications

Data Integrity

The database schema ensures data integrity through:

- Foreign key constraints
- Unique constraints on critical fields (email, phone, license_number)
- Not-null constraints on required fields
- Default values for status fields
- Timestamps for tracking creation and updates
- Cascade delete operations to maintain referential integrity

Health Appointment System: UX/UI Template Files

Here's a comprehensive list of all template files that define the UX/UI of the Health Appointment System, organized by category:

Base Templates

1. `templates/base.html` - Main layout template with navigation and footer
2. `templates/admin/base.html` - Admin-specific base layout

Public Pages

1. `templates/index.html` - Homepage/landing page
2. templates/about.html - About page
3. templates/contact.html - Contact form page

Authentication Interfaces

1. templates/auth/login.html - User login
2. `templates/auth/register_choice.html` - Choose registration type (patient/doctor)
3. templates/auth/register_patient.html - Patient registration form
4. templates/auth/register_doctor.html - Doctor registration form
5. templates/auth/verify_email.html - Email verification
6. templates/auth/verify_phone.html - Phone verification
7. templates/auth/forgot_password.html - Password reset request
8. templates/auth/reset_password.html - New password form
9. templates/auth/pending_verification.html - Waiting for verification status

Patient Interfaces

1. templates/patient/dashboard.html - Patient main dashboard
2. templates/patient/appointments.html - Patient appointment management
3. templates/patient/cancel_appointment.html - Appointment cancellation form

Doctor Interfaces

1. templates/doctor/dashboard.html - Doctor main dashboard
2. templates/doctor/appointments.html - Doctor appointment management
3. `templates/doctor/availability.html` - Set working hours and availability
4. templates/doctor/profile.html - Doctor profile management

Admin Interfaces

1. templates/admin/dashboard.html - Admin main dashboard
2. templates/admin/login.html - Admin-specific login
3. templates/admin/users.html - User management list
4. templates/admin/user_detail.html - Detailed user information
5. templates/admin/create_user.html - Create new user form
6. templates/admin/edit_user.html - Edit user information
7. templates/admin/doctors.html - Doctor management list
8. templates/admin/patients.html - Patient management list
9. templates/admin/doctor_verification.html - Doctor verification interface

Shared Functionality

1. templates/main/find_doctors.html - Doctor search interface
2. templates/main/doctor_profile.html - Public doctor profile view
3. templates/main/book_appointment.html - Appointment booking interface
4. templates/main/notifications.html - User notifications center

These template files collectively define the entire user interface of the Health Appointment System. Each template extends either the main base template or the admin base template and includes specific content for its respective functionality.

The templates use Bootstrap 5 for responsive design and styling, with Font Awesome icons for visual elements. The forms are created using Flask-WTF and styled consistently across the application.

Health Appointment System - Detailed Interface Description

1. Common Interface Elements

Navigation Bar

Location: Present on all pages (defined in base.html) **Features:**

- Logo and brand name "Health Appointment"
- Responsive collapsible menu
- Links to: Home, About, Contact, Find Doctors (when logged in)
- User menu with:
  - Notifications icon with unread count badge
  - User profile dropdown with links to dashboard, profile, and logout
- Authentication links (when not logged in): Login, Register

Footer

Location: Present on all pages (defined in base.html) **Features:**

- Copyright information
- Links to Terms of Service and Privacy Policy
- Social media links

2. Public Pages

2.1. Homepage (index.html)

Purpose: Landing page for all users **Layout:**

- Hero section with:
  - Headline: "Healthcare at Your Fingertips"
  - Subheading: "Connect with doctors, schedule appointments, and manage your health journey all in one place"
  - CTA buttons: "Get Started" and "Learn More"
  - Hero image
- Features section with three cards:
  - Find Doctors: Icon, title, and description
  - Book Appointments: Icon, title, and description
  - Health Records: Icon, title, and description
- How It Works section with numbered steps:
  - Step 1: Register
  - Step 2: Find a Doctor
  - Step 3: Book an Appointment
  - Step 4: Receive Care
- Testimonials section with carousel of patient reviews

