Sancturia Wildlife - MySQL Migration Guide & AI Prompt

MIGRATION OVERVIEW

This document contains the complete Sancturia Wildlife project documentation along with instructions for migrating from Firebase to MySQL + PHP backend.

AI PROMPT FOR MIGRATION

Copy this entire section to give to another AI:

TASK: Migrate Sancturia Wildlife from Firebase to MySQL + PHP

Context

I have a wildlife conservation website called "Sancturia Wildlife" currently using Firebase for authentication and database. I need to migrate the entire backend to MySQL with PHP while keeping all frontend functionality intact.

Current Stack

- **Frontend**: HTML5, CSS3, JavaScript (ES6+), Bootstrap 5.0.2
- **Backend**: Firebase Authentication, Firebase Firestore, Firebase Realtime Database
- **Current Auth**: Firebase REST API with API key

Required Stack

- **Frontend**: Same (no changes)
- **Backend**: PHP 7.4+ with MySQL 5.7+
- **New Auth**: PHP Sessions with password hashing (password_hash/password_verify)

What I Need You To Do

1. DATABASE DESIGN

Create a complete MySQL database schema with the following tables:

users table:

- user_id (INT, PRIMARY KEY, AUTO_INCREMENT)

- name (VARCHAR(100))
- email (VARCHAR(150), UNIQUE)
- password (VARCHAR(255), hashed)
- donation_total (DECIMAL(10,2), DEFAULT 0.00)
- adoptions_count (INT, DEFAULT 0)
- created_at (TIMESTAMP, DEFAULT CURRENT_TIMESTAMP)
- last_login (TIMESTAMP, NULL)

donations table:

- donation_id (INT, PRIMARY KEY, AUTO_INCREMENT)
- user_id (INT, FOREIGN KEY → users.user_id)
- amount (DECIMAL(10,2))
- sanctuary_name (VARCHAR(200))
- recurring_type (ENUM('none', 'monthly', 'yearly'))
- donation_date (TIMESTAMP, DEFAULT CURRENT_TIMESTAMP)
- payment_status (ENUM('pending', 'completed', 'failed'))

adoptions table:

- adoption_id (INT, PRIMARY KEY, AUTO_INCREMENT)
- user_id (INT, FOREIGN KEY → users.user_id)
- animal_name (VARCHAR(100))
- animal_type (VARCHAR(50))
- sanctuary_name (VARCHAR(200))
- adoption_date (TIMESTAMP, DEFAULT CURRENT_TIMESTAMP)

sanctuaries table:

- sanctuary_id (INT, PRIMARY KEY, AUTO_INCREMENT)
- name (VARCHAR(200), UNIQUE)
- location (VARCHAR(100))
- description (TEXT)

- image_path (VARCHAR(255))
- website_url (VARCHAR(255))
- created_at (TIMESTAMP, DEFAULT CURRENT_TIMESTAMP)

sessions table (for session management):

- session_id (VARCHAR(128), PRIMARY KEY)
- user_id (INT, FOREIGN KEY → users.user_id)
- session_data (TEXT)
- last_activity (TIMESTAMP)
- ip_address (VARCHAR(45))
- user_agent (VARCHAR(255))

Provide the complete SQL CREATE TABLE statements.

2. DATABASE CONNECTION FILE

Create **config/database.php** with:

- PDO connection to MySQL
- Error handling
- Connection pooling if possible
- Constants for database credentials (to be replaced with environment variables)

```php
<?php
// Example structure needed:
define('DB_HOST', 'localhost');
define('DB_NAME', 'sancturia_wildlife');
define('DB_USER', 'root');
define('DB_PASS', '');

// Create PDO connection with error handling
```

3. AUTHENTICATION SYSTEM

Replace Firebase Authentication with PHP Sessions:

A. Create includes/auth.php with functions:

- `register_user($name, $email, $password)` - Hash password, insert into users table
- `login_user($email, $password)` - Verify password, create session
- `logout_user()` - Destroy session
- `check_login()` - Verify if user is logged in
- `get_user_data($user_id)` - Fetch user information

B. Update signup.php:

- Remove all Firebase API calls
- Validate form inputs (name, email, password, confirm_password)
- Check if email already exists
- Hash password using `password_hash($password, PASSWORD_BCRYPT)`
- Insert into users table
- Create session and redirect to dashboard.php

C. Create login.php (backend):

- Validate email and password
- Query users table for matching email
- Verify password using `password_verify()`
- Create PHP session with user_id
- Redirect to dashboard.php

D. Update login.html form:

- Change form action to `login.php` (POST method)
- Keep all current form fields

4. DASHBOARD SYSTEM

Update **dashboard.php** to use MySQL instead of Firestore:

Required Changes:

- Remove all Firebase REST API calls

- Use PHP session to get logged-in user_id: `$user_id = $_SESSION['user_id'];`
- Query MySQL database for:
  - User profile (SELECT from users WHERE user_id = ?)
  - Recent donations (SELECT from donations WHERE user_id = ? ORDER BY donation_date DESC LIMIT 5)
  - Recent adoptions (SELECT from adoptions WHERE user_id = ? ORDER BY adoption_date DESC)
  - Recommended sanctuaries (SELECT from sanctuaries LIMIT 3)
- Use prepared statements for all queries to prevent SQL injection
- Keep same HTML structure and Bootstrap styling

Example structure needed:

```php
<?php
session_start();
require_once '../config/database.php';
require_once '../includes/auth.php';

// Check if logged in
if (!check_login()) {
    header("Location: ../login-signup/login.html");
    exit();
}

$user_id = $_SESSION['user_id'];

// Fetch user data
$stmt = $pdo->prepare("SELECT * FROM users WHERE user_id = ?");
$stmt->execute([$user_id]);
$user = $stmt->fetch(PDO::FETCH_ASSOC);

// Fetch donations
// Fetch adoptions
// Fetch sanctuaries
?>
```

5. DONATION PROCESSING

Create **pages/donate/process_donation.php:**

- Receive POST data from donate.html form (name, email, phone, amount, recurring)
- Validate all inputs
- Get user_id from session (if logged in) or create guest donation
- Insert into donations table
- Update users.donation_total
- Redirect to thankyou.html

Form Update: Update donate.html form action:

```
html

<form action="./process_donation.php" method="POST">
```

6. SANCTUARY DATA MIGRATION

Create **scripts/seed_sanctuaries.php:**

- Take the 52 sanctuaries from sancturies.js (Sanctuary class instances)
- Insert all into MySQL sanctuaries table
- This is a one-time migration script

Provide the complete PHP script to populate the database.

