

KAWIT RHU PATIENT-SIDE MODULE DOCUMENTATION

Version: 1.0.0

Date: October 16, 2025

Purpose: Complete technical documentation for patient-side modules to guide admin-side development

TABLE OF CONTENTS

- [System Architecture](#)
- [Database Schema Reference](#)
- [Authentication & Security Patterns](#)
- [Module 1: Dashboard](#)
- [Module 2: Appointments](#)
- [Module 3: Online Consultation](#)
- [Module 4: Health Records](#)
- [Module 5: Medical Certificates](#)
- [Module 6: Profile Management](#)
- [Common UI/UX Patterns](#)
- [JavaScript Patterns](#)
- [Admin Development Guidelines](#)

SYSTEM ARCHITECTURE

File Structure

```
Kawit RHU/
├── Patient/
│   ├── Dashboard.php           # Patient dashboard
│   ├── Appointment.php        # BHS appointment booking
│   ├── Consultation.php       # Online consultation requests
│   ├── Health_Record.php      # Complete health records view
│   ├── Medical_Certificates.php # Certificate requests
│   └── Profile.php            # Profile & settings management
├── Admin/
│   ├── Rural_Health_Unit/     # RHU admin modules (TO BUILD)
│   ├── Barangay_Health_Station/ # BHS admin modules (TO BUILD)
│   └── Pharmacy_Admin/        # Pharmacy modules (TO BUILD)
├── Pictures/                  # Logo images
├── login.php
└── logout.php
```

Technology Stack

- Backend:** PHP 7.4+ with PDO
- Database:** MySQL 5.7+
- Frontend:** Bootstrap 5.3.0, FontAwesome 6.4.0
- JavaScript:** Vanilla JS with Fetch API
- Session Management:** PHP Sessions

DATABASE SCHEMA REFERENCE

Core Tables Used Across All Modules

users - Authentication

- id (PK)
- username (UNIQUE)
- password (hashed)
- email (UNIQUE)
- role ENUM('super_admin', 'rhu_admin', 'bhs_admin', 'pharmacy_admin', 'patient')
- is_active BOOLEAN
- login_attempts INT
- locked_until DATETIME
- last_login DATETIME
- password_changed_at TIMESTAMP

patients - Patient Profiles

- id (PK)
- user_id (FK → users.id, UNIQUE)
- patient_id VARCHAR(20) UNIQUE (e.g., P-2024-0001)
- first_name, middle_name, last_name, suffix
- date_of_birth DATE
- gender ENUM('Male', 'Female', 'Other')
- civil_status ENUM('Single', 'Married', 'Widowed', 'Divorced', 'Separated')
- address TEXT
- barangay_id (FK → barangays.id)
- phone VARCHAR(15)
- email VARCHAR(100)
- blood_type VARCHAR(5)
- allergies TEXT
- medical_history TEXT
- emergency_contact_name, emergency_contact_phone, emergency_contact_relationship
- philhealth_number VARCHAR(20)
- occupation, educational_attainment, religion
- is_active BOOLEAN

appointments - Appointment System

- id (PK)
- patient_id (FK → patients.id)
- service_type_id (FK → service_types.id)
- appointment_date DATE
- appointment_time TIME (00:00:00 = admin will schedule)
- appointment_location ENUM('RHU', 'BHS')
- barangay_id (FK → barangays.id, required for BHS)
- status ENUM('pending', 'confirmed', 'completed', 'cancelled', 'no-show', 'rescheduled')
- notes TEXT
- reason_for_visit TEXT
- assigned_staff (FK → staff.id)
- created_by (FK → users.id)
- confirmed_by (FK → staff.id)
- confirmed_at DATETIME
- cancelled_reason TEXT

consultations - Medical Consultations

- id (PK)
- consultation_number VARCHAR(20) UNIQUE (e.g., CONS-2024-0001)
- patient_id (FK → patients.id)
- appointment_id (FK → appointments.id, nullable)
- consultation_type ENUM('online', 'walk-in', 'follow-up', 'emergency')
- consultation_date DATETIME
- chief_complaint TEXT
- history_of_present_illness TEXT
- symptoms TEXT
- vital_signs JSON
- physical_examination TEXT
- assessment TEXT
- diagnosis TEXT
- treatment_plan TEXT
- medications_prescribed TEXT
- recommendations TEXT
- follow_up_instructions TEXT
- priority ENUM('low', 'medium', 'high', 'urgent')
- status ENUM('pending', 'in_progress', 'completed', 'cancelled')
- assigned_doctor (FK → staff.id)
- consultation_location ENUM('RHU', 'BHS')
- google_meet_link VARCHAR(500) -- Added for online consultations
- barangay_id (FK → barangays.id)
- follow_up_date DATE

prescriptions - Medication Prescriptions

- id (PK)
- prescription_number VARCHAR(20) UNIQUE (e.g., RX-2024-0001)
- patient_id (FK → patients.id)
- consultation_id (FK → consultations.id, nullable)
- medication_name VARCHAR(100)
- generic_name VARCHAR(100)
- brand_name VARCHAR(100)
- dosage_strength VARCHAR(50)
- dosage_form VARCHAR(50)
- quantity_prescribed INT
- quantity_dispensed INT
- dosage_instructions TEXT
- frequency VARCHAR(50)
- duration VARCHAR(50)
- special_instructions TEXT
- prescribed_by (FK → staff.id)
- prescription_date DATE
- dispensed_by (FK → staff.id)
- dispensed_date DATETIME
- status ENUM('pending', 'partially_dispensed', 'fully_dispensed', 'cancelled', 'expired')

laboratory_results - Lab Tests

- id (PK)
- lab_number VARCHAR(20) UNIQUE (e.g., LAB-2024-0001)
- patient_id (FK → patients.id)
- consultation_id (FK → consultations.id, nullable)
- test_type VARCHAR(100)
- test_category ENUM('Hematology', 'Chemistry', 'Urinalysis', 'Microbiology', 'Serology', 'Parasitology', 'Other')
- test_date DATE
- result_date DATE
- specimen_type VARCHAR(50)
- test_results JSON
- normal_range TEXT
- interpretation TEXT
- remarks TEXT
- status ENUM('pending', 'processing', 'completed', 'cancelled')
- performed_by, verified_by, released_by (FK → staff.id)

medical_certificates - Certificate Management

- id (PK)
- certificate_number VARCHAR(50) UNIQUE (e.g., CERT-2024-0001)
- patient_id (FK → patients.id)
- consultation_id (FK → consultations.id, nullable)
- issued_by (FK → staff.id)
- certificate_type ENUM('Fit to Work', 'Medical Certificate', 'Health Certificate', 'Vaccination Certificate', 'Other')
- purpose VARCHAR(200)
- date_issued DATE
- valid_from, valid_until DATE
- diagnosis TEXT
- physical_findings TEXT
- recommendations TEXT
- restrictions TEXT
- fitness_status ENUM('Fit', 'Unfit', 'Conditional', 'Pending Further Evaluation')
- status ENUM('draft', 'issued', 'expired', 'revoked')

notifications - User Notifications

- id (PK)
- user_id (FK → users.id)
- type ENUM('appointment_reminder', 'lab_result', 'prescription_ready', 'announcement', 'system', 'referral_update', 'consultation_r
- title VARCHAR(200)
- message TEXT
- data JSON
- is_read BOOLEAN
- priority ENUM('low', 'medium', 'high')
- expires_at DATETIME
- read_at TIMESTAMP

system_logs - Audit Trail

- id (PK)
- user_id (FK → users.id)
- log_level ENUM('INFO', 'WARNING', 'ERROR', 'CRITICAL')
- action VARCHAR(100) -- e.g., 'APPOINTMENT_BOOKED', 'PROFILE_UPDATE'
- module VARCHAR(50)
- table_affected VARCHAR(50)
- record_id INT
- old_values JSON
- new_values JSON
- ip_address VARCHAR(45)
- session_id VARCHAR(255)
- created_at TIMESTAMP

🔒 AUTHENTICATION & SECURITY PATTERNS

Standard Authentication Check (Used in ALL Files)

```
<?php
session_start();

// Authentication check
if (!isset($_SESSION['user_id']) || $_SESSION['role'] !== 'patient') {
    header('Location: ../login.php');
    exit;
}

// Database connection
$host = 'localhost';
$dbname = 'kawit_rhu';
$username = 'root';
$password = '';

try {
    $pdo = new PDO("mysql:host=$host;dbname=$dbname", $username, $password);
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
} catch(PDOException $e) {
    die("Connection failed: " . $e->getMessage());
}

// Get patient information (standard across all modules)
$stmt = $pdo->prepare("
    SELECT p.*, u.username, b.barangay_name,
           YEAR(CURDATE()) - YEAR(p.date_of_birth) as age
    FROM patients p
    JOIN users u ON p.user_id = u.id
    LEFT JOIN barangays b ON p.barangay_id = b.id
    WHERE u.id = ?
");
$stmt->execute([$_SESSION['user_id']]);
$patient = $stmt->fetch(PDO::FETCH_ASSOC);

if (!$patient) {
    header('Location: ../login.php');
    exit;
}
?>
```

Security Best Practices Implemented

1. Session Management

- Session validation on every page load
- Role-based access control
- Auto-refresh session check (JavaScript, 5-minute intervals)

2. SQL Injection Prevention

- PDO prepared statements everywhere
- Parameter binding for all user inputs

3. XSS Prevention

- `htmlspecialchars()` on all output
- No direct echo of user input

4. CSRF Protection

- Session-based user validation
- Action-based POST handlers

5. Audit Logging

- All critical actions logged to `system_logs` table
- Includes `user_id`, `action`, `module`, and `timestamp`

Logging Pattern (Use for Admin Side)

```
// Log important actions
$logStmt = $pdo->prepare("
    INSERT INTO system_logs (user_id, action, module, record_id, new_values)
    VALUES (?, ?, ?, ?, ?)
");
$logStmt->execute([
    $_SESSION['user_id'],
    'ACTION_NAME', // e.g., 'APPOINTMENT_BOOKED'
    'ModuleName', // e.g., 'Appointments'
    $recordId,
    json_encode(['key' => 'value']) // Additional data
]);
```

📁 MODULE 1: DASHBOARD

File: `Dashboard.php`

Purpose

Central hub displaying patient overview: stats, upcoming appointments, recent consultations, lab results, prescriptions, and announcements.

Key Queries

1. Patient Profile with Age

```
$stmt = $pdo->prepare("
    SELECT p.*, u.username, b.barangay_name,
           YEAR(CURDATE()) - YEAR(p.date_of_birth) as age
    FROM patients p
    JOIN users u ON p.user_id = u.id
    LEFT JOIN barangays b ON p.barangay_id = b.id
    WHERE u.id = ?
");
```

2. Upcoming Appointments (Limit 5)

```
$stmt = $pdo->prepare("
    SELECT a.*, st.service_name, st.service_category,
           CONCAT(s.first_name, ' ', s.last_name) as doctor_name,
           b.barangay_name
    FROM appointments a
    LEFT JOIN service_types st ON a.service_type_id = st.id
    LEFT JOIN staff s ON a.assigned_staff = s.id
    LEFT JOIN barangays b ON a.barangay_id = b.id
    WHERE a.patient_id = ?
    AND a.appointment_date >= CURDATE()
    AND a.status IN ('pending', 'confirmed')
    ORDER BY a.appointment_date ASC, a.appointment_time ASC
    LIMIT 5
");
```

3. Recent Consultations (Limit 5, Completed Only)

```
$stmt = $pdo->prepare("
    SELECT c.*, CONCAT(s.first_name, ' ', s.last_name) as doctor_name
    FROM consultations c
    LEFT JOIN staff s ON c.assigned_doctor = s.id
    WHERE c.patient_id = ? AND c.status = 'completed'
    ORDER BY c.consultation_date DESC
    LIMIT 5
");
```

4. Recent Lab Results (Limit 3)

```
$stmt = $pdo->prepare("
    SELECT lr.*, CONCAT(s.first_name, ' ', s.last_name) as performed_by_name
    FROM laboratory_results lr
    LEFT JOIN staff s ON lr.performed_by = s.id
    WHERE lr.patient_id = ? AND lr.status = 'completed'
    ORDER BY lr.test_date DESC
    LIMIT 3
");
```