2.2. About Page (about.html)

Purpose: Information about the service **Features:**

- Mission statement
- Team information
- Service description
- Quality assurance information

2.3. Contact Page (contact.html)

Purpose: Allow users to contact administrators **Form Fields:**

- Name (text input)
- Email (email input)
- Subject (text input)
- Message (textarea)

- Submit button

3. Authentication Interfaces

3.1. Login Page (auth/login.html)

Purpose: User authentication **Form Fields:**

- Email (email input)
- Password (password input)
- Remember Me (checkbox)
- Submit button Additional Features:
- Forgot Password link
- Create Account link

3.2. Registration Choice (auth/register_choice.html)

Purpose: Allow users to select account type **Features:**

- Two cards with options:
  - Register as Patient: Icon, description, button
  - Register as Doctor: Icon, description, button

3.3. Patient Registration (auth/register_patient.html)

Purpose: Create patient account **Form Fields:**

- First Name (text input)
- Last Name (text input)
- Email (email input)
- Phone Number (text input)
- Password (password input)
- Confirm Password (password input)
- Date of Birth (date input)
- Gender (select: Male, Female, Other)
- Blood Type (select: A+, A-, B+, B-, AB+, AB-, O+, O-)
- Submit button

3.4. Doctor Registration (auth/register_doctor.html)

Purpose: Create doctor account **Form Fields:**

- First Name (text input)
- Last Name (text input)
- Email (email input)
- Phone Number (text input)
- Password (password input)
- Confirm Password (password input)
- Specialty (text input)
- License Number (text input)
- Years of Experience (number input)
- Location (text input)
- Languages (text input)
- Consultation Fee (number input)
- Profile Picture (file upload)
- License Document (file upload)
- Submit button

3.5. Email Verification (auth/verify_email.html)

Purpose: Verify user email **Features:**

- Verification code input
- Resend Code button
- Submit button

3.6. Forgot Password (auth/forgot_password.html)

Purpose: Reset forgotten password **Form Fields:**

- Email (email input)

- Submit button

3.7. Reset Password (auth/reset_password.html)
Purpose: Set new password after reset **Form Fields:**
- New Password (password input)
- Confirm Password (password input)
- Submit button

4. Patient Interfaces

4.1. Patient Dashboard (patient/dashboard.html)
Purpose: Main control panel for patients **Features:**
- Welcome header with patient name
- Statistics cards:
  - Upcoming Appointments (count with icon)
  - Medical Records (count with icon)
  - Prescriptions (count with icon)
  - Notifications (count with icon)
- Upcoming Appointments section:
  - Table with columns: Doctor, Date, Time, Status
  - Action buttons: View, Cancel
  - Empty state message if no appointments
- Profile Information card:
  - Personal details (name, email, phone)
  - Medical information (DOB, gender, blood type)
  - Edit Profile button

4.2. Patient Appointments (patient/appointments.html)
Purpose: View and manage appointments **Features:**
- Tabs for: Upcoming, Past, Cancelled
- Appointment cards with:
  - Doctor name and specialty
  - Date and time
  - Status indicator (confirmed, completed, cancelled)
  - Action buttons: View Details, Cancel, Reschedule

4.3. Cancel Appointment (patient/cancel_appointment.html)
Purpose: Cancel a booked appointment **Features:**
- Appointment details summary
- Cancellation reason (select or text input)
- Confirmation checkbox
- Cancel Appointment button
- Back button

5. Doctor Interfaces

5.1. Doctor Dashboard (doctor/dashboard.html)
Purpose: Main control panel for doctors **Features:**
- Welcome header with doctor name
- Statistics cards:
  - Today's Appointments (count with icon)
  - Upcoming Appointments (count with icon)
  - Notifications (count with icon)
  - Total Patients (count with icon)
- Quick Actions card:
  - Update Availability button
  - Edit Profile button
  - View All Appointments button
- Today's Schedule section:

        - Timeline view of appointments
        - Patient details for each slot
        - Empty state message if no appointments
    - Upcoming Appointments section:
        - List of next appointments
        - Date, time, and patient information
        - View All link
    - Profile Information card:
        - Professional details
        - Contact information
        - Verification status

5.2. Doctor Appointments (doctor/appointments.html)

Purpose: View and manage patient appointments **Features:**

- Calendar view with appointment slots
- List view option
- Filters for date range and status
- Appointment cards with:
    - Patient name and details
    - Date and time
    - Status indicator
    - Action buttons: Complete, Reschedule, Cancel

5.3. Doctor Availability (doctor/availability.html)

Purpose: Set working hours and unavailable dates **Features:**

- Weekly schedule with time slots
    - Checkboxes for each hour
    - Copy schedule to other days option
- Unavailable Dates section:
    - Date picker for blocking dates
    - List of currently blocked dates with delete option
- Save Changes button

5.4. Doctor Profile (doctor/profile.html)

Purpose: Edit professional profile **Form Fields:**

- Specialty (text input)
- License Number (text input)
- Years of Experience (number input)
- Location (text input)
- Languages (text input)
- Consultation Fee (number input)
- Bio/About (textarea)
- Education (textarea)
- Services Offered (textarea)
- Profile Picture (file upload)
- Save Changes button

6. Admin Interfaces

6.1. Admin Dashboard (admin/dashboard.html)

Purpose: Main control panel for administrators **Features:**

- Statistics cards:
    - Total Users (count with icon)
    - Patients (count with icon)
    - Doctors (count with icon)
    - Pending Verifications (count with icon)
- Recent Users section:
    - Table with newest registered users

        - User type indicators
        - Action buttons: View, Edit, Delete
        - View All link
    - Pending Doctor Verifications section:
        - List of doctors awaiting verification
        - License information
        - Action buttons: Verify, Reject, View Details
        - View All link

6.2. User Management (admin/users.html)

Purpose: View and manage all users **Features:**

- Search and filter options:
    - User type filter (All, Patients, Doctors, Admins)
    - Status filter (All, Active, Inactive)
    - Search by name or email
- Users table with columns:
    - ID
    - Name
    - Email
    - User Type
    - Status
    - Created Date
    - Actions (View, Edit, Delete)
- Pagination controls
- Create New User button

6.3. Doctor Verification (admin/doctor_verification.html)

Purpose: Verify doctor credentials **Features:**

- Doctor information:
    - Name and contact details
    - Specialty and experience
    - License information
- Document viewer for license document
- Verification form:
    - Status dropdown (Pending, Verified, Rejected)
    - Notes textarea
    - Submit button

6.4. Create User (admin/create_user.html)

Purpose: Create new user accounts **Form Fields:**

- User Type (select: Patient, Doctor, Admin)
- First Name (text input)
- Last Name (text input)
- Email (email input)
- Phone (text input)
- Password (password input)
- Confirm Password (password input)
- Status (select: Active, Inactive)
- Additional fields based on user type
- Create User button

7. Shared Interfaces

7.1. Find Doctors (main/find_doctors.html)

Purpose: Search for available doctors **Features:**

- Search filters:
    - Specialty (text input)
    - Location (text input)

        - Language (text input)
        - Submit button
    - Search tips card
    - Results section with doctor cards:
        - Profile picture
        - Name and specialty
        - Experience and location
        - Languages and consultation fee
        - Availability indicator
        - View Profile and Book Appointment buttons
    - No results message when applicable
    - Pagination controls