7. SESSION MANAGEMENT

Create **includes/session_handler.php:**

- Custom session handler using MySQL sessions table
- Session timeout after 30 minutes of inactivity
- Session hijacking prevention (check IP and user agent)
- Regenerate session ID on login

8. SECURITY IMPLEMENTATIONS

Implement the following security measures:

A. SQL Injection Prevention:

- Use PDO prepared statements for ALL database queries

- Never concatenate user input into SQL queries

B. XSS Prevention:

- Use `htmlspecialchars()` for all user-generated output
- Sanitize all form inputs

C. CSRF Protection:

- Create `includes/csrf.php` with token generation and validation
- Add CSRF tokens to all forms

D. Password Security:

- Minimum 8 characters
- Require at least one uppercase, one lowercase, one number
- Use `password_hash()` with PASSWORD_BCRYPT

E. Session Security:

- HttpOnly and Secure flags on session cookies
- Session regeneration on privilege escalation

9. API ENDPOINTS (PHP FILES)

Create RESTful-style PHP endpoints:

api/get_user.php - Return user data as JSON **api/get_donations.php** - Return user donations as JSON **api/get_sanctuaries.php** - Return sanctuaries list as JSON **api/search_sanctuary.php** - Search sanctuary by name

Each should:

- Check authentication
- Validate inputs
- Return JSON responses
- Handle errors properly

10. ERROR HANDLING

Create **includes/error_handler.php:**

- Custom error logging
- User-friendly error messages
- Development vs. production error display

11. TESTING DATA

Provide SQL INSERT statements for:

- 5 test users (with hashed passwords)
- 10 sample donations
- 5 sample adoptions
- All 52 sanctuaries from the original project

12. CONFIGURATION FILES

Create:

- **.htaccess** - URL rewriting, security headers
- **config/config.php** - Application-wide constants
- **.env.example** - Template for environment variables

13. MIGRATION CHECKLIST

Provide a complete checklist of:

- Files to create (with file paths)
- Files to modify (with what changes)
- Database tables to create
- Testing steps
- Deployment steps

DELIVERABLES REQUIRED

Please provide:

1. **Complete SQL schema** (CREATE TABLE statements for all 5 tables)
2. **config/database.php** (PDO connection)
3. **includes/auth.php** (all authentication functions)

4. **Updated signup.php** (MySQL version)
5. **New login.php** (backend processing)
6. **Updated dashboard.php** (MySQL queries replacing Firebase)
7. **pages/donate/process_donation.php** (donation processing)
8. **scripts/seed_sanctuaries.php** (migrate 52 sanctuaries)
9. **includes/session_handler.php** (custom session management)
10. **includes/csrf.php** (CSRF protection)
11. **includes/error_handler.php** (error handling)
12. **api/** folder files (4 API endpoints)
13. **.htaccess** (security and routing)
14. **README_MIGRATION.md** (step-by-step setup instructions)
15. **SQL test data** (INSERT statements for testing)

CODE STYLE REQUIREMENTS

- Use PDO, not mysqli
- Use prepared statements everywhere
- Follow PSR-12 coding standards
- Add comprehensive comments
- Include error handling in every function
- Use meaningful variable names
- Separate concerns (database, business logic, presentation)

IMPORTANT NOTES

- Keep ALL existing JavaScript files unchanged (home.js, sancturies.js, donate.js)
- Keep ALL existing HTML structure and CSS styling
- Only change backend from Firebase to MySQL/PHP
- Maintain localStorage functionality for donation amounts and sanctuary search
- Ensure dashboard.php displays data in exact same format as before
- All 52 sanctuaries must be preserved with same data

FRONTEND CHANGES NEEDED (Minimal)

- Update form actions from Firebase endpoints to PHP files
- Ensure AJAX calls (if any) point to new PHP API endpoints
- Add CSRF tokens to forms (hidden input fields)

TESTING REQUIREMENTS

After implementation, I should be able to:

1. Register a new user (name, email, password)
2. Login with email and password
3. View dashboard with profile, donations, adoptions
4. Browse all 52 sanctuaries
5. Search for specific sanctuary
6. Make a donation (insert into database)
7. See donation reflected in dashboard
8. Logout and login again

EXAMPLE FILE STRUCTURE NEEDED

```
sancturia-wildlife/
├── config/
│   ├── database.php (NEW)
│   └── config.php (NEW)
├── includes/
│   ├── auth.php (NEW)
│   ├── session_handler.php (NEW)
│   ├── csrf.php (NEW)
│   └── error_handler.php (NEW)
├── api/
│   ├── get_user.php (NEW)
│   ├── get_donations.php (NEW)
│   ├── get_sanctuaries.php (NEW)
│   └── search_sanctuary.php (NEW)
├── scripts/
│   └── seed_sanctuaries.php (NEW)
├── pages/
│   ├── donate/
```

```
│   │   ├── donate.html (MINOR UPDATE)
│   │   ├── donate.js (NO CHANGE)
│   │   ├── donateStyle.css (NO CHANGE)
│   │   └── process_donation.php (NEW)
│   ├── login-signup/
│   │   ├── login.html (MINOR UPDATE)
│   │   ├── signup.html (NO CHANGE)
│   │   ├── signup.php (MAJOR UPDATE)
│   │   └── login.php (NEW)
│   └── dashboard/
│       └── dashboard.php (MAJOR UPDATE)
├── .htaccess (NEW)
├── .env.example (NEW)
└── README_MIGRATION.md (NEW)
```

