VHMS Developer Guide

Table of Contents

---

1. Architecture Overview

Technology Stack

- **Backend**: PHP
- **Frontend**: HTML, CSS, JavaScript (jQuery)
- **Database**: MySQL/MariaDB
- **DataTables**: jQuery DataTables plugin
- **AJAX**: Asynchronous operations

Project Structure

```
VHMS/
├── addons/
│   └── helpers/
│       └── patient-unifier.php
├── lab/
│   ├── api/
│   │   └── lab/
│   │       ├── results/
│   │       │   ├── update.php
│   │       │   └── delete.php
│   │       └── tests/
│   │           ├── create.php
│   │           ├── delete.php
│   │           ├── update.php
│   │           └── read.php
│   ├── js/
│   │   └── results-enhanced.js
│   ├── ajax_update_result.php
│   ├── results.php
│   ├── receptionquery.php
│   ├── dashboard-3.php
│   ├── allpatientform.php
│   ├── results.php
│   ├── receptionquery.php
│   └── detailform.php
└── [other files]
```

---

2. Installation & Setup

Prerequisites

```
- PHP 7.4 or higher
- MySQL 5.7 or higher
- Apache/Nginx web server
- Composer (for dependency management)
```

Installation Steps

1. Clone Repository

```
git clone https://github.com/rescueteam0005/priver3
cd priver3
```

2. Configure Database

```
// config/database.php
define('DB_HOST', 'localhost');
define('DB_NAME', 'vhms_db');
define('DB_USER', 'your_username');
define('DB_PASS', 'your_password');
```

3. Import Database Schema

```
mysql -u username -p vhms_db < database/schema.sql
```

4. Set Permissions

```
chmod 755 addons/helpers/
chmod 644 config/*.php
```

---

3. Database Structure

Key Tables

patients

```
CREATE TABLE patients (
    id INT PRIMARY KEY AUTO_INCREMENT,
    patient_id VARCHAR(50) UNIQUE,
    first_name VARCHAR(100),
    last_name VARCHAR(100),
    date_of_birth DATE,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP
);
```

lab_tests

```
CREATE TABLE lab_tests (
    id INT PRIMARY KEY AUTO_INCREMENT,
    lab_code VARCHAR(50) UNIQUE,
    patient_id VARCHAR(50),
    requesting_doctor VARCHAR(200),
    clinical_summary TEXT,
    lmp DATE NULL,
    payment_status ENUM('PAID', 'NOT PAID', 'REJECTED'),
    test_status ENUM('PENDING', 'IN_PROGRESS', 'COMPLETED',
'REJECTED'),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (patient_id) REFERENCES patients(patient_id)
);
```

lab_results

```
CREATE TABLE lab_results (
    id INT PRIMARY KEY AUTO_INCREMENT,
    lab_code VARCHAR(50),
    test_name VARCHAR(200),
    result_value VARCHAR(500),
    normal_range VARCHAR(100),
    result_date TIMESTAMP,
    verified_by VARCHAR(200),
    FOREIGN KEY (lab_code) REFERENCES lab_tests(lab_code)
);
```

---

4. API Documentation

Update Results API

Endpoint: api/lab/results/update.php

Method: POST

Request Payload:

```
{
  "resultId": 73933,
  "testId": 44643,
  "patientId": 90593,
  "pdate": "2024-11-17",
  "measureId": 21343,
  "columnType": "'result'|'remark'",
  "newValue": "7.5 | Negative"
}
```

Success Response:

```
{
  "success": true,
  "message": "Result updated successfully",
  "data": {
    "newValue": "7.5",
    "columnType": "result",
    "timestamp": "2024-11-17 14:30:00"
  }
}
```

Error Response:

```
{
  "success": false,
  "message": "Missing required fields"
}
```

Implementation:

```php
<?php
// api/lab/results/update.php
header('Content-Type: application/json');

$data = json_decode(file_get_contents('php://input'), true);
$lab_code = $data['lab_code'];
$results = $data['results'];

// Bulk update logic
$updated = 0;
foreach ($results as $result) {
    $stmt = $pdo->prepare("
        UPDATE lab_results
        SET result_value = ?, normal_range = ?
        WHERE id = ? AND lab_code = ?
    ");
    $stmt->execute([
        $result['result_value'],
        $result['normal_range'],
        $result['id'],
        $lab_code
    ]);
    $updated += $stmt->rowCount();
}

echo json_encode([
    'success' => true,
    'updated_count' => $updated
]);
```