7.2. Doctor Profile (main/doctor_profile.html)

Purpose: View doctor information and book appointments **Features:**

- Profile sidebar:
    - Profile picture
    - Name and specialty
    - Experience badge
    - Location information
    - Languages spoken
    - Consultation fee
    - Book Appointment button
    - Back to Doctors link
- About section with tabs:
    - Professional Bio
    - Education & Training
    - Services Offered
- Reviews section:
    - Average rating
    - Individual reviews with ratings and comments
    - Date of review

7.3. Book Appointment (main/book_appointment.html)

Purpose: Schedule appointment with doctor **Features:**

- Doctor information sidebar:
    - Profile picture
    - Name and specialty
    - Consultation fee
    - Location
    - Back to Profile link
- Appointment booking form:
    - Date picker
    - Available time slots (displayed as selectable cards)
    - Appointment type (select: In-person, Video)
    - Reason for visit (textarea)
    - Book Appointment button
- Instructions and policies section

7.4. Notifications (main/notifications.html)

Purpose: View system notifications **Features:**

- Notification list:
    - Icon indicating type (appointment, system, etc.)
    - Message content
    - Timestamp
    - Read/unread indicator

        - Action button if applicable
    - Mark All as Read button
    - Filter options (All, Unread, Appointments, System)
    - Empty state message if no notifications

8. Design System

Colors

- Primary: Blue (#0d6efd) - Used for main actions, headers, and key UI elements
- Success: Green (#198754) - Used for positive actions and confirmations
- Danger: Red (#dc3545) - Used for destructive actions and errors
- Warning: Yellow (#ffc107) - Used for alerts and cautions
- Info: Light blue (#0dcaf0) - Used for informational elements
- Light: Off-white (#f8f9fa) - Used for backgrounds and subtle UI elements
- Dark: Dark gray (#212529) - Used for text and important UI elements

Typography

- Font family: System default sans-serif stack
- Headings: Bold weight with hierarchical sizing
- Body text: Regular weight with comfortable line height
- Special elements: Icons from Font Awesome library

Components

- Cards: Used extensively for grouping related information
- Buttons: Primary, outline, and text variants in different sizes
- Forms: Consistent styling with validation states
- Tables: Clean design with hover states and responsive behavior
- Alerts: Contextual feedback messages
- Badges: Small count and status indicators
- Modals: For confirmations and focused tasks
- Tabs: For organizing related content
- Pagination: For navigating multi-page content

Key Workflows

---

1. User Registration and Authentication

Patient Registration Flow:

1. User visits the homepage and clicks "Get Started"
2. User selects "Register as Patient" on the registration choice page
3. User completes the patient registration form with personal and basic health information
4. System sends a verification email to the provided address
5. User verifies their email by entering the code received
6. Patient account is activated and user is redirected to their dashboard

Doctor Registration Flow:

1. User selects "Register as Doctor" on the registration choice page
2. User completes the doctor registration form with professional credentials
3. User uploads required verification documents (license, certificates)
4. System sends a verification email to the provided address
5. User verifies their email by entering the code received
6. Doctor account is created in "Pending Verification" status
7. Administrator reviews the doctor's credentials

8. Upon approval, doctor account is activated and doctor is notified

Authentication Flow:

1. User visits the login page
2. User enters email and password
3. System validates credentials and determines user type
4. User is redirected to the appropriate dashboard based on their role
5.

---

2. Doctor Discovery and Appointment Booking

Doctor Search Flow:

1. Patient logs in and navigates to "Find Doctors"
2. Patient filters doctors by specialty, location, and/or language
3. System displays matching doctors with their basic information
4. Patient can view detailed doctor profiles by clicking on a doctor card

Appointment Booking Flow:

1. Patient views a doctor's profile
2. Patient clicks "Book Appointment" button
3. System displays the doctor's availability calendar
4. Patient selects a date from the calendar
5. System shows available time slots for the selected date
6. Patient selects a time slot and appointment type (in-person/video)
7. Patient enters reason for visit
8. Patient confirms the appointment
9. System creates the appointment and sends notifications to both patient and doctor
10. Appointment appears in both patient's and doctor's dashboards