SANCTUARY DATA TO MIGRATE

The following 52 sanctuaries from sancturies.js must be inserted into MySQL:

1. Ranthambore National Park (Rajasthan)
2. Kaziranga National Park (Assam)
3. Jim Corbett National Park (Uttarakhand)
4. Sundarban National Park (West Bengal)
5. Bandhavgarh National Park (Madhya Pradesh)
6. Gir National Park (Gujarat)
7. Periyar Wildlife Sanctuary (Kerala)
8. Manas National Park (Assam)
9. Keoladeo National Park (Rajasthan)
10. Nagarhole National Park (Karnataka) ... [and 42 more - all from the original sancturies.js file]

Each sanctuary has: id, name, location, description, img path, website URL

END OF AI PROMPT

ORIGINAL PROJECT DOCUMENTATION

(The complete documentation from the previous artifact continues below...)

Project Overview

Sancturia Wildlife is a web-based platform dedicated to wildlife conservation in India. The platform connects compassionate donors with wildlife sanctuaries across India, enabling direct financial support for conservation efforts. Users can explore sanctuaries, make donations, and symbolically adopt animals to support their care.

Project Structure

```
Sancturia-Wildlife/
├── index.html                  # Homepage
├── homeStyle.css               # Homepage styles
├── home.js                     # Homepage JavaScript
├── Assets_TBU/
│   ├── Background Images/      # Background images & logos
│   ├── Animal Images/          # Sanctuary and animal images
│   └── Logos Icons/            # Icons and animations
├── pages/
│   ├── about/
│   │   ├── about.html          # About page
│   │   └── aboutStyle.css      # About page styles
│   ├── donate/
│   │   ├── donate.html         # Donation page
│   │   ├── donateStyle.css     # Donation styles
│   │   ├── donate.js           # Donation JavaScript
│   │   └── thankyou.html       # Thank you page
│   ├── sanctuaries/
│   │   ├── sancturies.html     # Sanctuaries listing
│   │   ├── sancturiesStyle.css # Sanctuaries styles
│   │   └── sancturies.js       # Sanctuaries JavaScript
│   ├── adopt/
│   │   └── adopt.html          # Adoption page (under construction)
│   ├── login-signup/
│   │   ├── login.html          # Login page
│   │   ├── signup.html         # Signup page
│   │   ├── signup.php          # Signup backend
│   │   └── signupStyle.css     # Login/Signup styles
│   ├── dashboard/
│   │   └── dashboard.php       # User dashboard
```

```
│   └── navbar/
│       ├── navbar.html          # Reusable navbar component
│       └── navbar.js            # Navbar loader script
```

Technology Stack

Frontend

- **HTML5** - Structure and content
- **CSS3** - Styling and animations
- **JavaScript (ES6+)** - Client-side interactivity
- **Bootstrap 5.0.2** - Responsive framework
- **Google Fonts** - Sofia font family

Backend (CURRENT - Firebase)

- **PHP** - Server-side processing
- **Firebase Authentication** - User authentication
- **Firebase Firestore** - Database storage
- **Firebase Realtime Database** - User data storage

Backend (TARGET - MySQL)

- **PHP 7.4+** - Server-side processing
- **MySQL 5.7+** - Relational database
- **PDO** - Database abstraction layer
- **PHP Sessions** - Authentication management

External Libraries

- Bootstrap 5.0.2 (CSS & JS)
- Google Fonts (Sofia family)

Core Features

1. Homepage (index.html)

Purpose: Landing page showcasing the mission and featured sanctuaries

Key Elements:

- Hero section with tagline: "Protecting Indian Wildlife Sanctuaries"
- Quick donation section with preset amounts (₹100, ₹500, ₹1000, ₹2000)
- Custom donation input field
- Recurring donation options (monthly/yearly)
- Featured sanctuary cards with images (Jim Corbett, Ranthambore, Chilika, Nal Sarovar)
- Navigation to sanctuary details via "Visit" buttons
- Jungle-themed background image

JavaScript Functionality (home.js):

```javascript
```

```
// Preset donation buttons set placeholder values
btnRs.forEach(btn => {
    btn.addEventListener('click', (e) => {
        donationPreset = e.target.innerText;
        localStorage.setItem('DonationPreset', donationPreset);
        donationAmt.placeholder = btn.innerText;
    });
});

// Sanctuary redirection stores sanctuary name in localStorage
parkBtns.forEach(btn => {
    btn.addEventListener('click', (e) => {
        sancName = e.target.name;
        localStorage.setItem('SanctuaryName', sancName);
        window.location.href = './sancturies.html';
    });
});

// Radio button toggle functionality
radioButtons.forEach(radio => {
    radio.addEventListener('click', function() {
        if (this === lastChecked) {
            this.checked = false;
            lastChecked = null;
        } else {
            lastChecked = this;
        }
    });
});
```

CSS Highlights (homeStyle.css):

- Full viewport jungle background with blur effect
- Glassmorphism donation section (`backdrop-filter: blur(1px)`)
- Custom golden-themed radio buttons
- Responsive design with breakpoints at 1199px, 991px, 872px, 768px, 576px
- Hover animations on sanctuary images

---

2. Sanctuaries Page (sancturies.html)

Purpose: Comprehensive listing of 52 Indian wildlife sanctuaries with search functionality

Key Features:

- Search bar for finding specific sanctuaries
- Dynamic grid layout (auto-fit, minmax(270px, 1fr))
- Modal popup for searched sanctuary details
- Shuffled sanctuary display on page load
- Direct links to official sanctuary websites

