



Dental Clinic Patient Information System

Front-End Implementation Plan

Based on PRD Document Analysis

Version: 1.0

Date: January 2025








Table of Contents

1. Current Status
 2. Features to Implement
 3. Implementation Roadmap
 4. Detailed Feature Breakdown
 5. File Structure
 6. Design Guidelines
-







Current Status

Working Features:

-  Login System (Manager & Staff)
-  Dashboard (Role-based views)
-  Logout functionality
-  Session management
-  Role-based UI control







Files Cleaned Up:

-  Deleted: `patients.html`, `patient-detail.html`
 -  Deleted: `patients.css`, `patient-detail.css`
 -  Deleted: `patients.js`, `patient-detail.js`
 -  Deleted: `staff.css` (redundant)
-






Features to Implement

Manager Features (6 modules):

1. **Dashboard**  (Working)
2. **Patients**  (To implement)
3. **Staff Management**  (To implement)
4. **Audit Logs**  (To implement)
5. **Backup & Restore**  (To implement)
6. **Account Settings**  (To implement)

Staff Features (2 modules):

1. **Dashboard**  (Working)
 2. **Patients**  (To implement)
 3. **Account Settings**  (To implement)
-



Implementation Roadmap

Phase 1: Patient Management (Priority: HIGH)

Duration: 3-4 days

Files to create:

- `templates/patients.html`
- `static/css/patients.css`
- `static/js/patients.js`

Features:

- Patient list table with search/filters
 - Add new patient (modal)
 - Edit patient (modal)
 - View patient details (modal)
 - Delete patient (Manager only)
 - Export to CSV (Manager only)
 - Bulk delete (Manager only)
-

Phase 2: Staff Management (Priority: MEDIUM - Manager Only)

Duration: 2-3 days

Files to create:

- `templates/pages/staff-management.html` (replace coming-soon)
- `static/css/staff.css`
- `static/js/staff.js`

Features:

- Staff list table
 - Add new staff account
 - Edit staff account
 - Delete staff account
 - Reset staff password
 - View staff activity
-

Phase 3: Audit Logs (Priority: MEDIUM)

Duration: 2 days

Files to create:

- `templates/pages/audit-logs.html` (replace coming-soon)
- `static/css/audit-logs.css`
- `static/js/audit-logs.js`

Features:

- Activity log table
 - Filter by user/action/date
 - Export logs to CSV
 - Search functionality
 - Manager sees all logs
 - Staff sees only their own logs
-

Phase 4: Backup & Restore (Priority: MEDIUM - Manager Only)

Duration: 2 days

Files to create:

- `templates/pages/backup-restore.html` (replace coming-soon)
- `static/css/backup.css`
- `static/js/backup.js`

Features:

- Create backup button
 - Download backup file
 - Upload backup file
 - Restore from backup
 - Backup history list
 - Auto-backup settings
-

Phase 5: Account Settings (Priority: LOW)

Duration: 1-2 days

Files to create:

- `templates/pages/settings.html` (replace coming-soon)
- `static/css/settings.css`
- `static/js/settings.js`

Features:

- Change password
 - Update profile info
 - Notification preferences
 - Theme settings (optional)
-

Detailed Feature Breakdown

1. PATIENTS MODULE (Both Manager & Staff)

1.1 Patient List Page




File: `templates/patients.html`

Layout:



Features:

- **Table Columns:**
 - Patient ID
 - Full Name
 - Gender
 - Age (calculated from DOB)
 - Contact Number
 - Last Visit Date
 - Status (Active/Inactive)

- Actions (View , Edit , Delete )
- **Search & Filters:**
 - Search by name or ID
 - Filter by gender (Male/Female/Other)
 - Filter by status (Active/Inactive)
 - Reset filters button
- **Action Buttons:**
 - "Add New Patient" (opens modal)
 - "Export to CSV" (Manager only)
 - "Delete Selected" (Manager only with checkboxes)

1.2 Add Patient Modal

Modal Form Fields:

Basic Information:

- First Name* (required)
- Last Name* (required)
- Gender* (dropdown: Male/Female/Other)
- Date of Birth* (date picker)
- Contact Number* (required)
- Address

Medical Information (Manager only):

- Allergies
- Medical Conditions
- Current Medications
- Dentist Notes

Additional:

- Assigned Dentist (Manager only)
- Status (Active/Inactive - Manager only)

Buttons:

- Cancel
- Save (Manager) / Submit for Review (Staff)

1.3 Edit Patient Modal

Same as Add Patient Modal but pre-filled with existing data.