---

3. Appointment Management

Patient Appointment Management Flow:

1. Patient views upcoming appointments on their dashboard
2. Patient can view appointment details, cancel, or reschedule
3. For cancellation, patient provides a reason and confirms
4. System updates appointment status and notifies the doctor
5. For past appointments, patient can view history and any attached notes

Doctor Appointment Management Flow:

1. Doctor views today's and upcoming appointments on dashboard
2. Doctor can view appointment details, mark as completed, or cancel
3. Doctor can add notes after completing an appointment
4. Doctor can manage their availability by blocking dates or setting regular hours
5. System ensures new appointments can only be booked during available slots

---

4. Doctor Verification

Verification Flow:

1. Administrator logs in and sees pending verifications on dashboard
2. Administrator reviews doctor information and uploaded documents

3. Administrator can approve or reject the verification
4. If approved, doctor status changes to "Verified" and they can begin accepting appointments
5. If rejected, administrator provides a reason and doctor is notified
6. Doctor can resubmit verification with corrected information if rejected

---

5. User Management

Administrator User Management Flow:

1. Administrator can view all users from the admin dashboard
2. Administrator can create new users of any type
3. Administrator can edit user information or deactivate accounts
4. Administrator can reset user passwords if needed
5. Administrator can view system logs and activity

Data Flow

1. **User Data**: Collected during registration, stored securely with passwords hashed using bcrypt
2. **Appointment Data**: Created when bookings are made, linked to both patient and doctor
3. **Notification Data**: Generated by system events, stored until marked as read
4. **Verification Data**: Uploaded by doctors, reviewed by administrators, stored securely

Technical Workflow

1. **Request Handling**: Flask routes receive HTTP requests and direct them to appropriate handlers
2. **Authentication**: Flask-Login manages user sessions and access control
3. **Database Operations**: SQLAlchemy handles data persistence and retrieval
4. **Form Processing**: Flask-WTF validates and processes form submissions
5. **Email Notifications**: Flask-Mail sends transactional emails for important events
6. **File Uploads**: Managed securely for profile pictures and verification documents

Security Considerations

1. **Authentication**: Secure login with password hashing and session management
2. **Authorization**: Role-based access control for different user types
3. **Data Protection**: Input validation and sanitization to prevent injection attacks
4. **Session Security**: CSRF protection on all forms
5. **Document Security**: Verification documents stored securely with restricted access

Technologies Used

Backend Framework

- **Flask** (v2.2.5): Python web framework for building the application
- **Werkzeug** (v2.2.3): WSGI utility library for Flask

Database and ORM

- **SQLAlchemy** (v2.0.23): SQL toolkit and Object-Relational Mapping (ORM)
- **Flask-SQLAlchemy** (v3.0.5): Flask extension for SQLAlchemy
- **Flask-Migrate** (v4.0.5): Database migration handling

Database System

- **SQLite** (default development database): File-based database stored in `instance/health_app.db`
- **PostgreSQL** (production configuration available): Via psycopg2-binary (v2.9.9)

Authentication and Security

- **Flask-Login** (v0.6.2): User session management
- **Flask-Bcrypt** (v1.0.1): Password hashing
- **Flask-WTF** (v1.1.1): Form handling and CSRF protection
- **WTForms** (v3.0.1): Form validation and rendering

Email and Communication

- **Flask-Mail** (v0.9.1): Email sending capabilities
- **email-validator** (v2.0.0): Email validation

Image Processing

- **Pillow** (v10.0.0): Python Imaging Library for handling profile pictures

Utilities

- **python-dotenv** (v1.0.0): Environment variable management
- **phonenumbers** (v8.13.23): Phone number parsing and validation

Deployment

- **Gunicorn** (v21.2.0): WSGI HTTP server for production deployment

Frontend

- **Bootstrap 5**: CSS framework (loaded via CDN)
- **Font Awesome 6**: Icon library (loaded via CDN)
- **jQuery**: JavaScript library (loaded via CDN)
- **HTML5/CSS3/JavaScript**: Standard web technologies

What Should Be Kept in Windsurf Memory

For this Flask-based health appointment system, the following information should be kept in memory to assist with future interactions:

I'll provide a comprehensive list of technologies and database details used in the Health Appointment System, focusing on what should be kept in memory for future reference.