Sanctuary Data Structure (sancturies.js):

javascript

```
class Sanctuary {
    constructor(id, name, location, description, img, website) {
        this.id = id;
        this.name = name;
        this.location = location;
        this.description = description;
        this.img = img;
        this.website = website;
    }

    getInfo() {
        return `${this.id}, ${this.name}, ${this.location}, ${this.description}, ${this.img}, ${this.website}`;
    }
}

// 52 sanctuaries including:
// - Ranthambore National Park (Rajasthan)
// - Kaziranga National Park (Assam)
// - Jim Corbett National Park (Uttarakhand)
// - Sundarban National Park (West Bengal)
// ... and 48 more
```

Search Functionality:

javascript

```
function findSanctuaryByName(name) {
  return sanctuaries.find(sanctuary =>
    sanctuary.name.split(' ')[0].toLowerCase() === name.split(' ')[0].toLowerCase()
  );
}

function searchSanc() {
  try {
    searchedModal.style.display = 'block';
    searchedModal.innerHTML = null;
    searchVal = searchBar.value;
    searchInfo = findSanctuaryByName(searchVal).getInfo();
    searchInfo = searchInfo.split(', ');

    // Create modal card with sanctuary details
    const searchedCard = `
      <div class="sanctuary-card">
        <img src="${searchInfo[4]}" class="sanctuary-image">
        <h3>${searchInfo[1]}</h3>
        <p>Location: ${searchInfo[2]}</p>
        <p>${searchInfo[3]}</p>
        <button><a href="${searchInfo[5]}" target="_blank">Get Info Here</a></button>
        <a class="closeBtn btn btn-danger">Close</a>
      </div>
    `;

    searchedModal.innerHTML += searchedCard;
    searchedModal.animate(modalAnim, modalAnimTime);
  } catch (error) {
    alert("No Such Sanctuary in List!");
  }
}
```

Integration with Homepage:

- If redirected from homepage, automatically searches for sanctuary stored in localStorage
- Smooth scroll to top after search

---

3. Donation Page (donate.html)

Purpose: Tiered donation system with donor information collection

Donation Tiers:

1. **Tier Soil** - ₹5001+ (Leaves icon with animation)
2. **Tier Habitat** - ₹7501+ (Elephant icon with animation)
3. **Tier Earth** - ₹10001+ (Earth icon with animation)

Form Fields:

- Name (text, required)
- Email (email, required)
- Phone Number (tel, 10 digits, required)
- Donation Amount (number, required)
- Recurring options (monthly/yearly radio buttons)

JavaScript Interactions (donate.js):

```javascript
// Preset donation from homepage
let donationValue = localStorage.getItem('DonationPreset');
if(donationValue != null){
    donationPlaceholder.placeholder = donationValue;
}

// Animated tier card hover effects
tCard1.addEventListener('mouseover', ()=>{
    tVid1.loop = true;
    tVid1.play()
    tVid1.style.scale = '120%';
});

// Donation button animation
donateBtn.addEventListener('click', ()=>{
    tierIconMove() // Triggers all tier animations simultaneously
});
```

Design Features:

- Wildlife background video (wildlifeVideo.mp4)
- Glassmorphism form design

- Golden border accents
- Responsive grid layout
- Footer with social media links

---

4. About Page (about.html)

Purpose: Organization information and mission statement

Sections:

1. **Our Mission** - Platform purpose and conservation goals
2. **Our Vision** - Long-term ecological objectives
3. **Our Story** - Founded in 2024, addressing sanctuary resource limitations
4. **How We Work** - User-friendly donation process explanation
5. **Join Us** - Call-to-action for supporter engagement

Design:

- Header with green (#4CAF50) accent color
- Content in centered card (max-width: 70vw)
- Warm background (rgba(255, 244, 221, 0.897))
- HR separators between sections
- Responsive down to 600px

---

5. Authentication System (TO BE MIGRATED)

Current Implementation (Firebase)

```
php
```

```php
// signup.php (CURRENT - Firebase)
$firebase_api_key = "AIzaSyDsYiL3F1F6hcTHx3x7v9yRJMBEL-9iKGc";
$url = "https://identitytoolkit.googleapis.com/v1/accounts:signUp?key=" . $firebase_api_key;

$postData = [
    "email" => $email,
    "password" => $password,
    "returnSecureToken" => true
];

// cURL request to Firebase
```

Target Implementation (MySQL)

```php
// signup.php (TARGET - MySQL)
require_once '../config/database.php';
require_once '../includes/auth.php';

$result = register_user($name, $email, $password);
if ($result['success']) {
    $_SESSION['user_id'] = $result['user_id'];
    header("Location: ../dashboard/dashboard.php");
} else {
    $error = $result['error'];
}
```

---

6. Dashboard System (TO BE MIGRATED)

Current Implementation (Firebase Firestore)

```php
// dashboard.php (CURRENT)
$projectId = "sancturia-wildlife";
$firestoreUrl = "https://firestore.googleapis.com/v1/projects/$projectId/databases/(default)/documents";

$userDoc = fetchFromFirestore("$firestoreUrl/users/$uid");
$donationsQuery = "$firestoreUrl/donations?where=uid%3D'$uid'";
```

Target Implementation (MySQL)

```php
// dashboard.php (TARGET)
require_once '../config/database.php';

$stmt = $pdo->prepare("SELECT * FROM users WHERE user_id = ?");
$stmt->execute([$user_id]);
$user = $stmt->fetch(PDO::FETCH_ASSOC);

$stmt = $pdo->prepare("SELECT * FROM donations WHERE user_id = ? ORDER BY donation_date DESC LIMIT 5");
$stmt->execute([$user_id]);
$donations = $stmt->fetchAll(PDO::FETCH_ASSOC);
```