5. Recent Prescriptions (Limit 3)

```
$stmt = $pdo->prepare("
    SELECT pr.*, CONCAT(s.first_name, ' ', s.last_name) as prescribed_by_name
    FROM prescriptions pr
    LEFT JOIN staff s ON pr.prescribed_by = s.id
    WHERE pr.patient_id = ?
    ORDER BY pr.prescription_date DESC
    LIMIT 3
");
```

6. Active Announcements

```
$stmt = $pdo->prepare("
    SELECT a.*, CONCAT(s.first_name, ' ', s.last_name) as author_name
    FROM announcements a
    JOIN staff s ON a.author_id = s.id
    WHERE a.status = 'published'
    AND (a.expiry_date IS NULL OR a.expiry_date > NOW())
    AND (a.target_audience IN ('all', 'patients'))
    ORDER BY a.priority DESC, a.publish_date DESC
    LIMIT 5
");
```

7. Unread Notifications Count

```
$stmt = $pdo->prepare("
    SELECT COUNT(*) as unread_count
    FROM notifications
    WHERE user_id = ?
    AND is_read = 0
    AND (expires_at IS NULL OR expires_at > NOW())
");
```

UI Components

Stats Cards (4 Metrics)

```

<div class="stats-row">
  <div class="stat-card">
    <div class="stat-number"><?php echo count($upcomingAppointments); ?></div>
    <div class="stat-label">Upcoming Appointments</div>
  </div>
  <!-- Repeat for Consultations, Prescriptions, Lab Results -->
</div>

```

Grid Layout Pattern

```

.appointments-grid,
.consultations-grid,
.lab-results-grid,
.prescriptions-grid,
.announcements-grid {
  display: grid;
  grid-template-columns: repeat(auto-fill, minmax(280px, 1fr));
  gap: 1rem;
}

```

Status Badge Pattern

```

<span class="item-status status-<?php echo $status; ?>">
  <?php echo ucfirst($status); ?>
</span>

```

```

.status-pending { background-color: #fff3cd; color: #856404; }
.status-confirmed { background-color: #d1ecf1; color: #0c5460; }
.status-completed { background-color: #d4edda; color: #155724; }

```

Business Rules

1. **Appointments:** Show only future dates with status = 'pending' or 'confirmed'
2. **Consultations:** Display only 'completed' status
3. **Announcements:** Filter by target_audience ('all' or 'patients'), published, not expired
4. **Empty States:** Show friendly message when no data available

For Admin Dashboard

Key Differences:

- Admin sees ALL patients' appointments (filtered by location/barangay)
- Show counts: Total Patients Today, Pending Appointments, Urgent Referrals
- Include filters by date range, barangay, service type
- Add quick actions: Add Patient, View Queue, Generate Reports

📌 MODULE 2: APPOINTMENTS

File: Appointment.php

Purpose

BHS appointment booking system. **RHU services are walk-in only** (no appointments needed).

Business Logic

Service Availability Rules

1. **RHU Services** = Walk-in only (no appointment booking)
2. **BHS Services** = Appointment required
3. **Both** = Depends on service configuration

Key Features

1. Service Selection

```
// Get available service types
$stmt = $pdo->prepare("
    SELECT * FROM service_types
    WHERE is_active = 1
    ORDER BY service_category, service_name
");
```

Service Display Logic:

```
// Only show BHS services in booking form
foreach ($services as $service) {
    if ($service['available_at'] == 'BHS') {
        // Add to booking options
    } elseif ($service['available_at'] == 'RHU') {
        // Show informational message about walk-in
    }
}
```

2. Appointment Booking (POST: `book_appointment`)

Validation Workflow:

```

// Step 1: Block RHU appointments
if ($appointment_location === 'RHU') {
    echo json_encode([
        'success' => false,
        'message' => 'RHU services do not require appointments. Please visit during operating hours.'
    ]);
    exit;
}

// Step 2: Validate BHS requires barangay
if ($appointment_location === 'BHS' && empty($barangay_id)) {
    echo json_encode([
        'success' => false,
        'message' => 'Please select a barangay for BHS appointment.'
    ]);
    exit;
}

// Step 3: Validate future date
if (strtotime($appointment_date) < strtotime('today')) {
    echo json_encode([
        'success' => false,
        'message' => 'Appointment date must be today or in the future.'
    ]);
    exit;
}

// Step 4: Check one appointment per day rule
$checkStmt = $pdo->prepare("
    SELECT COUNT(*) FROM appointments
    WHERE patient_id = ?
    AND appointment_date = ?
    AND status NOT IN ('cancelled', 'no-show', 'completed')
");
$checkStmt->execute([$patient['id'], $appointment_date]);
if ($checkStmt->fetchColumn() > 0) {
    echo json_encode([
        'success' => false,
        'message' => 'Only one appointment per day is allowed.'
    ]);
    exit;
}

// Step 5: Insert appointment with time = 00:00:00 (admin will schedule)
$stmt = $pdo->prepare("
    INSERT INTO appointments (
        patient_id, service_type_id, appointment_date, appointment_time,
        appointment_location, barangay_id, reason_for_visit, notes,
        status, created_by
    )
    VALUES (?, ?, ?, '00:00:00', ?, ?, ?, ?, 'pending', ?)
");

```

Important: Time Slot Logic

- Patient submits appointment with `appointment_time = '00:00:00'` (placeholder)
- Admin reviews and assigns actual time slot
- Patient receives notification with confirmed time

3. Appointment Cancellation (POST: `cancel_appointment`)

```
// Validation: Can only cancel pending/confirmed appointments
$checkStmt = $pdo->prepare("
    SELECT * FROM appointments
    WHERE id = ?
    AND patient_id = ?
    AND status IN ('pending', 'confirmed', 'rescheduled')
    AND appointment_date >= CURDATE()
");

// Update status
$stmt = $pdo->prepare("
    UPDATE appointments
    SET status = 'cancelled',
        cancelled_reason = ?,
        updated_at = NOW()
    WHERE id = ?
");
```

4. Get Patient Appointments

```
$stmt = $pdo->prepare("
    SELECT a.*, st.service_name, st.service_category,
        CONCAT(s.first_name, ' ', s.last_name) as doctor_name,
        b.barangay_name
    FROM appointments a
    LEFT JOIN service_types st ON a.service_type_id = st.id
    LEFT JOIN staff s ON a.assigned_staff = s.id
    LEFT JOIN barangays b ON a.barangay_id = b.id
    WHERE a.patient_id = ?
    ORDER BY a.appointment_date DESC, a.appointment_time DESC
");

// Separate into Upcoming and Past
$today = date('Y-m-d');
foreach ($allAppointments as $appointment) {
    if ($appointment['status'] == 'completed' ||
        $appointment['status'] == 'cancelled' ||
        $appointment['status'] == 'no-show') {
        $pastAppointments[] = $appointment;
    } else if ($appointment['appointment_date'] >= $today) {
        $upcomingAppointments[] = $appointment;
    } else {
        $pastAppointments[] = $appointment;
    }
}
```

UI Components

Service Selector with Preview

```

document.getElementById('serviceSelector').addEventListener('change', function() {
  const selectedOption = this.options[this.selectedIndex];

  // Extract data attributes
  selectedServiceData = {
    id: this.value,
    name: selectedOption.dataset.name,
    category: selectedOption.dataset.category,
    description: selectedOption.dataset.description,
    availableAt: selectedOption.dataset.availableAt
  };

  // Show service details card
  document.getElementById('serviceDetailsCard').style.display = 'block';
});

```

Booking Modal

```

function showBookingModal(serviceData) {
  const modalHTML = `
    <div class="modal-body">
      <form id="bookingForm">
        <input type="date" id="appointment_date" required
          min="${tomorrow}">
        <select id="barangay_id" required>...</select>
        <textarea id="reason_for_visit"></textarea>
        <textarea id="notes"></textarea>
      </form>
    </div>
  `;
}

```

Tab Navigation (Upcoming / Past)

```

document.querySelectorAll('.tab-btn').forEach(btn => {
  btn.addEventListener('click', function() {
    const tabName = this.dataset.tab;

    // Remove active from all
    document.querySelectorAll('.tab-btn').forEach(b =>
      b.classList.remove('active'));
    document.querySelectorAll('.tab-content').forEach(c =>
      c.classList.remove('active'));

    // Add active to selected
    this.classList.add('active');
    document.getElementById(tabName).classList.add('active');
  });
});

```

Notification Pattern

```
// Create notification after booking
$notificationStmt = $pdo->prepare("
    INSERT INTO notifications (user_id, type, title, message, data)
    VALUES (?, 'appointment_reminder', 'Appointment Booked', ?, ?)
");
$notificationStmt->execute([
    $_SESSION['user_id'],
    'Your appointment is pending admin approval. You will be notified once confirmed.',
    json_encode(['appointment_id' => $appointmentId])
]);
```

For Admin Side

Admin Appointment Management Should Include:

1. **View All Appointments** (by location, barangay, date)
2. **Confirm/Schedule Appointments**
 - Assign specific time slot to pending appointments
 - Assign staff member
3. **Calendar View** (daily/weekly schedule)
4. **Time Slot Management**
5. **Bulk Actions** (confirm multiple, send reminders)
6. **Appointment History** (all statuses)
7. **No-show Tracking**
8. **Reports** (appointments by service, barangay, status)

Key Admin Query:

```
// Get pending appointments needing time assignment
SELECT a.*, p.first_name, p.last_name, p.phone,
       st.service_name, b.barangay_name
FROM appointments a
JOIN patients p ON a.patient_id = p.id
JOIN service_types st ON a.service_type_id = st.id
LEFT JOIN barangays b ON a.barangay_id = b.id
WHERE a.status = 'pending'
AND a.appointment_time = '00:00:00'
ORDER BY a.created_at ASC
```

📁 MODULE 3: ONLINE CONSULTATION

File: `Consultation.php`

Purpose

Online consultation request system with Google Meet integration. Patients request consultations, admin schedules and creates meeting links.

Workflow

```
Patient Request → Admin Review → Admin Schedules →
Admin Creates Google Meet Link → Consultation Happens →
Admin Completes with Diagnosis/Treatment
```

Key Features

1. Request Online Consultation (POST: `request_consultation`)

```

// Generate consultation number
$year = date('Y');
$stmt = $pdo->prepare("
    SELECT COUNT(*) FROM consultations
    WHERE consultation_number LIKE ?
");
$stmt->execute(["CONS-$year-"]);
$count = $stmt->fetchColumn();
$consultation_number = 'CONS-' . $year . '-' . str_pad($count + 1, 4, '0', STR_PAD_LEFT);

// Insert as PENDING (admin needs to accept first)
$stmt = $pdo->prepare("
    INSERT INTO consultations (
        consultation_number, patient_id, consultation_type,
        consultation_date, chief_complaint, history_of_present_illness,
        symptoms, priority, status, consultation_location
    )
    VALUES (?, ?, 'online', NOW(), ?, ?, ?, 'medium', 'pending', 'RHU')
");

```

Business Rule:

- All online consultation requests start as `status = 'pending'`
- Priority automatically set to 'medium' (admin can adjust)
- `consultation_date` = request time, will be updated when scheduled

2. Cancel Consultation Request (POST: `cancel_consultation`)

```

// Only pending consultations can be cancelled by patient
$checkStmt = $pdo->prepare("
    SELECT id, consultation_number, status
    FROM consultations
    WHERE id = ? AND patient_id = ? AND status = 'pending'
");

if ($consultation) {
    $stmt = $pdo->prepare("
        UPDATE consultations
        SET status = 'cancelled'
        WHERE id = ?
    ");
}

```

3. Get Patient Consultations

```

$stmt = $pdo->prepare("
    SELECT c.*, CONCAT(s.first_name, ' ', s.last_name) as doctor_name,
        b.barangay_name as consultation_location_name
    FROM consultations c
    LEFT JOIN staff s ON c.assigned_doctor = s.id
    LEFT JOIN barangays b ON c.barangay_id = b.id
    WHERE c.patient_id = ? AND c.consultation_type = 'online'
    ORDER BY c.consultation_date DESC, c.created_at DESC
");