Technologies Used in the Health Appointment System

Backend Framework

- **Flask** (v2.2.5): Main Python web framework
- **Werkzeug** (v2.2.3): WSGI utility library for Flask

Database and ORM

- **SQLAlchemy** (v2.0.23): SQL toolkit and Object-Relational Mapping
- **Flask-SQLAlchemy** (v3.0.5): Flask extension for SQLAlchemy integration
- **Flask-Migrate** (v4.0.5): Database migration management

Database Systems

- **SQLite**: Default development database (stored in `instance/health_app.db`)
- **PostgreSQL**: Production database configuration (via psycopg2-binary v2.9.9)

Authentication and Security

- **Flask-Login** (v0.6.2): User session management
- **Flask-Bcrypt** (v1.0.1): Password hashing and security
- **Flask-WTF** (v1.1.1): Form handling with CSRF protection
- **WTForms** (v3.0.1): Form validation and rendering

Email and Communication

- **Flask-Mail** (v0.9.1): Email functionality
- **email-validator** (v2.0.0): Email validation library

Media Handling

- **Pillow** (v10.0.0): Image processing for profile pictures and documents

Utilities

- **python-dotenv** (v1.0.0): Environment variable management
- **phonenumbers** (v8.13.23): Phone number validation and formatting

Deployment

- **Gunicorn** (v21.2.0): WSGI HTTP server for production

Frontend

- **Bootstrap 5**: CSS framework (via CDN)
- **Font Awesome 6**: Icon library (via CDN)
- **jQuery**: JavaScript library for interactive elements
- **HTML5/CSS3/JavaScript**: Standard web technologies

Database Structure

The application uses a relational database with the following key models:

1. **User**: Base user model with authentication details
    - Fields: id, email, password, first_name, last_name, phone, user_type, is_active, created_at
2. **Patient**: Extended user profile for patients
    - Fields: id, user_id, date_of_birth, gender, blood_type, medical_history
3. **Doctor**: Extended user profile for healthcare providers
    - Fields: id, user_id, specialty, license_number, years_of_experience, location, languages, consultation_fee, profile_picture, bio, education, services
4. **Appointment**: Booking records between patients and doctors
    - Fields: id, patient_id, doctor_id, date, time, status, appointment_type, reason, notes
5. **VerificationDocument**: Documents uploaded by doctors for verification
    - Fields: id, doctor_id, document_type, file_path, uploaded_at, verified
6. **Notification**: System notifications for users
    - Fields: id, user_id, message, created_at, is_read, notification_type

What Should Be Kept in Memory

For future interactions with this project, the following information should be maintained in memory:

1. Application Architecture:
    - Flask-based web application with blueprint structure
    - Three-tier user role system (patients, doctors, administrators)
    - RESTful routing patterns
2. Technology Stack:
    - Flask as the primary framework
    - SQLAlchemy for database operations
    - Bootstrap 5 for frontend styling
3. Database Configuration:
    - SQLite for development
    - PostgreSQL for production
    - Migration management with Flask-Migrate
4. Key Workflows:
    - User authentication flows (different for each role)
    - Appointment booking and management process
    - Doctor verification workflow
5. Security Implementations:
    - Password hashing with bcrypt
    - Form validation and CSRF protection
    - Role-based access control
6. Deployment Configuration:
    - Gunicorn as WSGI server
    - Environment variable management with python-dotenv