Database Schema (MySQL Target)

users table

```sql
CREATE TABLE users (
    user_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(100) NOT NULL,
    email VARCHAR(150) UNIQUE NOT NULL,
    password VARCHAR(255) NOT NULL,
    donation_total DECIMAL(10,2) DEFAULT 0.00,
    adoptions_count INT DEFAULT 0,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    last_login TIMESTAMP NULL,
    INDEX idx_email (email)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

donations table

```sql
```

```sql
CREATE TABLE donations (
    donation_id INT PRIMARY KEY AUTO_INCREMENT,
    user_id INT,
    donor_name VARCHAR(100) NOT NULL,
    donor_email VARCHAR(150) NOT NULL,
    donor_phone VARCHAR(15),
    amount DECIMAL(10,2) NOT NULL,
    sanctuary_name VARCHAR(200),
    recurring_type ENUM('none', 'monthly', 'yearly') DEFAULT 'none',
    donation_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    payment_status ENUM('pending', 'completed', 'failed') DEFAULT 'completed',
    FOREIGN KEY (user_id) REFERENCES users(user_id) ON DELETE SET NULL,
    INDEX idx_user_id (user_id),
    INDEX idx_donation_date (donation_date)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

adoptions table

sql

```sql
CREATE TABLE adoptions (
    adoption_id INT PRIMARY KEY AUTO_INCREMENT,
    user_id INT NOT NULL,
    animal_name VARCHAR(100) NOT NULL,
    animal_type VARCHAR(50),
    sanctuary_name VARCHAR(200),
    adoption_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES users(user_id) ON DELETE CASCADE,
    INDEX idx_user_id (user_id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

sanctuaries table

sql

```sql
CREATE TABLE sanctuaries (
    sanctuary_id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(200) UNIQUE NOT NULL,
    location VARCHAR(100) NOT NULL,
    description TEXT,
    image_path VARCHAR(255),
    website_url VARCHAR(255),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    INDEX idx_name (name)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

sessions table

sql

```sql
CREATE TABLE sessions (
    session_id VARCHAR(128) PRIMARY KEY,
    user_id INT NOT NULL,
    session_data TEXT,
    last_activity TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
    ip_address VARCHAR(45),
    user_agent VARCHAR(255),
    FOREIGN KEY (user_id) REFERENCES users(user_id) ON DELETE CASCADE,
    INDEX idx_user_id (user_id),
    INDEX idx_last_activity (last_activity)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

Migration Checklist

Phase 1: Database Setup

- [ ] Create MySQL database 'sancturia_wildlife'
- [ ] Create all 5 tables (users, donations, adoptions, sanctuaries, sessions)
- [ ] Run seed_sanctuaries.php to insert 52 sanctuaries
- [ ] Insert test user data
- [ ] Verify database connections

Phase 2: Core Files Creation

- [ ] Create config/database.php
- [ ] Create config/config.php

- [ ] Create includes/auth.php
- [ ] Create includes/session_handler.php
- [ ] Create includes/csrf.php
- [ ] Create includes/error_handler.php

Phase 3: Authentication Migration

- [ ] Update signup.php (remove Firebase, add MySQL)
- [ ] Create login.php (backend processing)
- [ ] Update login.html form action
- [ ] Test registration flow
- [ ] Test login flow

Phase 4: Dashboard Migration

- [ ] Update dashboard.php (replace Firestore with MySQL)
- [ ] Test user profile display
- [ ] Test donations display
- [ ] Test adoptions display
- [ ] Test recommended sanctuaries

Phase 5: Donation Processing

- [ ] Create process_donation.php
- [ ] Update donate.html form action
- [ ] Test donation insertion
- [ ] Test donation_total update
- [ ] Test redirect to thank you page

Phase 6: API Endpoints

- [ ] Create api/get_user.php
- [ ] Create api/get_donations.php
- [ ] Create api/get_sanctuaries.php
- [ ] Create api/search_sanctuary.php
- [ ] Test all endpoints

Phase 7: Security Implementation

- [ ] Add CSRF tokens to all forms
- [ ] Implement prepared statements everywhere
- [ ] Add input validation and sanitization

- [ ] Configure session security
- [ ] Add .htaccess security headers

Phase 8: Testing

- [ ] Test complete user journey (register → login → donate → dashboard)
- [ ] Test sanctuary search functionality
- [ ] Test recurring donation options
- [ ] Test logout and re-login
- [ ] Test error handling

Phase 9: Deployment

- [ ] Configure production database
- [ ] Update database credentials
- [ ] Set up environment variables
- [ ] Configure .htaccess
- [ ] Test on production server

Files to Create (New)

1. **config/database.php** - Database connection
2. **config/config.php** - Application constants
3. **includes/auth.php** - Authentication functions
4. **includes/session_handler.php** - Session management
5. **includes/csrf.php** - CSRF protection
6. **includes/error_handler.php** - Error handling
7. **pages/donate/process_donation.php** - Donation processing
8. **pages/login-signup/login.php** - Login backend
9. **scripts/seed_sanctuaries.php** - Sanctuary data migration
10. **api/get_user.php** - User data API
11. **api/get_donations.php** - Donations API
12. **api/get_sanctuaries.php** - Sanctuaries API
13. **api/search_sanctuary.php** - Search API

14. **.htaccess** - Security and routing
15. **.env.example** - Environment variables template
16. **README_MIGRATION.md** - Setup instructions

Files to Modify

1. **pages/login-signup/signup.php** - Remove Firebase, add MySQL
2. **pages/dashboard/dashboard.php** - Replace Firestore with MySQL
3. **pages/donate/donate.html** - Update form action
4. **pages/login-signup/login.html** - Update form action

LocalStorage Usage (No Change)

The project uses browser localStorage for cross-page data persistence:

1. **DonationPreset** - Stores selected donation amount
2. **SanctuaryName** - Stores selected sanctuary for search

Example:

```javascript
// Set on homepage
localStorage.setItem('DonationPreset', '1000');

// Retrieve on donate page
let donationValue = localStorage.getItem('DonationPreset');
donationPlaceholder.placeholder = donationValue;

// Clear storage
localStorage.clear();
```

Complete 52 Sanctuaries Data (For Migration)

This data from sancturies.js must be inserted into MySQL sanctuaries table:

```javascript
```

```
const sanctuaries = [
  {id: 1, name: 'Ranthambore National Park', location: 'Rajasthan', description: 'Famous for Bengal tigers and diverse wildlife
  {id: 2, name: 'Kaziranga National Park', location: 'Assam', description: 'Home to the Indian one-horned rhinoceros. Amazin
  {id: 3, name: 'Jim Corbett National Park', location: 'Uttarakhand', description: 'First national park in India known for its div
  {id: 4, name: 'Sundarban National Park', location: 'West Bengal', description: 'Unique mangrove ecosystem and royal Beng
  {id: 5, name: 'Bandhavgarh National Park', location: 'Madhya Pradesh', description: 'Known for large populations of tigers
  {id: 6, name: 'Gir National Park', location: 'Gujarat', description: 'Famous for Asiatic lions and diverse flora.', img: '/Assets_
  {id: 7, name: 'Periyar Wildlife Sanctuary', location: 'Kerala', description: 'Home to elephants tigers and beautiful landscapes
  {id: 8, name: 'Manas National Park', location: 'Assam', description: 'Rich biodiversity and home to endangered species.', im
  {id: 9, name: 'Keoladeo National Park', location: 'Rajasthan', description: 'Famous for birdwatching and diverse habitats.', i
  {id: 10, name: 'Nagarhole National Park', location: 'Karnataka', description: 'Known for its rich wildlife including elephants
  {id: 11, name: 'Tadoba Andhari Tiger Reserve', location: 'Maharashtra', description: 'One of the largest tiger reserves in Indi
  {id: 12, name: 'Sariska Tiger Reserve', location: 'Rajasthan', description: 'Known for its tigers and rich wildlife diversity.', i
  {id: 13, name: 'Panna National Park', location: 'Madhya Pradesh', description: 'Home to diverse wildlife including leopards
  {id: 14, name: 'Satpura National Park', location: 'Madhya Pradesh', description: 'Famous for its hilly terrain and wildlife div
  {id: 15, name: 'Kanha National Park', location: 'Madhya Pradesh', description: 'Known for its vast meadows and tiger popul
  {id: 16, name: 'Pench National Park', location: 'Madhya Pradesh', description: 'Home to diverse wildlife and scenic landsca
  {id: 17, name: 'Mudumalai Wildlife Sanctuary', location: 'Tamil Nadu', description: 'Famous for its elephants and rich biodi
  {id: 18, name: 'Bhitar Kanika Wildlife Sanctuary', location: 'Odisha', description: 'Unique estuarine ecosystem with rich wil
  {id: 19, name: 'Chinnar Wildlife Sanctuary', location: 'Kerala', description: 'Home to endangered species like the Nilgiri Tal
  {id: 20, name: 'Bhadra Wildlife Sanctuary', location: 'Karnataka', description: 'Rich in biodiversity and famous for its wildli
  {id: 21, name: 'Valley of Flowers National Park', location: 'Uttarakhand', description: 'Famous for its stunning alpine flowe
  {id: 22, name: 'Eravikulam National Park', location: 'Kerala', description: 'Home to the Nilgiri Tahr and rich biodiversity.', i
  {id: 23, name: 'Rajaji National Park', location: 'Uttarakhand', description: 'Known for its rich flora and fauna especially elep
  {id: 24, name: 'Mukurthi National Park', location: 'Tamil Nadu', description: 'Home to diverse wildlife and beautiful landsc
  {id: 25, name: 'Anamalai Tiger Reserve', location: 'Tamil Nadu', description: 'Famous for its tigers and rich flora.', img: '/As
  {id: 26, name: 'Desert National Park', location: 'Rajasthan', description: 'Unique desert ecosystem with diverse wildlife.', im
  {id: 27, name: 'Hemis National Park', location: 'Ladakh', description: 'Famous for its rugged terrain and unique wildlife.', in
  {id: 28, name: 'Dandeli Wildlife Sanctuary', location: 'Karnataka', description: 'Home to rich biodiversity and adventure act
  {id: 29, name: 'Simlipal National Park', location: 'Odisha', description: 'Known for its stunning landscapes and wildlife.', im
  {id: 30, name: 'Palamau Tiger Reserve', location: 'Jharkhand', description: 'Home to tigers and diverse wildlife.', img: '/Asse
  {id: 31, name: 'Buxa Tiger Reserve', location: 'West Bengal', description: 'Famous for its rich biodiversity and scenic beauty
  {id: 32, name: 'Mouling National Park', location: 'Arunachal Pradesh', description: 'Known for its diverse flora and fauna.',
  {id: 33, name: 'Sundha Mata Wildlife Sanctuary', location: 'Rajasthan', description: 'Home to diverse wildlife in a picturesq
  {id: 34, name: 'Bannerghatta National Park', location: 'Karnataka', description: 'Famous for its wildlife and picturesque lanc
  {id: 35, name: 'Simbalbara National Park', location: 'Himachal Pradesh', description: 'Known for its rich biodiversity in the
  {id: 36, name: 'Betla National Park', location: 'Jharkhand', description: 'Home to diverse wildlife and beautiful landscapes.',
  {id: 37, name: 'Mukundra Hills National Park', location: 'Rajasthan', description: 'Famous for its rich biodiversity and sceni
  {id: 38, name: 'Great Himalayan National Park', location: 'Himachal Pradesh', description: 'Home to unique Himalayan wil
  {id: 39, name: 'Neora Valley National Park', location: 'West Bengal', description: 'Known for its rich biodiversity and scenic
  {id: 40, name: 'Dudhwa National Park', location: 'Uttar Pradesh', description: 'Famous for its diverse wildlife and scenic lan
  {id: 41, name: 'Nameri National Park', location: 'Assam', description: 'Home to one-horned rhinoceroses and rich flora.', im
  {id: 42, name: 'Murlen National Park', location: 'Mizoram', description: 'Known for its unique wildlife and beautiful landsc
```