```

Consultation Statuses

Status Workflow

1. **pending** - Patient submitted, waiting for admin review

```

// UI shows: "Waiting for Approval" message
// Actions: Cancel button available

```

2. **in_progress** - Admin accepted, scheduled consultation time

```
// UI shows: Scheduled time, assigned doctor
// Actions: Google Meet link (when available)
// google_meet_link field populated by admin
```

3. **completed** - Consultation finished, diagnosis entered

```
// UI shows: Diagnosis, treatment plan, prescriptions link
// Actions: View health record, view prescriptions
```

4. **cancelled** - Cancelled by patient or admin

```
// UI shows: "Cancelled" badge
// No actions available
```

Google Meet Integration

Patient View:

```
<?php if ($consultation['status'] == 'in_progress'): ?>
    <?php if (!empty($consultation['google_meet_link'])): ?>
        <a href="<?php echo htmlspecialchars($consultation['google_meet_link']); ?>"
            target="_blank" class="gmeet-link">
            <i class="fab fa-google me-2"></i>Join Video Consultation
        </a>
    <?php else: ?>
        <button class="gmeet-link" disabled>
            <i class="fab fa-google me-2"></i>Waiting for Meeting Link
        </button>
        <small class="text-muted">
            The doctor will send the Google Meet link at your scheduled time.
        </small>
    <?php endif; ?>
<?php endif; ?>
```

Admin Responsibility:

- Create Google Meet meeting
- Copy meeting link
- Paste into consultations.google_meet_link field
- Patient automatically sees link when page refreshes

UI States

Pending State

```
<div class="alert alert-warning">
    <i class="fas fa-hourglass-half"></i>
    <strong>Waiting for Approval:</strong> Your request is pending.
</div>
<button onclick="cancelConsultation(<?php echo $id; ?>)">
    Cancel Request
</button>
```

In Progress State

```

<div class="alert alert-success">
    <i class="fas fa-check-circle"></i>
    <strong>Consultation Approved!</strong> Scheduled for:
    <?php echo date('F j, Y @ g:i A', strtotime($consultation_date)); ?>
</div>
<div class="scheduled-time-box">
    <i class="fas fa-calendar-check"></i>
    <strong>Your Scheduled Time:</strong>
    <span><?php echo $formatted_time; ?></span>
    <small>Be online 5 minutes early.</small>
</div>

```

Completed State

```

<div class="alert alert-success">
    <strong>Consultation Completed</strong>
    Attended by: <?php echo $doctor_name; ?>
</div>
<div class="record-label">Diagnosis</div>
<div class="record-value"><?php echo htmlspecialchars($diagnosis); ?></div>

<div class="record-label">Treatment Plan</div>
<div class="record-value"><?php echo nl2br(htmlspecialchars($treatment_plan)); ?></div>

<div class="d-flex gap-2">
    <a href="Health_Record.php" class="btn btn-outline-primary">
        View Full Health Record
    </a>
</div>

```

For Admin Side

Admin Online Consultation Management:

1. View Pending Requests

```

SELECT c.*, p.first_name, p.last_name, p.phone, p.email
FROM consultations c
JOIN patients p ON c.patient_id = p.id
WHERE c.status = 'pending'
AND c.consultation_type = 'online'
ORDER BY c.priority DESC, c.created_at ASC

```

2. Accept & Schedule Consultation

- Update status = 'in_progress'
- Assign assigned_doctor
- Set consultation_date (scheduled time)
- Create Google Meet link
- Update google_meet_link field
- Send notification to patient

3. Conduct Consultation

- Use Google Meet link
- Take notes during consultation

4. Complete Consultation

- Update status = 'completed'
- Enter vital signs (JSON format)
- Enter diagnosis, treatment plan
- Create prescriptions (if needed)
- Order lab tests (if needed)
- Issue medical certificate (if needed)

Admin Query Example:


```
// Update consultation after scheduling
$stmt = $pdo->prepare("
    UPDATE consultations
    SET status = 'in_progress',
        assigned_doctor = ?,
        consultation_date = ?,
        google_meet_link = ?
    WHERE id = ?
");
$stmt->execute([$doctor_id, $scheduled_datetime, $meet_link, $consultation_id]);

// Notify patient
$notifStmt = $pdo->prepare("
    INSERT INTO notifications (user_id, type, title, message, data, priority)
    VALUES (?, 'consultation_ready', 'Consultation Scheduled', ?, ?, 'high')
");
```

📁 MODULE 4: HEALTH RECORDS

File: `Health_Record.php`

Purpose

Comprehensive view of patient's complete medical history: consultations, lab results, prescriptions, certificates, and referrals.

Tab Structure

Consultations	Lab Results	Prescriptions
Certificates	Referrals	

Key Features

1. Tab Navigation

```
document.querySelectorAll('.tab-btn').forEach(btn => {
    btn.addEventListener('click', function() {
        const tabName = this.dataset.tab;

        // Switch active tab
        document.querySelectorAll('.tab-btn').forEach(b =>
            b.classList.remove('active'));
        document.querySelectorAll('.tab-content').forEach(c =>
            c.classList.remove('active'));

        this.classList.add('active');
        document.getElementById(tabName).classList.add('active');
    });
});
```

2. Filter System

```
function filterItems(tabId, status) {
    const items = document.querySelectorAll(`#${tabId} .record-card`);

    items.forEach(item => {
        if (status === 'all') {
            item.style.display = '';
        } else {
            const itemStatus = item.dataset.status;
            item.style.display = itemStatus === status ? '' : 'none';
        }
    });

    // Show empty state if no results
    const visibleItems = Array.from(items).filter(
        item => item.style.display !== 'none'
    );
    const emptyState = document.querySelector(`#${tabId} .empty-filter-state`);

    if (visibleItems.length === 0 && emptyState) {
        emptyState.style.display = 'block';
    } else if (emptyState) {
        emptyState.style.display = 'none';
    }
}
```

Tab 1: Consultations

Query

```
$stmt = $pdo->prepare("
    SELECT c.*,
           CONCAT(s.first_name, ' ', s.last_name) as doctor_name,
           b.barangay_name as consultation_location_name
    FROM consultations c
    LEFT JOIN staff s ON c.assigned_doctor = s.id
    LEFT JOIN barangays b ON c.barangay_id = b.id
    WHERE c.patient_id = ?
    ORDER BY c.consultation_date DESC
");
```

Display Pattern

```
<div class="record-card consultation-card" data-status="<?php echo $status; ?>">
  <div class="record-header">
    <div class="record-date">
      <?php echo date('F j, Y g:i A', strtotime($consultation_date)); ?>
    </div>
    <span class="record-status status-<?php echo $status; ?>">
      <?php echo ucfirst($status); ?>
    </span>
  </div>

  <div class="record-content">
    <div class="record-label">Consultation Number</div>
    <div class="record-value"><?php echo $consultation_number; ?></div>

    <div class="record-label">Type</div>
    <div class="record-value"><?php echo ucfirst($consultation_type); ?></div>

    <div class="record-label">Chief Complaint</div>
    <div class="record-value"><?php echo htmlspecialchars($chief_complaint); ?></div>

    <!-- Vital Signs (JSON) -->
    <?php if ($vital_signs): ?>
      <div class="vital-signs">
        <div class="vital-signs-grid">
          <?php
            $vitals = json_decode($vital_signs, true);
            foreach ($vitals as $key => $value):
              if ($value):
                ?>
                <div class="vital-item">
                  <div class="vital-label"><?php echo ucfirst(str_replace('_', ' ', $key)); ?></div>
                  <div class="vital-value"><?php echo htmlspecialchars($value); ?></div>
                </div>
              <?php
                endif;
              endforeach;
            ?>
          </div>
        </div>
      <?php endif; ?>

    <!-- Diagnosis, Treatment Plan, etc. -->
  </div>
</div>
```

Vital Signs JSON Format:

```
{
  "temperature": "38.5",
  "blood_pressure": "120/80",
  "pulse_rate": "85",
  "respiratory_rate": "18",
  "weight": "70",
  "height": "170"
}
```

Tab 2: Laboratory Results

Query

```
$stmt = $pdo->prepare("
    SELECT lr.*,
           CONCAT(s1.first_name, ' ', s1.last_name) as performed_by_name,
           CONCAT(s2.first_name, ' ', s2.last_name) as verified_by_name,
           c.consultation_number
    FROM laboratory_results lr
    LEFT JOIN staff s1 ON lr.performed_by = s1.id
    LEFT JOIN staff s2 ON lr.verified_by = s2.id
    LEFT JOIN consultations c ON lr.consultation_id = c.id
    WHERE lr.patient_id = ?
    ORDER BY lr.test_date DESC
");
```

Display Pattern

```
<div class="record-card lab-card" data-status="<?php echo $status; ?>">
    <div class="record-label">Lab Number</div>
    <div class="record-value"><?php echo $lab_number; ?></div>

    <div class="record-label">Test Name</div>
    <div class="record-value"><?php echo htmlspecialchars($test_type); ?></div>

    <!-- Test Results (JSON) -->
    <?php if ($test_results && $status == 'completed'): ?>
        <div class="lab-results-data">
            <?php
                $results = json_decode($test_results, true);
                if ($results && is_array($results)):
                    foreach ($results as $key => $value):
                        ?>
                            <div>
                                <strong><?php echo ucfirst(str_replace('_', ' ', $key)); ?></strong>
                                <?php echo htmlspecialchars($value); ?>
                            </div>
                        <?php endforeach; endif; ?>
                    </div>
                <?php endif; ?>

            <div class="record-label">Normal Range</div>
            <div class="record-value"><?php echo htmlspecialchars($normal_range); ?></div>

            <div class="record-label">Interpretation</div>
            <div class="record-value lab-normal"><?php echo htmlspecialchars($interpretation); ?></div>
        </div>
```

Test Results JSON Format:

```
{
    "WBC": "11.5",
    "RBC": "4.8",
    "Hemoglobin": "14.2",
    "Hematocrit": "42.1",
    "Platelet": "250"
}
```

Tab 3: Prescriptions

Query

```
$stmt = $pdo->prepare("
    SELECT pr.*,
           CONCAT(s1.first_name, ' ', s1.last_name) as prescribed_by_name,
           CONCAT(s2.first_name, ' ', s2.last_name) as dispensed_by_name,
           c.consultation_number
    FROM prescriptions pr
    LEFT JOIN staff s1 ON pr.prescribed_by = s1.id
    LEFT JOIN staff s2 ON pr.dispensed_by = s2.id
    LEFT JOIN consultations c ON pr.consultation_id = c.id
    WHERE pr.patient_id = ?
    ORDER BY pr.prescription_date DESC
");
```

Display Pattern

```
<div class="record-card prescription-card" data-status="<?php echo $status; ?>">
    <div class="medication-name"><?php echo htmlspecialchars($medication_name); ?></div>

    <div class="record-label">Generic Name</div>
    <div class="record-value"><?php echo htmlspecialchars($generic_name); ?></div>

    <div class="dosage-info">
        <div><strong>Strength:</strong> <?php echo $dosage_strength; ?></div>
        <div><strong>Form:</strong> <?php echo $dosage_form; ?></div>
        <div><strong>Frequency:</strong> <?php echo $frequency; ?></div>
        <div><strong>Duration:</strong> <?php echo $duration; ?></div>
        <div><strong>Quantity:</strong> <?php echo $quantity_prescribed; ?>
            <?php if ($status != 'pending'): ?>
                (Dispensed: <?php echo $quantity_dispensed; ?>)
            <?php endif; ?>
        </div>
    </div>

    <div class="record-label">Instructions</div>
    <div class="record-value"><?php echo nl2br(htmlspecialchars($dosage_instructions)); ?></div>
</div>
```

Tab 4: Medical Certificates

Query

```
$stmt = $pdo->prepare("
    SELECT mc.*,
           CONCAT(s.first_name, ' ', s.last_name) as issued_by_name
    FROM medical_certificates mc
    LEFT JOIN staff s ON mc.issued_by = s.id
    WHERE mc.patient_id = ?
    ORDER BY mc.date_issued DESC
");
```