1.4 View Patient Modal

Read-only view of all patient information with tabs:

- **Personal Info Tab**
- **Medical History Tab** (Manager only)
- **Visit History Tab**

Buttons:

- Close
- Edit (redirects to Edit Modal)

1.5 Delete Confirmation

Manager:

- Simple confirmation dialog
- "Are you sure you want to delete [Patient Name]?"
- Cancel / Delete buttons

Staff:

- Request deletion (creates a pending request for Manager)
- Manager can approve/reject in Audit Logs




2. STAFF MANAGEMENT MODULE (Manager Only)

2.1 Staff List Page

File: `templates/pages/staff-management.html`

Features:

- Staff table with columns:
 - Staff ID
 - Full Name

- Username
- Role (Manager/Staff)
- Status (Active/Inactive)
- Last Login
- Actions (Edit , Reset Password , Delete )
- Action buttons:
 - "Add New Staff" (opens modal)
 - "Export Staff List"

2.2 Add Staff Modal

Form Fields:

- Full Name*
- Username* (unique)
- Password* (with strength indicator)
- Confirm Password*
- Role* (dropdown: Manager/Staff)
- Email
- Phone Number
- Status (Active/Inactive)

2.3 Edit Staff Modal

Same as Add Staff but:

- Cannot edit username
- Password field optional (blank = no change)

2.4 Reset Password Modal

Simple modal with:

- New Password*
 - Confirm Password*
 - "Generate Random Password" button
-

3. AUDIT LOGS MODULE

3.1 Activity Logs Page

File: `templates/pages/audit-logs.html`

Features:

- Activity log table:
 - Timestamp
 - User (Name + Role)
 - Action Type (Create/Update/Delete/View/Login/Logout)
 - Target (e.g., "Patient: John Doe")
 - Details
 - IP Address (optional)
- **Filters:**
 - Date range picker
 - User filter (Manager sees all, Staff sees only their own)
 - Action type filter
 - Search by keyword
- **Export:**
 - "Export Logs to CSV" button

Action Badges:

- Create (Green)
 - Update (Blue)
 - Delete (Red)
 - View (Gray)
 - Login (Purple)
 - Logout (Orange)
-

4. BACKUP & RESTORE MODULE (Manager Only)

4.1 Backup & Restore Page

File: templates/pages/backup-restore.html

Layout:

Backup Section

 Create Backup Now

Last Backup: 2025-01-15 10:30 AM

Backup History

| # | Date | Size | Actions |
|---|---------|------|--|
| 1 | 1/15/25 | 2MB |  Download |

Restore Section

 Upload Backup File

 Restore from Backup

Features:

- Create backup (downloads .db file)
- Backup history table
- Download previous backups
- Upload backup file for restore
- Restore confirmation dialog
- Auto-backup settings (enable/disable)

5. ACCOUNT SETTINGS MODULE (Both Roles)

5.1 Settings Page

File: templates/pages/settings.html

Sections:

Profile Settings:

- Full Name
- Email
- Phone Number
- Profile Picture (optional for Phase 2)

Security Settings:

- Change Password
 - Current Password*
 - New Password*
 - Confirm New Password*

Preferences:

- Notifications (Enable/Disable)
- Date Format (MM/DD/YYYY or DD/MM/YYYY)
- Time Zone

System Info (Read-only):

- Account Created Date
- Last Login
- Total Logins
- Account Status

File Structure (After Implementation)

```
dental-clinic-management-system/  
├── index.html  
├── templates/  
│   ├── login.html ✓  
│   ├── dashboard.html ✓  
│   ├── patients.html 🔑  
│   └── pages/  
│       ├── staff-management.html 🔑  
│       ├── audit-logs.html 🔑  
│       ├── backup-restore.html 🔑  
│       └── settings.html 🔑  
└── static/
```

```
| |— css/
| |   |— style.css ✓ (global)
| |   |— login.css ✓
| |   |— dashboard.css ✓
| |   |— patients.css 🔧
| |   |— staff.css 🔧
| |   |— audit-logs.css 🔧
| |   |— backup.css 🔧
| |   |— settings.css 🔧
| |— js/
| |   |— auth.js ✓
| |   |— main.js ✓
| |   |— login.js ✓
| |   |— dashboard.js ✓
| |   |— patients.js 🔧
| |   |— staff.js 🔧
| |   |— audit-logs.js 🔧
| |   |— backup.js 🔧
| |   |— settings.js 🔧
|— docs/
|   |— Dental Clinic Patient Information System.docx
|— README.md
```

Legend:

✓ Complete

🔧 To be implemented

Design Guidelines

Color Palette:

CSS

```
/* Primary Pink Theme */  
--primary-pink: #E8B4BC;  
--primary-pink-dark: #D89BA7;  
--primary-pink-light: #F3C4D0;
```

```
/* Neutrals */  
--text-dark: #1A1A1A;  
--text-muted: #64748B;  
--text-light: #94A3B8;  
--bg-gray: #F7F8FA;  
--bg-white: #FFFFFF;
```

```
/* Borders */  
--border-light: #E2E8F0;  
--border-lighter: #F1F5F9;
```

```
/* Status Colors */  
--success: #10B981;  
--warning: #F59E0B;  
--error: #EF4444;  
--info: #3B82F6;
```

Typography:

- **Headings:** Poppins (600/700)
- **Body:** Inter (400/500/600)
- **Base Size:** 14px
- **Heading Sizes:** H1: 28px, H2: 22px, H3: 18px

Spacing:

- Card Padding: 24px
- Section Margin: 32px
- Button Padding: 12px 24px
- Input Padding: 12px 14px

Components:

- **Border Radius:** 10-14px
- **Shadows:** 0 2px 8px rgba(0,0,0,0.06)

- **Transitions:** 0.2s ease
 - **Button Hover:** translateY(-2px)
-

Technical Implementation Notes

Data Management (Front-End Only):

Since this is **front-end only**, we'll use:

1. **LocalStorage** for demo data:

```
javascript

// Example structure
const demoPatients = [
  {
    id: 'PT-0001',
    firstName: 'John',
    lastName: 'Doe',
    gender: 'Male',
    dob: '1990-05-15',
    contact: '09171234567',
    address: 'Quezon City',
    status: 'active',
    lastVisit: '2025-01-10',
    medicalHistory: {
      allergies: 'Penicillin',
      conditions: 'Hypertension',
      medications: 'Losartan 50mg',
      notes: 'Regular checkups needed'
    }
  }
];

localStorage.setItem('clinic_patients', JSON.stringify(demoPatients));
```

2. **SessionStorage** for current user session:

```
javascript
```

```
sessionStorage.setItem('clinic_user', JSON.stringify({
  username: 'manager',
  role: 'manager',
  name: 'Dr. Maria Santos'
}));
```

3. Demo Data Generation:

- Pre-populate with 10-15 sample patients
- 3-4 sample staff accounts
- Mock activity logs

Form Validation:

```
javascript

// Patient form validation
function validatePatientForm(data) {
  const errors = {};

  if (!data.firstName) errors.firstName = 'First name is required';
  if (!data.lastName) errors.lastName = 'Last name is required';
  if (!data.contact) errors.contact = 'Contact is required';
  if (data.contact && !/^09\d{9}$/.test(data.contact)) {
    errors.contact = 'Invalid Philippine phone number';
  }

  return {
    isValid: Object.keys(errors).length === 0,
    errors
  };
}
```

Export to CSV:

```
javascript
```

```
function exportPatientsToCSV(patients) {  
  const headers = ['ID', 'Name', 'Gender', 'DOB', 'Contact', 'Status'];  
  const rows = patients.map(p => [  
    p.id,  
    `${p.firstName} ${p.lastName}`,  
    p.gender,  
    p.dob,  
    p.contact,  
    p.status  
  ]);  
  
  const csv = [headers, ...rows]  
    .map(row => row.map(cell => `"${cell}"`).join(','))  
    .join('\n');  
  
  const blob = new Blob([csv], { type: 'text/csv' });  
  const url = URL.createObjectURL(blob);  
  const a = document.createElement('a');  
  a.href = url;  
  a.download = `patients_${new Date().toISOString().split('T')[0]}.csv`;  
  a.click();  
}
```

Backup & Restore:

javascript


```
// Backup all data
function createBackup() {
  const backup = {
    version: '1.0',
    timestamp: new Date().toISOString(),
    data: {
      patients: JSON.parse(localStorage.getItem('clinic_patients') || '[]'),
      staff: JSON.parse(localStorage.getItem('clinic_staff') || '[]'),
      logs: JSON.parse(localStorage.getItem('clinic_logs') || '[]')
    }
  };

  const json = JSON.stringify(backup, null, 2);
  const blob = new Blob([json], { type: 'application/json' });
  const url = URL.createObjectURL(blob);
  const a = document.createElement('a');
  a.href = url;
  a.download = `clinic_backup_${backup.timestamp.split('T')[0]}.json`;
  a.click();
}
```

```
// Restore from backup
function restoreBackup(file) {
  const reader = new FileReader();
  reader.onload = (e) => {
    const backup = JSON.parse(e.target.result);
    localStorage.setItem('clinic_patients', JSON.stringify(backup.data.patients));
    localStorage.setItem('clinic_staff', JSON.stringify(backup.data.staff));
    localStorage.setItem('clinic_logs', JSON.stringify(backup.data.logs));
    alert('Backup restored successfully!');
    location.reload();
  };
  reader.readAsText(file);
}
```