```
    {id: 43, name: 'Ranganathittu Bird Sanctuary', location: 'Karnataka', description: 'Famous for its rich birdlife and scenic bea
    {id: 44, name: 'Cotigao Wildlife Sanctuary', location: 'Goa', description: 'Home to diverse wildlife and beautiful beaches.', i
    {id: 45, name: 'Kuno Wildlife Sanctuary', location: 'Madhya Pradesh', description: 'Known for its rich biodiversity and wild
    {id: 46, name: 'Jaldapara National Park', location: 'West Bengal', description: 'Famous for its diverse wildlife and river ecos
    {id: 47, name: 'Khangchendzonga National Park', location: 'Sikkim', description: 'Home to the Khangchendzonga range and
    {id: 48, name: 'Orang National Park', location: 'Assam', description: 'Famous for its one-horned rhinoceros and rich biodive
    {id: 49, name: 'Pobitora Wildlife Sanctuary', location: 'Assam', description: 'Home to diverse wildlife and beautiful wetland
    {id: 50, name: 'Dibru Saikhowa National Park', location: 'Assam', description: 'Known for its unique ecosystems and rich b
    {id: 51, name: 'Chilika Wildlife Sanctuary', location: 'Odisha', description: "Asia's largest brackish water lagoon famous for
    {id: 52, name: 'Nal Sarovar Bird Sanctuary', location: 'Gujarat', description: 'A wetland known for its winter migratory birds
];
```

Design System (No Changes Required)

Color Palette

- **Primary Green**: #4CAF50 (buttons, headers)
- **Golden/Warning**: #FFA500, #FFD700, goldenrod (accents, donations)
- **Dark**: #333, #114357 (text, borders)
- **Light**: #f4f4f4, aliceblue (backgrounds)
- **Coral**: #f29492 (gradients)

Typography

- **Primary Font**: Sofia (Google Fonts)
- **Fallback**: Arial, sans-serif
- **Hero Text**: 60px (desktop)
- **Headings**: 30-50px
- **Body**: 18-22px

Spacing & Layout

- **Container Max Width**: 70-80vw
- **Border Radius**: 0.8rem - 1.5rem
- **Card Padding**: 20-40px
- **Grid Gap**: 1.5rem

Effects

- **Backdrop Filter**: blur(1-3px) for glassmorphism
- **Box Shadow**: 0 2px 10-40px rgba(0, 0, 0, 0.1-0.4)
- **Transitions**: 0.3-0.5s ease
- **Hover Scale**: 101-105%

Responsive Breakpoints (No Changes)

```css
@media (min-width: 1550px)  /* Large desktops */
@media (max-width: 1199px)  /* Hide column1, column4 */
@media (max-width: 991px)   /* Adjust hero text, navbar */
@media (max-width: 872px)   /* Donation section width */
@media (max-width: 768px)   /* Hide all sanctuary columns */
@media (max-width: 576px)   /* Full width donation section */
@media (max-width: 600px)   /* About page adjustments */
```

Security Best Practices (MySQL Implementation)

1. SQL Injection Prevention

```php
// ALWAYS use prepared statements
$stmt = $pdo->prepare("SELECT * FROM users WHERE email = ?");
$stmt->execute([$email]);

// NEVER do this
$query = "SELECT * FROM users WHERE email = '$email'"; // DANGEROUS!
```

2. Password Security

```php
```

```php
// Registration - Hash password
$hashed_password = password_hash($password, PASSWORD_BCRYPT);

// Login - Verify password
if (password_verify($input_password, $stored_password)) {
    // Password correct
}
```

3. XSS Prevention

```php
// Output user data safely
echo htmlspecialchars($user['name'], ENT_QUOTES, 'UTF-8');
```

4. CSRF Protection

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

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

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

5. Session Security

```php
// Set secure session parameters
ini_set('session.cookie_httponly', 1);
ini_set('session.cookie_secure', 1);
ini_set('session.use_strict_mode', 1);

// Regenerate session ID on login
session_regenerate_id(true);
```

Testing Data for MySQL

Test Users (5 users)

sql

```sql
-- Password for all test users: "Test@123"
INSERT INTO users (name, email, password, donation_total, adoptions_count) VALUES
('Rajesh Kumar', 'rajesh@example.com', '$2y$10$92IXUNpkjO0rOQ5byMi.Ye4oKoEa3Ro9llC/.og/at2.uheWG/igi', 5000.00
('Priya Sharma', 'priya@example.com', '$2y$10$92IXUNpkjO0rOQ5byMi.Ye4oKoEa3Ro9llC/.og/at2.uheWG/igi', 10000.00,
('Amit Patel', 'amit@example.com', '$2y$10$92IXUNpkjO0rOQ5byMi.Ye4oKoEa3Ro9llC/.og/at2.uheWG/igi', 2500.00, 1),
('Sneha Gupta', 'sneha@example.com', '$2y$10$92IXUNpkjO0rOQ5byMi.Ye4oKoEa3Ro9llC/.og/at2.uheWG/igi', 7500.00, 2
('Vikram Singh', 'vikram@example.com', '$2y$10$92IXUNpkjO0rOQ5byMi.Ye4oKoEa3Ro9llC/.og/at2.uheWG/igi', 15000.0
```

Test Donations (10 donations)

sql