Display Pattern with Download

```

<div class="record-card certificate-card" data-status="<?php echo $status; ?>">
    <span class="certificate-number"><?php echo $certificate_number; ?></span>

    <div class="record-label">Certificate Type</div>
    <div class="record-value"><?php echo htmlspecialchars($certificate_type); ?></div>

    <div class="record-label">Purpose</div>
    <div class="record-value"><?php echo htmlspecialchars($purpose); ?></div>

    <?php if ($status == 'issued'): ?>
        <div style="margin-top: 1rem; text-align: center;">
            <button onclick="window.open('../Admin/Rural_Health_Unit/print_certificate.php?id=<?php echo $id; ?>', '_blank')"
                style="background: var(--kawit-gradient); color: white; border: none; padding: 10px 20px; border-radius: 10px; w
                <i class="fas fa-download"></i> Download Certificate
            </button>
        </div>
    <?php endif; ?>
</div>

```

Tab 5: Referrals

Query

```

$stmt = $pdo->prepare("
    SELECT r.*,
           CONCAT(s.first_name, ' ', s.last_name) as referred_by_name,
           c.consultation_number
    FROM referrals r
    LEFT JOIN staff s ON r.referred_by = s.id
    LEFT JOIN consultations c ON r.consultation_id = c.id
    WHERE r.patient_id = ?
    ORDER BY r.referral_date DESC
");

```

Display Pattern

```

<div class="record-card referral-card urgency-<?php echo $urgency_level; ?>"
    data-status="<?php echo $status; ?>">

    <div class="record-label">Referral Number</div>
    <div class="record-value"><?php echo $referral_number; ?></div>

    <div class="record-label">Referred To</div>
    <div class="record-value"><?php echo htmlspecialchars($referred_to_facility); ?></div>

    <div class="record-label">Urgency Level</div>
    <div class="record-value">
        <span class="badge bg-<?php echo $urgency_level == 'urgent' ? 'danger' : 'success'; ?>">
            <?php echo ucfirst($urgency_level); ?>
        </span>
    </div>

    <div class="record-label">Reason for Referral</div>
    <div class="record-value"><?php echo nl2br(htmlspecialchars($referral_reason)); ?></div>
</div>

```

For Admin Side

Admin Health Records Management:

1. **View Any Patient's Records** (with patient selector)
2. **Add New Records** (consultations, lab results, prescriptions)
3. **Edit/Update Records** (before completion)

4. **Print/Export Records** (PDF generation)
5. **Medical History Timeline** (chronological view)
6. **Search Functionality** (by date, type, diagnosis)

📄 MODULE 5: MEDICAL CERTIFICATES

File: `Medical_Certificates.php`

Purpose

Request and view medical certificates for employment, school, travel, and other purposes.

Key Features

1. Request Certificate (POST: `request_certificate`)

```
// Generate certificate number
$year = date('Y');
$stmt = $pdo->prepare("
    SELECT COUNT(*) FROM medical_certificates
    WHERE certificate_number LIKE ?
");
$stmt->execute(["CERT-$year-"]);
$count = $stmt->fetchColumn();
$cert_number = 'CERT-' . $year . '-' . str_pad($count + 1, 4, '0', STR_PAD_LEFT);

// Handle "Other" purpose
$final_purpose = ($purpose === 'Other') ? $other_purpose : $purpose;

// Insert as pending (admin will process)
$stmt = $pdo->prepare("
    INSERT INTO medical_certificates (
        certificate_number, patient_id, issued_by,
        certificate_type, purpose, date_issued, status
    )
    VALUES (?, ?, 1, ?, ?, CURDATE(), 'pending')
");
$stmt->execute([
    $cert_number,
    $patient['id'],
    $certificate_type,
    $final_purpose
]);
```

Business Rules:

- Requests start with `status = 'pending'`
- `issued_by` defaults to 1 (system default, admin can reassign)
- `date_issued` = request date
- Admin processes: examines patient, fills details, changes to 'issued'

2. Get Patient Certificates

```

$stmt = $pdo->prepare("
    SELECT mc.*,
           CONCAT(s.first_name, ' ', s.last_name) as issued_by_name,
           c.consultation_number
    FROM medical_certificates mc
    LEFT JOIN staff s ON mc.issued_by = s.id
    LEFT JOIN consultations c ON mc.consultation_id = c.id
    WHERE mc.patient_id = ?
    ORDER BY
        CASE mc.status
            WHEN 'pending' THEN 1
            WHEN 'draft' THEN 2
            WHEN 'issued' THEN 3
            ELSE 4
        END,
        mc.date_issued DESC
");

```

Sorting Logic: Pending first, then draft, then issued, then others

3. Certificate Types & Purposes

Certificate Types:

- Medical Certificate (general)
- Fit to Work Certificate
- Health Certificate
- Vaccination Certificate
- Medical Clearance

Common Purposes:

- Employment
- School Requirements
- Travel Requirements
- Insurance Claim
- Government Requirements
- Visa Application
- License Application
- Other (with specification)

UI Components

Request Form


```

<form id="requestForm">
  <select id="certificate_type" required>
    <option value="">Select certificate type</option>
    <option value="Medical Certificate">General Medical Certificate</option>
    <option value="Fit to Work Certificate">Fit to Work Certificate</option>
    <option value="Health Certificate">Health Certificate</option>
    <option value="Vaccination Certificate">Vaccination Certificate</option>
    <option value="Medical Clearance">Medical Clearance</option>
  </select>

  <select id="purpose" required>
    <option value="">Select purpose</option>
    <option value="Employment">Employment</option>
    <option value="School Requirements">School Requirements</option>
    <option value="Travel">Travel Requirements</option>
    <option value="Insurance">Insurance Claim</option>
    <option value="Government Requirements">Government Requirements</option>
    <option value="Visa Application">Visa Application</option>
    <option value="License Application">License Application</option>
    <option value="Other">Other (Please specify)</option>
  </select>

  <div id="otherPurposeDiv" style="display: none;">
    <input type="text" id="other_purpose" placeholder="Enter specific purpose">
  </div>
</form>

```

Purpose Toggle Logic

```

document.getElementById('purpose').addEventListener('change', function() {
  const otherDiv = document.getElementById('otherPurposeDiv');
  const otherInput = document.getElementById('other_purpose');

  if (this.value === 'Other') {
    otherDiv.style.display = 'block';
    otherInput.required = true;
  } else {
    otherDiv.style.display = 'none';
    otherInput.required = false;
    otherInput.value = '';
  }
});

```

Certificate Display by Status

Pending:

```

<div class="certificate-actions">
  <small class="text-muted">
    <i class="fas fa-clock"></i>
    Your certificate is being processed. You will be notified when ready.
  </small>
</div>

```

Issued:

```

<div class="certificate-actions">
  <a href="#" class="btn-download"
    onclick="event.preventDefault(); downloadCertificate(<?php echo $id; ?>);">
    <i class="fas fa-download"></i> Download PDF
  </a>
</div>

```

```
function downloadCertificate(certificateId) {
    // Path from Patient folder to Admin folder
    window.open('../Admin/Rural_Health_Unit/print_certificate.php?id=' + certificateId, '_blank');
    return false;
}
```

For Admin Side

Admin Certificate Management:

1. View Pending Requests

```
SELECT mc.*, p.first_name, p.last_name, p.date_of_birth,
       p.address, p.phone
FROM medical_certificates mc
JOIN patients p ON mc.patient_id = p.id
WHERE mc.status = 'pending'
ORDER BY mc.created_at ASC
```

2. Process Certificate

- View patient information
- Option to schedule examination (if needed)
- Link to recent consultation (optional)
- Fill certificate details:
 - diagnosis
 - physical_findings
 - recommendations
 - restrictions
 - fitness_status
 - valid_from, valid_until dates
- Generate QR code for verification
- Update status to 'issued'

3. Print/Generate PDF

- Create print_certificate.php in Admin folder
- Use TCPDF or similar library
- Include QR code, digital signature
- Official RHU letterhead

4. Certificate Template System

- Different templates for different types
- Customizable fields
- E-signature integration

Admin Query Example:

```
// Complete certificate issuance
$stmt = $pdo->prepare("
    UPDATE medical_certificates
    SET status = 'issued',
        diagnosis = ?,
        physical_findings = ?,
        recommendations = ?,
        fitness_status = ?,
        valid_from = ?,
        valid_until = ?,
        qr_code = ?,
        consultation_id = ?
    WHERE id = ?
");

// Generate QR code (certificate verification)
$qr_data = json_encode([
    'cert_number' => $certificate_number,
    'patient_id' => $patient_id,
    'issued_date' => date('Y-m-d'),
    'verify_url' => 'https://kawitrhu.gov.ph/verify/' . $certificate_number
]);
```

📄 MODULE 6: PROFILE MANAGEMENT

File: `Profile.php`

Purpose

Patient profile editing and password management. Clear distinction between editable and read-only fields.

Field Categories

Read-Only Fields (Cannot be changed by patient)

```
// System-assigned or sensitive data
- patient_id           // Auto-generated
- username             // Set at registration
- first_name           // Legal name
- middle_name
- last_name
- suffix
- date_of_birth        // Legal document
- age                  // Calculated
- gender               // Legal document
- blood_type           // Requires lab verification
```

Editable Fields (Patient can update)

```
- civil_status
- religion
- occupation
- educational_attainment
- phone
- email
- address
- barangay_id
- allergies
- philhealth_number
- emergency_contact_name
- emergency_contact_phone
- emergency_contact_relationship
```

Key Features

1. Update Profile (POST: `update_profile`)

```
$stmt = $pdo->prepare("
    UPDATE patients
    SET civil_status = ?,
        address = ?,
        barangay_id = ?,
        phone = ?,
        email = ?,
        allergies = ?,
        philhealth_number = ?,
        occupation = ?,
        educational_attainment = ?,
        religion = ?,
        emergency_contact_name = ?,
        emergency_contact_phone = ?,
        emergency_contact_relationship = ?,
        updated_at = CURRENT_TIMESTAMP
    WHERE id = ?
");

// Log the update
$logStmt = $pdo->prepare("
    INSERT INTO system_logs (user_id, action, module, record_id, new_values)
    VALUES (?, 'PROFILE_UPDATE', 'Patients', ?, ?)
");
```

2. Change Password (POST: change_password)

```

// Get current password
$stmt = $pdo->prepare("SELECT password FROM users WHERE id = ?");
$stmt->execute([$SESSION['user_id']]);
$user = $stmt->fetch(PDO::FETCH_ASSOC);

// Verify current password
if (!password_verify($current_password, $user['password'])) {
    echo json_encode(['success' => false, 'message' => 'Current password is incorrect']);
    exit;
}

// Validate new password
if ($new_password !== $confirm_password) {
    echo json_encode(['success' => false, 'message' => 'New passwords do not match']);
    exit;
}

if (strlen($new_password) < 6) {
    echo json_encode(['success' => false, 'message' => 'Password must be at least 6 characters']);
    exit;
}

// Update password
$hashed_password = password_hash($new_password, PASSWORD_DEFAULT);
$stmt = $pdo->prepare("
    UPDATE users
    SET password = ?,
        password_changed_at = CURRENT_TIMESTAMP,
        updated_at = CURRENT_TIMESTAMP
    WHERE id = ?
");

// Log password change
$logStmt = $pdo->prepare("
    INSERT INTO system_logs (user_id, action, module, record_id)
    VALUES (?, 'PASSWORD_CHANGE', 'Users', ?)
");

```

UI Components

Form Sections

```

<!-- Read-Only Section -->
<div class="form-section">
    <div class="form-section-title">
        <i class="fas fa-lock"></i> System Information (Cannot be changed)
    </div>
    <input type="text" class="form-control readonly-field"
        value="<?php echo $patient_id; ?>" readonly>
</div>

<!-- Editable Section -->
<div class="form-section">
    <div class="form-section-title">
        <i class="fas fa-edit"></i> Personal Details
    </div>
    <select class="form-select" id="civil_status" required>
        <option value="Single" <?php echo $civil_status == 'Single' ? 'selected' : ''; ?>>Single</option>
        <!-- ... -->
    </select>
</div>