Sample Data Structure

Patient Object:

javascript

```
{
  id: 'PT-0001',
  firstName: 'John',
  lastName: 'Doe',
  gender: 'Male',
  dob: '1990-05-15',
  age: 34, // calculated
  contact: '09171234567',
  email: 'john.doe@email.com',
  address: 'Quezon City, Metro Manila',
  status: 'active', // active | inactive
  lastVisit: '2025-01-10',
  assignedDentist: 'Dr. Maria Santos',
  medicalHistory: {
    allergies: 'Penicillin',
    conditions: 'Hypertension',
    medications: 'Losartan 50mg',
    notes: 'Regular checkups needed'
  },
  visits: [
    {
      date: '2025-01-10',
      provider: 'Dr. Maria Santos',
      service: 'Teeth Cleaning',
      notes: 'Routine cleaning, no issues'
    }
  ],
  createdAt: '2024-06-15T10:30:00Z',
  createdBy: 'manager',
  updatedAt: '2025-01-10T14:20:00Z',
  updatedBy: 'staff'
}
```

Staff Object:

javascript

```
{
  id: 'ST-0001',
  username: 'staff',
  name: 'Carla Reyes',
  role: 'staff', // manager | staff
  email: 'carla@clinic.com',
  phone: '09171234567',
  status: 'active',
  createdAt: '2024-01-01T08:00:00Z',
  lastLogin: '2025-01-15T09:15:00Z',
  totalLogins: 145
}
```

Activity Log Object:

javascript

```
{
  id: 'LOG-0001',
  timestamp: '2025-01-15T10:30:00Z',
  user: 'staff',
  userName: 'Carla Reyes',
  action: 'CREATE', // CREATE | UPDATE | DELETE | VIEW | LOGIN | LOGOUT
  target: 'Patient',
  targetId: 'PT-0001',
  targetName: 'John Doe',
  details: 'Created new patient record',
  ipAddress: '192.168.1.100' // optional
}
```

✓ Implementation Checklist

Phase 1: Patients Module

- ☐ Create `patients.html` structure
- ☐ Create `patients.css` styling
- ☐ Create `patients.js` logic
- ☐ Implement patient list table
- ☐ Implement search & filters
- ☐ Implement Add Patient modal
- ☐ Implement Edit Patient modal

- ☐ Implement View Patient modal
- ☐ Implement Delete Patient (Manager)
- ☐ Implement Export to CSV (Manager)
- ☐ Add demo data (10-15 patients)
- ☐ Test all CRUD operations
- ☐ Test role-based access control

Phase 2: Staff Management

- ☐ Update `staff-management.html`
- ☐ Create `staff.css`
- ☐ Create `staff.js`
- ☐ Implement staff list table
- ☐ Implement Add Staff modal
- ☐ Implement Edit Staff modal
- ☐ Implement Reset Password
- ☐ Implement Delete Staff
- ☐ Add demo data (3-4 staff)
- ☐ Test all operations

Phase 3: Audit Logs

- ☐ Update `audit-logs.html`
- ☐ Create `audit-logs.css`
- ☐ Create `audit-logs.js`
- ☐ Implement activity log table
- ☐ Implement filters
- ☐ Implement export to CSV
- ☐ Add auto-logging to all actions
- ☐ Generate demo logs
- ☐ Test filtering

Phase 4: Backup & Restore

- ☐ Update `backup-restore.html`
- ☐ Create `backup.css`
- ☐ Create `backup.js`
- ☐ Implement backup creation
- ☐ Implement backup download
- ☐ Implement restore from file
- ☐ Implement backup history

☐ Test backup/restore flow

Phase 5: Account Settings

☐ Update `settings.html`

☐ Create `settings.css`

☐ Create `settings.js`

☐ Implement profile settings

☐ Implement change password

☐ Implement preferences

☐ Test all settings

Next Steps

1. **Review this plan** - Make sure all features align with your vision
2. **Choose starting phase** - I recommend Phase 1 (Patients Module)
3. **Create first feature** - Let's build the Patients page together!

Would you like me to start implementing **Phase 1: Patients Module** now? 