Delete Results API

Endpoint: api/lab/results/delete.php

Method: POST

Request Payload:

```
{
  "items": [
    {
      "resultId": 501,
      "testId": 2123,
      "patientId": 14553,
      "pdate": "2023-11-15",
      "measureId": 43455
    },
    {
      "resultId": 435,
      "testId": 5840,
      "patientId": 10023,
      "pdate": "2025-06-22",
      "measureId": 38590
    },
    {
      "resultId": 113,
      "testId": 9093,
      "patientId": 29087,
      "pdate": "2024-16-22",
      "measureId": 90043
    }
  ]
}
```

Success Response:

```
{
  "success": true,
  "deleted": 3,
  "message": "Successfully deleted 2 result(s) and removed 0
orphaned test order(s)",
  "testOrdersDeleted": 0,
  "warnings": ["Item #1: No matching result found to delete"],
  "errors": ["Item #0: Database execute error: Cannot delete
protected record"]
}
```

Implementation:

```php
<?php
// api/lab/results/delete.php
header('Content-Type: application/json');

$data = json_decode(file_get_contents('php://input'), true);
$ids = $data['ids'];

$placeholders = implode(',', array_fill(0, count($ids), '?'));
$stmt = $pdo->prepare("DELETE FROM lab_results WHERE id IN
($placeholders)");
$stmt->execute($ids);

echo json_encode([
  'success' => true,
  'deleted_count' => $stmt->rowCount()
]);
```

---

5. Frontend Components

DataTable Implementation

Initialization:

```
// lab/results.php
$(document).ready(function () {
  const resultsTable = $("#resultsTable").DataTable({
    order: [[0, "desc"]], // Latest first
    ajax: {
      url: "api/lab/results/fetch.php",
      dataSrc: "data",
    },
    columns: [
      { data: "lab_code" },
      { data: "patient_name" },
      { data: "requesting_doctor" },
      { data: "test_status" },
      {
        data: "payment_status",
        render: function (data) {
          const colors = {
            "NOT PAID": "red",
            PAID: "green",
            REJECTED: "grey",
          };
          return `<span style="color:
${colors[data]}">${data}</span>`;
        },
      },
    ],
  });
});
```

Dynamic Button Generation

Payment & Test Status Dependent Buttons:

```
function generateActionButtons(paymentStatus, testStatus) {
    let buttons = '';

    if (paymentStatus === 'PAID') {
        if (testStatus === 'PENDING') {
            buttons += '<button class="btn-
verify">Verify</button>';
        } else if (testStatus === 'IN_PROGRESS') {
            buttons += '<button class="btn-
validate">Validate</button>';
        } else if (testStatus === 'COMPLETED') {
            buttons += '<button class="btn-view">View
Results</button>';
            buttons += '<button class="btn-
print">Print</button>';
        }
    } else if (paymentStatus === 'NOT PAID') {
        buttons += '<button class="btn-payment"
style="background-color: red;">Process Payment</button>';
    } else if (paymentStatus === 'REJECTED') {
        buttons += '<button class="btn-info" style="background-
color: grey;" disabled>Rejected</button>';
    }

    return buttons;
}
```

AJAX Update Handler

File: ajax_update_result.php

```
function updateResult(resultId, newData) {
  $.ajax({
    url: 'ajax_update_result.php',
    method: 'POST',
    data: {
      id: resultId,
      result_value: newData.result_value,
      normal_range: newData.normal_range,
    },
    success: function (response) {
      if (response.success) {
        // Refresh DataTable
        resultsTable.ajax.reload(null, false);
        showNotification('Result updated successfully',
'success');
      }
    },
    error: function () {
      showNotification('Update failed', 'error');
    },
  });
}
```

Row Selection & Interaction

```
// Single click - highlight
$('#resultsTable tbody').on('click', 'tr', function () {
  $(this).toggleClass('selected');
});

// Double click - edit
$('#resultsTable tbody').on('dblclick', 'tr', function () {
  const data = resultsTable.row(this).data();
  openEditModal(data);
});

// Right click - context menu
$('#resultsTable tbody').on('contextmenu', 'tr', function (e) {
  e.preventDefault();
  showContextMenu(e.pageX, e.pageY);
});
```