```

Input Formatters

Phone Number Formatting:

```
function formatPhoneNumber(input) {
  let value = input.value.replace(/\D/g, '');
  if (value.length > 11) value = value.slice(0, 11);

  if (value.length >= 4) {
    value = value.slice(0, 4) + '-' + value.slice(4);
  }
  if (value.length >= 9) {
    value = value.slice(0, 9) + '-' + value.slice(9);
  }

  input.value = value;
}

document.getElementById('phone').addEventListener('input', function() {
  formatPhoneNumber(this);
});
// Format: 0917-123-4567
```

PhilHealth Number Formatting:

```
document.getElementById('philhealth_number').addEventListener('input', function(e) {
  let value = e.target.value.replace(/\D/g, '');
  if (value.length > 12) value = value.slice(0, 12);

  if (value.length >= 2) {
    value = value.slice(0, 2) + '-' + value.slice(2);
  }
  if (value.length >= 12) {
    value = value.slice(0, 12) + '-' + value.slice(12);
  }

  e.target.value = value;
});
// Format: XX-XXXXXXXX-X
```

Real-Time Timestamp Update

```

// After successful profile update
if (data.success) {
    showAlert(data.message, 'success');

    // Update timestamp immediately without page reload
    const now = new Date();
    const options = {
        year: 'numeric',
        month: 'long',
        day: 'numeric',
        hour: 'numeric',
        minute: '2-digit',
        hour12: true
    };
    const formattedDate = now.toLocaleDateString('en-US', options);

    // Find and update the timestamp
    const allParagraphs = document.querySelectorAll('.form-section p');
    allParagraphs.forEach(p => {
        if (p.innerHTML.includes('Last Profile Update:')) {
            p.innerHTML = '<strong>Last Profile Update:</strong> ' + formattedDate;
        }
    });
}

```

Account Information Section

```

<div class="form-section">
    <div class="form-section-title">Account Activity</div>
    <p><strong>Account Created:</strong>
        <?php echo date('F j, Y g:i A', strtotime($created_at)); ?>
    </p>
    <p><strong>Last Profile Update:</strong>
        <?php echo $updated_at ? date('F j, Y g:i A', strtotime($updated_at)) : 'Never updated'; ?>
    </p>
    <p><strong>Last Password Change:</strong>
        <?php echo $password_changed_at ? date('F j, Y g:i A', strtotime($password_changed_at)) : 'Never changed'; ?>
    </p>
    <p><strong>Account Status:</strong>
        <span class="badge bg-success">Active</span>
    </p>
</div>

```

For Admin Side

Admin Patient Profile Management:

1. View/Edit All Patient Information

- Can edit ALL fields (including read-only for patient)
- Add medical notes/flags
- Update blood type (with lab verification)
- Manage account status (active/inactive)

2. Complete Registration

- For walk-in patients without accounts
- Create user account + patient profile
- Generate patient_id

3. Bulk Import

- CSV import for multiple patients
- Data validation
- Duplicate checking

4. Patient Search

- By patient_id, name, phone, barangay
- Advanced filters

5. Patient History Summary

5. Patient History Summary

- Quick stats: total consultations, prescriptions, etc.
- Recent activity timeline
- Medical alerts (allergies, chronic conditions)

COMMON UI/UX PATTERNS

Color Scheme (CSS Variables)

```
:root {
  --kawit-pink: #FFA6BE;
  --light-pink: #FFE4E6;
  --dark-pink: #FF7A9A;
  --text-dark: #2c3e50;
  --kawit-gradient: linear-gradient(135deg, #FFA6BE 0%, #FF7A9A 100%);
  --light-bg: #f8f9fc;
}
```

Sidebar Navigation (Consistent Across All Modules)

```
<div class="sidebar">
  <div class="sidebar-header">
    <div class="logo-container">
      
      
      
    </div>
    <div class="logo-circle">
      <i class="fas fa-heartbeat"></i>
    </div>
    <div class="logo-text">
      KAWIT<br>
      <small style="font-size: 0.8rem; opacity: 0.8;">RHU</small>
    </div>
  </div>
  <nav class="sidebar-nav">
    <div class="nav-item">
      <a href="Dashboard.php" class="nav-link active">
        <i class="fas fa-home"></i>
        <span>Dashboard</span>
      </a>
    </div>
    <!-- Repeat for other modules -->
  </nav>
</div>
```

Active State Logic:

```
// Set active class based on current page
$current_page = basename($_SERVER['PHP_SELF']);

<a href="Dashboard.php" class="nav-link <?php echo $current_page == 'Dashboard.php' ? 'active' : ''; ?>">
```

Top Navigation Bar (Standard Header)


```

<nav class="top-navbar">
  <h1 class="page-title">Page Title</h1>
  <div class="user-info">
    <div class="user-avatar">
      <?php
        $initials = strtoupper(
          substr($patient['first_name'], 0, 1) .
          substr($patient['last_name'], 0, 1)
        );
        echo $initials;
      ?>
    </div>
    <div class="user-details">
      <div class="user-name">
        <?php echo htmlspecialchars($patient['first_name'] . ' ' . substr($patient['last_name'], 0, 1) . '.'); ?>
      </div>
      <div class="user-role">
        Patient ID: <?php echo htmlspecialchars($patient['patient_id']); ?>
      </div>
    </div>

    <!-- User Menu Dropdown -->
    <div class="dropdown">
      <button class="btn btn-link text-dark p-0 ms-2" type="button" data-bs-toggle="dropdown">
        <i class="fas fa-chevron-down"></i>
      </button>
      <ul class="dropdown-menu dropdown-menu-end">
        <li>
          <a class="dropdown-item" href="Profile.php">
            <i class="fas fa-user-edit me-2"></i>Edit Profile
          </a>
        </li>
        <li><hr class="dropdown-divider"></li>
        <li>
          <a class="dropdown-item text-danger" href="../logout.php"
            onclick="return confirm('Are you sure you want to log out?')">
            <i class="fas fa-sign-out-alt me-2"></i>Log Out
          </a>
        </li>
      </ul>
    </div>
  </div>
</nav>

```

Content Section Layout

```

<div class="content-section">
  <div class="section-header">
    <div class="section-icon">
      <i class="fas fa-icon-name"></i>
    </div>
    <h3 class="section-title">Section Title</h3>
  </div>

  <!-- Content here -->
</div>

```

```
.content-section {
  background: white;
  border-radius: 20px;
  padding: 2rem;
  box-shadow: 0 8px 25px rgba(0, 0, 0, 0.1);
  margin-bottom: 2rem;
  border-left: 5px solid var(--kawit-pink);
}
```

Button Styles

```
/* Primary Action Button */
.btn-save,
.btn-request,
.btn-book {
  background: var(--kawit-gradient);
  border: none;
  color: white;
  padding: 12px 30px;
  border-radius: 10px;
  font-weight: 600;
  transition: all 0.3s ease;
}

.btn-save:hover {
  transform: translateY(-2px);
  box-shadow: 0 5px 15px rgba(255, 122, 154, 0.4);
  color: white;
}

/* Danger Button */
.btn-cancel {
  background: #dc3545;
  border: none;
  color: white;
  padding: 6px 12px;
  border-radius: 6px;
  font-size: 0.8rem;
  font-weight: 600;
  transition: all 0.3s ease;
}

.btn-cancel:hover {
  background: #c82333;
  color: white;
}
```

Status Badge Patterns

```
<span class="status-badge status-<?php echo $status; ?>">
  <?php echo ucfirst($status); ?>
</span>
```

```

/* Appointment Statuses */
.status-pending { background-color: #fff3cd; color: #856404; }
.status-confirmed { background-color: #d1ecf1; color: #0c5460; }
.status-completed { background-color: #d4edda; color: #155724; }
.status-cancelled { background-color: #f8d7da; color: #721c24; }
.status-rescheduled { background-color: #e2e3e5; color: #383d41; }
.status-no-show { background-color: #f8d7da; color: #721c24; }

/* Consultation Statuses */
.status-in-progress { background-color: #d1ecf1; color: #0c5460; }

/* Prescription Statuses */
.status-partially-dispensed { background-color: #fff3cd; color: #856404; }
.status-fully-dispensed { background-color: #d4edda; color: #155724; }
.status-expired { background-color: #f8d7da; color: #721c24; }

/* Certificate Statuses */
.status-draft { background-color: #fff3cd; color: #856404; }
.status-issued { background-color: #d4edda; color: #155724; }
.status-active { background-color: #d4edda; color: #155724; }
.status-revoked { background-color: #f8d7da; color: #721c24; }

/* Priority Levels */
.priority-low { background-color: #e8f5e8; color: #2e7d32; }
.priority-medium { background-color: #fff3e0; color: #f57c00; }
.priority-high { background-color: #ffebee; color: #c62828; }
.priority-urgent { background-color: #ffebee; color: #c62828; }

```

Grid Layout Pattern

```

/* Responsive Grid for Cards */
.items-grid {
  display: grid;
  grid-template-columns: repeat(auto-fill, minmax(350px, 1fr));
  gap: 1.5rem;
}

/* Responsive Breakpoints */
@media (max-width: 1200px) {
  .items-grid {
    grid-template-columns: repeat(auto-fill, minmax(300px, 1fr));
  }
}

@media (max-width: 768px) {
  .items-grid {
    grid-template-columns: 1fr;
  }
}

```

Empty State Pattern

```

<div class="empty-state">
  <i class="fas fa-icon-name"></i>
  <h5>No Data Available</h5>
  <p>Description of what will appear here.</p>
  <button class="btn btn-primary" onclick="action()">
    Take Action
  </button>
</div>

```

```

.empty-state {
  text-align: center;
  padding: 3rem;
  color: #6c757d;
}

.empty-state i {
  font-size: 4rem;
  margin-bottom: 1rem;
  opacity: 0.5;
}

```

Loading States

```

// Button loading state
button.disabled = true;
button.innerHTML = '<i class="fas fa-spinner fa-spin me-2"></i>Loading...';

// After completion
button.disabled = false;
button.innerHTML = '<i class="fas fa-save me-2"></i>Save Changes';

```

Alert Messages

```

function showAlert(message, type = 'danger') {
  const alertDiv = document.getElementById('alertMessage');
  alertDiv.className = `alert alert-${type}`;
  alertDiv.innerHTML = `
    <i class="fas fa-${type === 'success' ? 'check-circle' : 'exclamation-triangle'} me-2"></i>
    ${message}
  `;
  alertDiv.style.display = 'block';

  // Scroll to top
  window.scrollTo({top: 0, behavior: 'smooth'});

  // Auto-hide success messages
  if (type === 'success') {
    setTimeout(() => {
      alertDiv.style.display = 'none';
    }, 5000);
  }
}

```

Form Validation Patterns

```
// Required field validation
if (!field.value.trim()) {
    alert('Please fill in all required fields.');
```



```
    return;
}

// Email validation
const emailPattern = /^[^\\s@]+@[^\\s@]+\\.([^\\s@]+)$/;
if (!emailPattern.test(email.value)) {
    alert('Please enter a valid email address.');
```



```
    return;
}

// Phone validation (Philippine format)
const phonePattern = /^09\d{9}$/;
if (!phonePattern.test(phone.value.replace(/\\D/g, ''))) {
    alert('Please enter a valid Philippine mobile number.');
```



```
    return;
}
```

Responsive Design Breakpoints

```
/* Mobile First Approach */
@media (max-width: 768px) {
  .main-container {
    flex-direction: column;
  }

  .sidebar {
    width: 100%;
    height: auto;
  }

  .sidebar-nav {
    display: flex;
    overflow-x: auto;
    padding: 1rem;
  }

  .nav-item {
    margin: 0 0.5rem;
    min-width: 140px;
  }

  .nav-link {
    flex-direction: column;
    padding: 1rem;
    text-align: center;
  }

  .nav-link i {
    margin: 0 0 0.5rem 0;
  }

  .dashboard-content {
    padding: 1rem;
  }

  .top-navbar {
    flex-direction: column;
    align-items: flex-start;
    gap: 1rem;
  }
}

@media (max-width: 576px) {
  .content-section {
    padding: 1.5rem;
  }

  .section-title {
    font-size: 1.3rem;
  }
}
```

📄 JAVASCRIPT PATTERNS

Session Check (Auto Logout)

```
// Check on page load
window.onload = function() {
    <?php if (!isset($_SESSION['user_id'])): ?>
        window.location.href = '../login.php';
    <?php endif; ?>
};

// Auto-refresh session check every 5 minutes
setInterval(function() {
    fetch('CurrentPage.php')
        .then(response => response.text())
        .then(data => {
            if (data.includes('Location: ../login.php')) {
                window.location.href = '../login.php';
            }
        })
        .catch(error => {
            console.log('Session check failed:', error);
        });
}, 300000); // 5 minutes = 300000ms
```

AJAX Form Submission Pattern

```

document.getElementById('formId').addEventListener('submit', function(e) {
    e.preventDefault();

    // Get form values
    const field1 = document.getElementById('field1').value;
    const field2 = document.getElementById('field2').value;

    // Validate
    if (!field1 || !field2) {
        alert('Please fill in all required fields.');
```

```

        return;
    }

    // Disable button
    const submitBtn = document.getElementById('submitBtn');
    submitBtn.disabled = true;
    submitBtn.innerHTML = '<i class="fas fa-spinner fa-spin me-2"></i>Processing...';

    // Prepare data
    const formData = new FormData();
    formData.append('action', 'action_name');
    formData.append('field1', field1);
    formData.append('field2', field2);

    // Submit
    fetch('CurrentPage.php', {
        method: 'POST',
        body: formData
    })
    .then(response => response.json())
    .then(data => {
        if (data.success) {
            alert(data.message);
            location.reload(); // or update UI
        } else {
            alert(data.message || 'Error occurred');
```

```

        }
    })
    .catch(error => {
        console.error('Error:', error);
        alert('An error occurred. Please try again.');
```

```

    })
    .finally(() => {
        submitBtn.disabled = false;
        submitBtn.innerHTML = 'Submit';
    });
});

```

Modal Management


```

// Show modal dynamically
function showModal(title, content) {
  const modalHTML = `
    <div class="modal fade" id="dynamicModal" tabindex="-1">
      <div class="modal-dialog">
        <div class="modal-content">
          <div class="modal-header">
            <h5 class="modal-title">${title}</h5>
            <button type="button" class="btn-close" data-bs-dismiss="modal"></button>
          </div>
          <div class="modal-body">${content}</div>
          <div class="modal-footer">
            <button type="button" class="btn btn-secondary" data-bs-dismiss="modal">Close</button>
          </div>
        </div>
      </div>
    </div>
  `;

  // Remove existing modal
  const existingModal = document.getElementById('dynamicModal');
  if (existingModal) {
    existingModal.remove();
  }

  // Add new modal
  document.body.insertAdjacentHTML('beforeend', modalHTML);

  // Show modal
  const modal = new bootstrap.Modal(document.getElementById('dynamicModal'));
  modal.show();
}

```

Date Handling

```

// Get tomorrow's date (for min date in appointments)
const tomorrow = new Date();
tomorrow.setDate(tomorrow.getDate() + 1);
const tomorrowStr = tomorrow.toISOString().split('T')[0];

// Set min date attribute
document.getElementById('appointment_date').setAttribute('min', tomorrowStr);

// Format date for display
function formatDate(dateString) {
  const date = new Date(dateString);
  const options = {
    year: 'numeric',
    month: 'long',
    day: 'numeric',
    hour: 'numeric',
    minute: '2-digit',
    hour12: true
  };
  return date.toLocaleDateString('en-US', options);
}

```

Filter/Search Functions

```

// Filter items by status
function filterItems(tabId, status) {
  const items = document.querySelectorAll(`#${tabId} .record-card`);

  items.forEach(item => {
    if (status === 'all') {
      item.style.display = '';
    } else {
      const itemStatus = item.dataset.status;
      item.style.display = itemStatus === status ? '' : 'none';
    }
  });

  // Update active filter button
  const filterButtons = document.querySelectorAll(`#${tabId}-filters .filter-btn`);
  filterButtons.forEach(btn => {
    btn.classList.remove('active');
    if (btn.dataset.filter === status) {
      btn.classList.add('active');
    }
  });

  // Show empty state if no results
  const visibleItems = Array.from(items).filter(item => item.style.display !== 'none');
  const emptyState = document.querySelector(`#${tabId} .empty-filter-state`);

  if (visibleItems.length === 0 && emptyState) {
    emptyState.style.display = 'block';
  } else if (emptyState) {
    emptyState.style.display = 'none';
  }
}

```

Confirmation Dialogs

```

// Simple confirmation
if (!confirm('Are you sure you want to cancel this appointment?')) {
  return;
}

// Prompt for reason
const reason = prompt('Please provide a reason for cancellation (optional):');
if (reason === null) {
  return; // User clicked cancel
}

// Use reason in form submission
formData.append('cancellation_reason', reason || 'Patient requested cancellation');

```

🔒 ADMIN DEVELOPMENT GUIDELINES

General Principles

1. Follow Patient-Side Patterns

- Use same authentication structure
- Maintain consistent UI/UX
- Follow same database patterns
- Use identical CSS variables and classes

2. Enhanced Permissions

- Admin can view/edit ALL records
- Admin can access patient-restricted fields

- Admin has additional actions (approve, assign, schedule)

3. Data Validation

- Server-side validation is mandatory
- Client-side validation for UX
- Input sanitization with `htmlspecialchars()`
- Prepared statements for all queries

4. Audit Trail

- Log all admin actions to `system_logs`
- Include old and new values for updates
- Track who performed the action

Role-Based Dashboard Content

RHU Admin Should See:

- Total patients registered
- Today's walk-in consultations
- Pending online consultation requests
- Urgent referrals
- Lab results pending verification
- Medical certificate requests
- Low stock medicines alert
- Appointment confirmations needed

BHS Admin Should See:

- Patients from assigned barangay only
- BHS appointments for their barangay
- Maternal care schedule
- Immunization schedule
- Pending referrals from BHS
- Consultation history for barangay

Pharmacy Admin Should See:

- Pending prescriptions to dispense
- Low stock alerts
- Expired medicines
- Medicine transaction history
- Inventory reports
- Dispensing queue

Admin-Specific Queries

Get All Patients with Filters

```

$query = "
    SELECT p.*, b.barangay_name,
           YEAR(CURDATE()) - YEAR(p.date_of_birth) as age,
           (SELECT COUNT(*) FROM consultations WHERE patient_id = p.id) as total_consultations,
           (SELECT MAX(consultation_date) FROM consultations WHERE patient_id = p.id) as last_visit
    FROM patients p
    LEFT JOIN barangays b ON p.barangay_id = b.id
    WHERE 1=1
";

// Add filters dynamically
if (!empty($search)) {
    $query .= " AND (p.first_name LIKE ? OR p.last_name LIKE ? OR p.patient_id LIKE ?)";
}

if (!empty($barangay_filter)) {
    $query .= " AND p.barangay_id = ?";
}

if (!empty($age_from)) {
    $query .= " AND YEAR(CURDATE()) - YEAR(p.date_of_birth) >= ?";
}

$query .= " ORDER BY p.created_at DESC LIMIT ? OFFSET ?";

```

Get Pending Appointments for Scheduling

```

SELECT a.*, p.first_name, p.last_name, p.phone, p.email,
       st.service_name, st.duration_minutes,
       b.barangay_name
FROM appointments a
JOIN patients p ON a.patient_id = p.id
JOIN service_types st ON a.service_type_id = st.id
LEFT JOIN barangays b ON a.barangay_id = b.id
WHERE a.status = 'pending'
AND a.appointment_time = '00:00:00'
AND a.appointment_location = 'BHS'
ORDER BY a.created_at ASC

```

Get Today's Walk-in Queue

```

SELECT c.*, p.first_name, p.last_name, p.date_of_birth,
       TIMESTAMPDIFF(MINUTE, c.created_at, NOW()) as waiting_time
FROM consultations c
JOIN patients p ON c.patient_id = p.id
WHERE c.consultation_type = 'walk-in'
AND DATE(c.created_at) = CURDATE()
AND c.status IN ('pending', 'in_progress')
ORDER BY c.priority DESC, c.created_at ASC

```

Admin Action Patterns

Approve/Confirm Appointment

```

// POST: confirm_appointment
$stmt = $pdo->prepare("
    UPDATE appointments
    SET status = 'confirmed',
        appointment_time = ?,
        assigned_staff = ?,
        confirmed_by = ?,
        confirmed_at = NOW()
    WHERE id = ?
");
$stmt->execute([$assigned_time, $staff_id, $_SESSION['user_id'], $appointment_id]);

// Notify patient
$notifStmt = $pdo->prepare("
    INSERT INTO notifications (user_id, type, title, message, data, priority)
    VALUES (
        (SELECT user_id FROM patients WHERE id = ?),
        'appointment_reminder',
        'Appointment Confirmed',
        'Your appointment has been confirmed for ? at ?. Please arrive 15 minutes early.',
        ?,
        'high'
    )
");

```

Process Consultation

```

// Complete walk-in consultation
$stmt = $pdo->prepare("
    UPDATE consultations
    SET status = 'completed',
        chief_complaint = ?,
        symptoms = ?,
        vital_signs = ?,
        physical_examination = ?,
        diagnosis = ?,
        treatment_plan = ?,
        medications_prescribed = ?,
        recommendations = ?,
        assigned_doctor = ?,
        consultation_date = NOW()
    WHERE id = ?
");

// Create prescriptions if needed
foreach ($prescriptions as $rx) {
    $rxStmt = $pdo->prepare("
        INSERT INTO prescriptions (
            prescription_number, patient_id, consultation_id,
            medication_name, dosage_strength, dosage_form,
            quantity_prescribed, dosage_instructions, frequency,
            duration, prescribed_by, prescription_date, status
        ) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, CURDATE(), 'pending')
    ");
    $rxStmt->execute([/* ... */]);
}

```

Issue Medical Certificate

```

// Update certificate with exam results
$stmt = $pdo->prepare("
    UPDATE medical_certificates
    SET status = 'issued',
        consultation_id = ?,
        diagnosis = ?,
        physical_findings = ?,
        recommendations = ?,
        restrictions = ?,
        fitness_status = ?,
        valid_from = CURDATE(),
        valid_until = DATE_ADD(CURDATE(), INTERVAL ? DAY),
        issued_by = ?,
        qr_code = ?
    WHERE id = ?
");

// Generate QR code for verification
$qr_data = json_encode([
    'cert_number' => $certificate_number,
    'patient_id' => $patient_id,
    'issued_date' => date('Y-m-d'),
    'issued_by' => $_SESSION['user_id'],
    'verify_url' => 'https://kawitrhu.gov.ph/verify/' . $certificate_number
]);

```

Multi-Action Forms

```

<!-- Admin can perform multiple actions on one page -->
<form method="post">
    <input type="hidden" name="action" value="update_consultation">
    <input type="hidden" name="consultation_id" value="<?php echo $id; ?>">

    <!-- Vital Signs -->
    <input name="temperature" placeholder="Temperature">
    <input name="blood_pressure" placeholder="BP">
    <!-- ... -->

    <!-- Diagnosis -->
    <textarea name="diagnosis"></textarea>
    <textarea name="treatment_plan"></textarea>

    <!-- Actions -->
    <button type="submit" name="action" value="save_draft">Save Draft</button>
    <button type="submit" name="action" value="complete">Complete Consultation</button>

    <!-- Quick Actions -->
    <button type="button" onclick="showPrescriptionForm()">Add Prescription</button>
    <button type="button" onclick="showLabOrderForm()">Order Lab Test</button>
    <button type="button" onclick="showCertificateForm()">Issue Certificate</button>
    <button type="button" onclick="showReferralForm()">Create Referral</button>
</form>