---

6. Code Standards

PHP Coding Standards

File Structure:

```php
<?php
/**
 * File: patient-unifier.php
 * Description: Eliminates duplicate patient records
 * Author: Nzenong K. Mc Braxton Development Team
 * Date: 2024
 */

// Database connection
require_once '../config/database.php';

class PatientUnifier {
    private $pdo;

    public function __construct($pdo) {
        $this->pdo = $pdo;
    }

    /**
     * Remove duplicate patients based on patient_id
     * @return int Number of duplicates removed
     */
    public function removeDuplicates() {
        $sql = "
            DELETE p1 FROM patients p1
            INNER JOIN patients p2
            WHERE p1.id > p2.id
            AND p1.patient_id = p2.patient_id
        ";

        $stmt = $this->pdo->prepare($sql);
        $stmt->execute();

        return $stmt->rowCount();
    }
}
```

JavaScript Coding Standards

Naming Conventions:

```javascript
// Variables: camelCase
let patientName = 'John Doe';
let labCode = 'LAB2024001';

// Functions: camelCase
function fetchPatientData(patientId) {
  // Implementation
}

// Constants: UPPER_SNAKE_CASE
const API_BASE_URL = '/api/';
const MAX_RESULTS = 100;

// Classes: PascalCase
class ResultsManager {
  constructor() {
    this.table = null;
  }
}
```

Database Query Sanitization

Always use prepared statements:

```php
<?php

// ❌ WRONG - SQL Injection vulnerable
$sql = "SELECT * FROM patients WHERE id = " . $_GET['id'];

// ☑ CORRECT - Safe
$stmt = $pdo->prepare("SELECT * FROM patients WHERE id = ?");
$stmt->execute([$_GET['id']]);
```

---

7. Troubleshooting

Common Issues & Solutions

Issue 1: Lab Code Not Persisting

```php
<?php

// Problem: Lab code lost between pages
// Solution: Ensure session persistence

session_start();
$_SESSION['lab_code'] = $labCode;

// Or pass via hidden input
echo '<input type="hidden" name="lab_code" value="' .
htmlspecialchars($labCode) . '">';
```

Issue 2: DataTable Not Updating After Delete

```javascript
// Problem: Table not refreshing
// Solution: Use reload with proper parameters

// ✗ WRONG
resultsTable.ajax.reload();

// ☑ CORRECT - maintains current page
resultsTable.ajax.reload(null, false);
```

Issue 3: Selector Not Working on All Rows

```javascript
// Problem: Event only firing on first column
// ✗ WRONG
$('.tr').on('click', function() { ... });

// ☑ CORRECT
$('#resultsTable tbody').on('click', 'tr', function() { ... });
```

Debugging Tips

Enable PHP Error Display:

```php
<?php

// Development only
ini_set('display_errors', 1);
ini_set('display_startup_errors', 1);
error_reporting(E_ALL);
```

SQL Query Debugging:

```php
<?php

try {
    $stmt = $pdo->prepare($sql);
    $stmt->execute($params);
} catch (PDOException $e) {
    error_log("SQL Error: " . $e->getMessage());
    error_log("SQL Query: " . $sql);
    error_log("Parameters: " . print_r($params, true));
}
```

Performance Optimization

Database Indexing:

```sql
-- Add indexes for frequently queried columns
CREATE INDEX idx_patient_id ON patients(patient_id);
CREATE INDEX idx_lab_code ON lab_tests(lab_code);
CREATE INDEX idx_payment_status ON lab_tests(payment_status);
CREATE INDEX idx_created_at ON lab_tests(created_at DESC);
```

AJAX Request Optimization:

```javascript
// Debounce search input
let searchTimeout;
$('#searchInput').on('keyup', function () {
```

```
  clearTimeout(searchTimeout);
  searchTimeout = setTimeout(function () {
    resultsTable.search($('#searchInput').val()).draw();
  }, 300); // Wait 300ms after user stops typing
});
```

Appendix: Git Commit History Analysis

Total commits analyzed: 26
Development period: Until **November 22, 2025**
Primary developer: **Nzenong K. Mc Braxton**

Commit Categories:

- Laboratory Module: 12 commits
- UI/UX Improvements: 6 commits
- CRUD Operations: 5 commits
- Bug Fixes: 3 commits

End of Documentation

For questions or support, contact the Nzenong K. Mc Braxton Development Team