```

Reports & Analytics

```
// Monthly summary report
SELECT
    COUNT(DISTINCT c.id) as total_consultations,
    COUNT(DISTINCT CASE WHEN c.consultation_type = 'walk-in' THEN c.id END) as walk_in,
    COUNT(DISTINCT CASE WHEN c.consultation_type = 'online' THEN c.id END) as online,
    COUNT(DISTINCT a.id) as total_appointments,
    COUNT(DISTINCT p.id) as unique_patients,
    COUNT(DISTINCT pr.id) as prescriptions_issued,
    COUNT(DISTINCT lr.id) as lab_tests_performed,
    COUNT(DISTINCT mc.id) as certificates_issued
FROM consultations c
LEFT JOIN appointments a ON MONTH(a.appointment_date) = MONTH(CURDATE())
LEFT JOIN prescriptions pr ON MONTH(pr.prescription_date) = MONTH(CURDATE())
LEFT JOIN laboratory_results lr ON MONTH(lr.test_date) = MONTH(CURDATE())
LEFT JOIN medical_certificates mc ON MONTH(mc.date_issued) = MONTH(CURDATE())
LEFT JOIN patients p ON c.patient_id = p.id
WHERE MONTH(c.consultation_date) = MONTH(CURDATE())
AND YEAR(c.consultation_date) = YEAR(CURDATE())
```

Notifications to Patients

```

// Send notification function
function notifyPatient($patient_id, $type, $title, $message, $data = null, $priority = 'medium') {
    global $pdo;

    // Get user_id from patient_id
    $stmt = $pdo->prepare("SELECT user_id FROM patients WHERE id = ?");
    $stmt->execute([$patient_id]);
    $user_id = $stmt->fetchColumn();

    if ($user_id) {
        $notifStmt = $pdo->prepare("
            INSERT INTO notifications (user_id, type, title, message, data, priority)
            VALUES (?, ?, ?, ?, ?, ?)
        ");
        $notifStmt->execute([
            $user_id,
            $type,
            $title,
            $message,
            $data ? json_encode($data) : null,
            $priority
        ]);
    }
}

// Usage examples
notifyPatient($patient_id, 'appointment_reminder',
    'Appointment Tomorrow',
    'Reminder: You have an appointment tomorrow at 10:00 AM at BHS Binakayan.',
    ['appointment_id' => $appointment_id],
    'high'
);

notifyPatient($patient_id, 'lab_result',
    'Lab Results Ready',
    'Your laboratory results are now available. View them in your health records.',
    ['lab_id' => $lab_id],
    'medium'
);

notifyPatient($patient_id, 'prescription_ready',
    'Prescription Ready',
    'Your prescription is ready for pickup at the RHU Pharmacy.',
    ['prescription_id' => $rx_id],
    'medium'
);

```

Search Functionality


```

// Real-time search
document.getElementById('searchInput').addEventListener('input', function(e) {
    const searchTerm = e.target.value.toLowerCase();
    const items = document.querySelectorAll('.search-item');

    items.forEach(item => {
        const text = item.textContent.toLowerCase();
        item.style.display = text.includes(searchTerm) ? '' : 'none';
    });
});

// Advanced search with filters
function performSearch() {
    const formData = new FormData();
    formData.append('action', 'search_patients');
    formData.append('search_term', document.getElementById('searchTerm').value);
    formData.append('barangay', document.getElementById('barangayFilter').value);
    formData.append('age_from', document.getElementById('ageFrom').value);
    formData.append('age_to', document.getElementById('ageTo').value);
    formData.append('gender', document.getElementById('genderFilter').value);

    fetch('Patients.php', {
        method: 'POST',
        body: formData
    })
    .then(response => response.json())
    .then(data => {
        displaySearchResults(data.results);
    });
}

```

Calendar/Schedule View

```

// For appointment scheduling
function showCalendar(month, year) {
    // Fetch appointments for the month
    fetch(`Appointments.php?action=get_calendar&month=${month}&year=${year}`)
    .then(response => response.json())
    .then(data => {
        renderCalendar(data.appointments);
    });
}

function renderCalendar(appointments) {
    // Group appointments by date
    const grouped = {};
    appointments.forEach(apt => {
        const date = apt.appointment_date;
        if (!grouped[date]) grouped[date] = [];
        grouped[date].push(apt);
    });

    // Render calendar grid with appointment counts
    // Color-code by availability
}

```

UTILITY FUNCTIONS TO IMPLEMENT

Number Formatting

```

// Format number with leading zeros
function generateUniqueNumber($prefix, $table, $column, $year = null) {
    global $pdo;

    if ($year === null) {
        $year = date('Y');
    }

    $pattern = "$prefix-$year-";
    $stmt = $pdo->prepare("SELECT COUNT(*) FROM $table WHERE $column LIKE ?");
    $stmt->execute([$pattern]);
    $count = $stmt->fetchColumn();

    return $prefix . '-' . $year . '-' . str_pad($count + 1, 4, '0', STR_PAD_LEFT);
}

// Usage:
$consultation_number = generateUniqueNumber('CONS', 'consultations', 'consultation_number');
// Returns: CONS-2025-0001

$prescription_number = generateUniqueNumber('RX', 'prescriptions', 'prescription_number');
// Returns: RX-2025-0001

```

Date/Time Formatting

```

// Format date for display
function formatDate($date, $format = 'F j, Y') {
    if (!$date || $date == '0000-00-00') {
        return 'Not set';
    }
    return date($format, strtotime($date));
}

// Format datetime for display
function formatDateTime($datetime, $format = 'F j, Y g:i A') {
    if (!$datetime || $datetime == '0000-00-00 00:00:00') {
        return 'Not set';
    }
    return date($format, strtotime($datetime));
}

// Calculate age
function calculateAge($birthdate) {
    $today = new DateTime();
    $dob = new DateTime($birthdate);
    $age = $today->diff($dob)->y;
    return $age;
}

// Get time ago
function timeAgo($datetime) {
    $time = strtotime($datetime);
    $now = time();
    $diff = $now - $time;

    if ($diff < 60) {
        return 'Just now';
    } elseif ($diff < 3600) {
        $mins = floor($diff / 60);
        return $mins . ' minute' . ($mins > 1 ? 's' : '') . ' ago';
    } elseif ($diff < 86400) {
        $hours = floor($diff / 3600);
        return $hours . ' hour' . ($hours > 1 ? 's' : '') . ' ago';
    } elseif ($diff < 604800) {
        $days = floor($diff / 86400);
        return $days . ' day' . ($days > 1 ? 's' : '') . ' ago';
    } else {
        return date('F j, Y', $time);
    }
}

```

String Utilities

```

// Get initials from name
function getInitials($firstName, $lastName) {
    return strtoupper(substr($firstName, 0, 1) . substr($lastName, 0, 1));
}

// Sanitize output (use everywhere)
function safe($str) {
    return htmlspecialchars($str, ENT_QUOTES, 'UTF-8');
}

// Truncate text
function truncate($text, $length = 100, $suffix = '...') {
    if (strlen($text) <= $length) {
        return $text;
    }
    return substr($text, 0, $length) . $suffix;
}

```

Status Utilities

```

// Get status badge HTML
function statusBadge($status, $type = 'appointment') {
    $statusClass = strtolower(str_replace('_', '-', $status));
    $statusLabel = ucfirst(str_replace('_', ' ', $status));

    return '<span class="status-badge status-' . $statusClass . '">' . $statusLabel . '</span>';
}

// Get priority badge HTML
function priorityBadge($priority) {
    $colors = [
        'low' => 'success',
        'medium' => 'warning',
        'high' => 'danger',
        'urgent' => 'danger'
    ];

    $color = $colors[$priority] ?? 'secondary';
    return '<span class="badge bg-' . $color . '">' . ucfirst($priority) . '</span>';
}

```

Validation Functions

```
// Validate Philippine mobile number
function isValidPhilippineMobile($phone) {
    $phone = preg_replace('/\D/', '', $phone);
    return preg_match('/^09\d{9}$/', $phone);
}

// Validate email
function isValidEmail($email) {
    return filter_var($email, FILTER_VALIDATE_EMAIL) !== false;
}

// Validate PhilHealth number
function isValidPhilHealth($number) {
    $number = preg_replace('/\D/', '', $number);
    return strlen($number) == 12;
}

// Validate date is future
function isFutureDate($date) {
    return strtotime($date) >= strtotime('today');
}
```

🗄 DATABASE TRANSACTION PATTERNS

Safe Transaction Wrapper

```

function executeTransaction($callback) {
    global $pdo;

    try {
        $pdo->beginTransaction();
        $result = $callback($pdo);
        $pdo->commit();
        return ['success' => true, 'data' => $result];
    } catch (Exception $e) {
        $pdo->rollBack();
        error_log('Transaction failed: ' . $e->getMessage());
        return ['success' => false, 'error' => $e->getMessage()];
    }
}

// Usage:
$result = executeTransaction(function($pdo) use ($patient_id, $prescriptions) {
    // Insert consultation
    $stmt = $pdo->prepare("INSERT INTO consultations (...) VALUES (...)");
    $stmt->execute(...);
    $consultation_id = $pdo->lastInsertId();

    // Insert prescriptions
    foreach ($prescriptions as $rx) {
        $stmt = $pdo->prepare("INSERT INTO prescriptions (...) VALUES (...)");
        $stmt->execute(...);
    }

    // Create notification
    $stmt = $pdo->prepare("INSERT INTO notifications (...) VALUES (...)");
    $stmt->execute(...);

    return $consultation_id;
});

if ($result['success']) {
    echo "Consultation created with ID: " . $result['data'];
} else {
    echo "Error: " . $result['error'];
}

```

🔒 SECURITY CHECKLIST

Input Validation

- 🔒 All user inputs validated server-side
- 🔒 Use prepared statements (PDO) - NEVER concatenate SQL
- 🔒 Sanitize output with `htmlspecialchars()`
- 🔒 Validate data types (int, date, email, phone)
- 🔒 Check for required fields
- 🔒 Validate file uploads (if applicable)

Authentication & Authorization

- 🔒 Session validation on every page
- 🔒 Role-based access control
- 🔒 Check record ownership (patient can only access own records)
- 🔒 Password hashing with `password_hash()`
- 🔒 Password verification with `password_verify()`
- 🔒 Session timeout handling

SQL Injection Prevention

```
// ❌ NEVER DO THIS
$query = "SELECT * FROM patients WHERE id = " . $_POST['id'];

// ✅ ALWAYS DO THIS
$stmt = $pdo->prepare("SELECT * FROM patients WHERE id = ?");
$stmt->execute([$_POST['id']]);
```

XSS Prevention

```
// ❌ NEVER DO THIS
echo $_POST['name'];
echo $patient['address'];

// ✅ ALWAYS DO THIS
echo htmlspecialchars($_POST['name']);
echo htmlspecialchars($patient['address']);
```

CSRF Protection

```
// Generate token (login page)
$_SESSION['csrf_token'] = bin2hex(random_bytes(32));

// Include in forms
<input type="hidden" name="csrf_token" value="<?php echo $_SESSION['csrf_token']; ?>">

// Validate on submission
if (!isset($_POST['csrf_token']) || $_POST['csrf_token'] !== $_SESSION['csrf_token']) {
    die('CSRF token validation failed');
}
```

📖 CODE STANDARDS & CONVENTIONS

Naming Conventions

Variables

```
// Use snake_case
$patient_id
$appointment_date
$consultation_number
$is_active
```

Functions

```
// Use camelCase
function calculateAge($birthdate) {}
function formatDateTime($datetime) {}
function getPatientById($id) {}
```

Database Tables

```
// Use snake_case, plural for tables
patients
appointments
consultations
medical_certificates
```

Database Columns

```
// Use snake_case
first_name
appointment_date
created_at
is_active
```

CSS Classes

```
// Use kebab-case
.content-section
.status-badge
.nav-link
.btn-primary
```

File Organization


```

<?php
// 1. Session & Security
session_start();
if (!isset($_SESSION['user_id']) || $_SESSION['role'] !== 'patient') {
    header('Location: ../login.php');
    exit;
}

// 2. Database Connection
$host = 'localhost';
$dbname = 'kawit_rhu';
$username = 'root';
$password = '';

try {
    $pdo = new PDO("mysql:host=$host;dbname=$dbname", $username, $password);
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
} catch(PDOException $e) {
    die("Connection failed: " . $e->getMessage());
}

// 3. Get Patient Information
$stmt = $pdo->prepare("SELECT ... FROM patients ...");
$stmt->execute([$_SESSION['user_id']]);
$patient = $stmt->fetch(PDO::FETCH_ASSOC);

// 4. Handle POST Actions
if (isset($_POST['action'])) {
    if ($_POST['action'] == 'action_name') {
        // Handle action
        // Return JSON for AJAX
        exit;
    }
}

// 5. Get Page Data
$stmt = $pdo->prepare("SELECT ... FROM ...");
$stmt->execute([...]);
$data = $stmt->fetchAll(PDO::FETCH_ASSOC);

?>
<!DOCTYPE html>
<html>
<head>
    <!-- 6. HTML Head -->
</head>
<body>
    <!-- 7. HTML Body -->
</body>
</html>

```

Comments

```

// Use comments for complex logic
// Get appointments separated by status
$upcomingAppointments = [];
$pastAppointments = [];

foreach ($allAppointments as $appointment) {
    // Completed, cancelled, or no-show always go to past
    if ($appointment['status'] == 'completed' ||
        $appointment['status'] == 'cancelled' ||
        $appointment['status'] == 'no-show') {
        $pastAppointments[] = $appointment;
    }
    // Future dates with active status go to upcoming
    else if ($appointment['appointment_date'] >= $today) {
        $upcomingAppointments[] = $appointment;
    }
    // Past dates that weren't marked completed
    else {
        $pastAppointments[] = $appointment;
    }
}

```

Error Handling

```

try {
    // Database operations
    $stmt = $pdo->prepare("...");
    $stmt->execute(...);

    // Log action
    $logStmt = $pdo->prepare("INSERT INTO system_logs ...");
    $logStmt->execute(...);

    echo json_encode(['success' => true, 'message' => 'Success message']);
} catch (PDOException $e) {
    // Log error (don't expose to user)
    error_log('Database error: ' . $e->getMessage());

    // User-friendly message
    echo json_encode(['success' => false, 'message' => 'An error occurred. Please try again.']);
} catch (Exception $e) {
    error_log('General error: ' . $e->getMessage());
    echo json_encode(['success' => false, 'message' => 'An unexpected error occurred.']);
}

```

🔧 TESTING CHECKLIST

Functional Testing

Patient Modules

- ☐ Can login successfully
- ☐ Can view dashboard with correct data
- ☐ Can book BHS appointments
- ☐ Cannot book RHU appointments (blocked)
- ☐ Can request online consultations
- ☐ Can cancel pending appointments
- ☐ Can cancel pending consultations
- ☐ Can view health records (all tabs)

- ☐ Can filter health records by status
- ☐ Can request medical certificates
- ☐ Can download issued certificates
- ☐ Can update profile (editable fields only)
- ☐ Cannot edit read-only fields
- ☐ Can change password
- ☐ Session expires after timeout
- ☐ Can logout successfully

Admin Modules (To Build)

- ☐ Can view all patients
- ☐ Can search/filter patients
- ☐ Can add new patients
- ☐ Can edit patient information
- ☐ Can view pending appointments
- ☐ Can confirm/schedule appointments
- ☐ Can view pending consultation requests
- ☐ Can schedule online consultations
- ☐ Can create Google Meet links
- ☐ Can complete consultations
- ☐ Can create prescriptions
- ☐ Can order lab tests
- ☐ Can issue medical certificates
- ☐ Can create referrals
- ☐ Can view reports
- ☐ Can manage inventory (pharmacy)

Security Testing

- ☐ SQL injection attempts blocked
- ☐ XSS attempts sanitized
- ☐ Unauthorized access blocked
- ☐ Session hijacking prevented
- ☐ Role-based access enforced
- ☐ Password complexity enforced
- ☐ Passwords properly hashed
- ☐ CSRF tokens validated


UI/UX Testing

- ☐ Responsive on mobile (< 768px)
- ☐ Responsive on tablet (768px - 1024px)
- ☐ Responsive on desktop (> 1024px)
- ☐ All buttons work
- ☐ All forms validate
- ☐ Loading states display
- ☐ Success messages show
- ☐ Error messages clear
- ☐ Empty states friendly
- ☐ Status badges correct colors
- ☐ Icons display correctly

☒ DEPLOYMENT CHECKLIST

Pre-Deployment

- ☐ Test all functionality
- ☐ Check all queries for performance
- ☐ Add indexes to frequently queried columns
- ☐ Validate all user inputs
- ☐ Sanitize all outputs
- ☐ Remove debug code
- ☐ Remove console.log statements
- ☐ Configure error reporting (production mode)
- ☐ Set up database backups

-  Document admin credentials

Configuration

```
// config.php (create this file)
<?php
// Database Configuration
define('DB_HOST', 'localhost');
define('DB_NAME', 'kawit_rhu');
define('DB_USER', 'root');
define('DB_PASS', '');

// Application Settings
define('APP_NAME', 'Kawit RHU Health Information System');
define('APP_VERSION', '1.0.0');
define('TIMEZONE', 'Asia/Manila');

// Security Settings
define('SESSION_TIMEOUT', 1800); // 30 minutes
define('PASSWORD_MIN_LENGTH', 6);
define('MAX_LOGIN_ATTEMPTS', 5);

// File Upload Settings
define('UPLOAD_MAX_SIZE', 5242880); // 5MB
define('ALLOWED_FILE_TYPES', ['jpg', 'jpeg', 'png', 'pdf']);

// Email Settings (for notifications)
define('SMTP_HOST', 'smtp.gmail.com');
define('SMTP_PORT', 587);
define('SMTP_USER', 'your-email@gmail.com');
define('SMTP_PASS', 'your-password');

// Set timezone
date_default_timezone_set(TIMEZONE);

// Error reporting
if (APP_ENV == 'production') {
    error_reporting(0);
    ini_set('display_errors', 0);
} else {
    error_reporting(E_ALL);
    ini_set('display_errors', 1);
}
?>
```

Production Settings

```
// .htaccess (for Apache)
# Prevent directory listing
Options -Indexes

# Prevent access to sensitive files
<FilesMatch "^\. ">
    Order allow,deny
    Deny from all
</FilesMatch>

# Force HTTPS (if SSL available)
RewriteEngine On
RewriteCond %{HTTPS} off
RewriteRule ^(.*)$ https://%{HTTP_HOST}%{REQUEST_URI} [L,R=301]

# PHP settings
php_value upload_max_filesize 5M
php_value post_max_size 5M
php_value max_execution_time 300
php_value max_input_time 300
```

🔮 FUTURE ENHANCEMENTS

Phase 2 Features (After Basic Admin is Complete)

1. SMS Notifications

- Appointment reminders
- Lab result alerts
- Prescription ready notifications

2. Email Integration

- Send consultation summaries
- Email medical certificates
- Automated reports

3. Advanced Analytics

- Patient demographics dashboard
- Disease surveillance
- Service utilization trends
- Predictive analytics

4. Mobile App

- Native iOS/Android apps
- Push notifications
- Offline mode

5. Telemedicine Enhancement

- Built-in video calling (not just Google Meet)
- Chat functionality
- File sharing during consultation

6. Inventory Management

- Automated reorder alerts
- Expiry tracking
- Supplier management

7. Queue Management

- Digital queue numbers
- Waiting time estimates
- SMS/display board integration

8. Document Management

- Scan and upload physical documents
- OCR for old records
- Cloud backup

9. Patient Portal Enhancement

- Family account linking
- Dependent management

- Health tracking (weight, BP, sugar)

10. Integration with DOH Systems

- eHIMS reporting
- PhilHealth connectivity
- National immunization registry

🔧 SUPPORT & MAINTENANCE

System Logs Monitoring

```
-- Check recent errors
SELECT * FROM system_logs
WHERE log_level IN ('ERROR', 'CRITICAL')
AND created_at >= DATE_SUB(NOW(), INTERVAL 24 HOUR)
ORDER BY created_at DESC;

-- Check login attempts
SELECT user_id, COUNT(*) as attempts, MAX(created_at) as last_attempt
FROM system_logs
WHERE action = 'LOGIN_FAILED'
AND created_at >= DATE_SUB(NOW(), INTERVAL 1 HOUR)
GROUP BY user_id
HAVING attempts > 3;

-- Monitor appointment booking patterns
SELECT DATE(created_at) as date, COUNT(*) as bookings
FROM appointments
WHERE created_at >= DATE_SUB(NOW(), INTERVAL 30 DAY)
GROUP BY DATE(created_at)
ORDER BY date DESC;
```

Database Maintenance

```
-- Archive old logs (monthly)
DELETE FROM system_logs
WHERE created_at < DATE_SUB(NOW(), INTERVAL 6 MONTH);

-- Clean expired notifications
DELETE FROM notifications
WHERE is_read = 1
AND created_at < DATE_SUB(NOW(), INTERVAL 3 MONTH);

-- Optimize tables
OPTIMIZE TABLE appointments;
OPTIMIZE TABLE consultations;
OPTIMIZE TABLE system_logs;
```

Backup Strategy

```
# Daily backup script
#!/bin/bash
DATE=$(date +%Y%m%d_%H%M%S)
BACKUP_DIR="/backups/kawit_rhu"
DB_NAME="kawit_rhu"

# Create backup
mysqldump -u root -p $DB_NAME > $BACKUP_DIR/backup_$DATE.sql

# Compress
gzip $BACKUP_DIR/backup_$DATE.sql

# Keep only last 30 days
find $BACKUP_DIR -name "backup_*.sql.gz" -mtime +30 -delete

# Upload to cloud (optional)
# aws s3 cp $BACKUP_DIR/backup_$DATE.sql.gz s3://your-bucket/
```

📖 TRAINING DOCUMENTATION

For Admin Users

RHU Admin Training Topics

- 1. Patient registration and management
- 2. Walk-in consultation processing
- 3. Online consultation scheduling
- 4. Laboratory result encoding
- 5. Medical certificate issuance
- 6. Prescription management
- 7. Referral creation
- 8. Report generation

BHS Admin Training Topics

- 1. BHS appointment management
- 2. Maternal care scheduling
- 3. Immunization tracking
- 4. Basic consultation recording
- 5. Referral to RHU process

Pharmacy Admin Training Topics

- 1. Prescription dispensing
- 2. Inventory management
- 3. Stock monitoring and reordering
- 4. Expiry tracking
- 5. Transaction recording

Quick Reference Cards

QUICK REFERENCE: APPOINTMENT

1. Go to Appointments > Pending

2. Review patient details

3. Check calendar for available slots

4. Click "Confirm & Schedule"

5. Select time slot

6. Assign staff (if needed)

7. Click "Confirm"

8. Patient receives notification

📋 FINAL CHECKLIST FOR ADMIN DEVELOPMENT

Setup Phase

- ☐ Copy patient-side authentication pattern
- ☐ Copy database connection pattern
- ☐ Copy CSS variables and base styles
- ☐ Copy sidebar/navbar structure
- ☐ Set up role-based access control

Core Functionality

- ☐ Dashboard with admin metrics
- ☐ Patient management (CRUD)
- ☐ Appointment management
- ☐ Consultation processing
- ☐ Laboratory result management
- ☐ Prescription management
- ☐ Certificate issuance
- ☐ Referral management

Advanced Features

- ☐ Search and filter functionality
- ☐ Calendar/schedule view
- ☐ Reports generation
- ☐ Notification system
- ☐ Audit logging
- ☐ Inventory management (pharmacy)

Testing & Deployment

- ☐ Functional testing completed
- ☐ Security testing passed
- ☐ UI/UX testing on all devices
- ☐ Performance testing
- ☐ User acceptance testing
- ☐ Training materials prepared
- ☐ Deployment checklist completed

📌 CONCLUSION

This documentation provides a complete reference for building the admin side of the Kawit RHU system. Key principles to follow:

1. **Consistency:** Use the same patterns, styles, and structures as the patient side
2. **Security:** Validate all inputs, sanitize all outputs, use prepared statements
3. **Usability:** Clear UI, helpful messages, intuitive workflows
4. **Maintainability:** Clean code, proper comments, logical organization
5. **Scalability:** Efficient queries, proper indexing, modular design

By following these patterns and guidelines, you'll create an admin system that seamlessly integrates with the patient side while providing powerful management capabilities for healthcare staff.

Document Version: 1.0.0

Last Updated: October 16, 2025

For Questions: Contact System Administrator

Good luck with your admin development! 🍀