```sql
INSERT INTO donations (user_id, donor_name, donor_email, donor_phone, amount, sanctuary_name, recurring_type) VALU
(1, 'Rajesh Kumar', 'rajesh@example.com', '9876543210', 1000.00, 'Ranthambore National Park', 'monthly'),
(1, 'Rajesh Kumar', 'rajesh@example.com', '9876543210', 2000.00, 'Jim Corbett National Park', 'none'),
(2, 'Priya Sharma', 'priya@example.com', '9876543211', 5000.00, 'Kaziranga National Park', 'yearly'),
(2, 'Priya Sharma', 'priya@example.com', '9876543211', 3000.00, 'Gir National Park', 'monthly'),
(3, 'Amit Patel', 'amit@example.com', '9876543212', 2500.00, 'Periyar Wildlife Sanctuary', 'none'),
(4, 'Sneha Gupta', 'sneha@example.com', '9876543213', 7500.00, 'Sundarban National Park', 'yearly'),
(5, 'Vikram Singh', 'vikram@example.com', '9876543214', 10000.00, 'Bandhavgarh National Park', 'monthly'),
(5, 'Vikram Singh', 'vikram@example.com', '9876543214', 5000.00, 'Kanha National Park', 'none'),
(1, 'Rajesh Kumar', 'rajesh@example.com', '9876543210', 2000.00, 'Tadoba Andhari Tiger Reserve', 'none'),
(2, 'Priya Sharma', 'priya@example.com', '9876543211', 2000.00, 'Pench National Park', 'monthly');
```

Test Adoptions (5 adoptions)

sql

```
INSERT INTO adoptions (user_id, animal_name, animal_type, sanctuary_name) VALUES
(1, 'Raja', 'Tiger', 'Ranthambore National Park'),
(1, 'Ganga', 'Rhinoceros', 'Kaziranga National Park'),
(2, 'Simba', 'Lion', 'Gir National Park'),
(2, 'Lakshmi', 'Elephant', 'Periyar Wildlife Sanctuary'),
(2, 'Ravi', 'Tiger', 'Jim Corbett National Park'),
(4, 'Rani', 'Tiger', 'Bandhavgarh National Park'),
(4, 'Kiran', 'Leopard', 'Nagarhole National Park'),
(5, 'Arjun', 'Tiger', 'Kanha National Park'),
(5, 'Maya', 'Elephant', 'Mudumalai Wildlife Sanctuary'),
(5, 'Shera', 'Tiger', 'Tadoba Andhari Tiger Reserve');
```

Environment Variables (.env.example)

```
env
```

```
# Database Configuration
DB_HOST=localhost
DB_NAME=sancturia_wildlife
DB_USER=root
DB_PASS=

# Application Settings
APP_ENV=development
APP_DEBUG=true
APP_URL=http://localhost/sancturia-wildlife

# Session Settings
SESSION_LIFETIME=30
SESSION_SECURE=false
SESSION_HTTPONLY=true

# Security
CSRF_TOKEN_LENGTH=32

# Email Settings (for future implementation)
SMTP_HOST=smtp.gmail.com
SMTP_PORT=587
SMTP_USER=
SMTP_PASS=

# Payment Gateway (for future implementation)
RAZORPAY_KEY_ID=
RAZORPAY_KEY_SECRET=
```

.htaccess Configuration

apache

```
# Enable Rewrite Engine
RewriteEngine On

# Force HTTPS (production only)
# RewriteCond %{HTTPS} off
# RewriteRule ^(.*)$ https://%{HTTP_HOST}%{REQUEST_URI} [L,R=301]

# Prevent directory listing
Options -Indexes

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

# Protect config directory
<DirectoryMatch "config">
    Order deny,allow
    Deny from all
</DirectoryMatch>

# Protect includes directory
<DirectoryMatch "includes">
    Order deny,allow
    Deny from all
</DirectoryMatch>

# Set default charset
AddDefaultCharset UTF-8

# Enable compression
<IfModule mod_deflate.c>
    AddOutputFilterByType DEFLATE text/html text/plain text/xml text/css text/javascript application/javascript
</IfModule>

# Browser caching
<IfModule mod_expires.c>
    ExpiresActive On
    ExpiresByType image/jpg "access plus 1 year"
    ExpiresByType image/jpeg "access plus 1 year"
    ExpiresByType image/gif "access plus 1 year"
    ExpiresByType image/png "access plus 1 year"
```

```
    ExpiresByType text/css "access plus 1 month"
    ExpiresByType application/javascript "access plus 1 month"
</IfModule>

# Security headers
<IfModule mod_headers.c>
    Header set X-XSS-Protection "1; mode=block"
    Header set X-Content-Type-Options "nosniff"
    Header set X-Frame-Options "SAMEORIGIN"
    Header set Referrer-Policy "strict-origin-when-cross-origin"
</IfModule>

# PHP settings
php_value upload_max_filesize 10M
php_value post_max_size 10M
php_value max_execution_time 300
php_value max_input_time 300

# Custom error pages
ErrorDocument 404 /404.html
ErrorDocument 500 /500.html
```

README_MIGRATION.md Structure

The AI should create a comprehensive README_MIGRATION.md with:

1. **Prerequisites**
   - PHP 7.4+
   - MySQL 5.7+
   - Apache/Nginx server
   - PDO PHP extension
2. **Installation Steps**
   - Clone repository
   - Create database
   - Import SQL schema
   - Configure database connection
   - Set file permissions

- Run seed scripts

3. **Configuration**
   - Database credentials
   - Environment variables
   - Session settings
   - Security settings

4. **Testing**
   - Test user credentials
   - Testing checklist
   - Expected results

5. **Deployment**
   - Production checklist
   - Security hardening
   - Performance optimization

6. **Troubleshooting**
   - Common errors
   - Debug mode
   - Log locations

Performance Optimization

Database Optimization

```sql
-- Add indexes for frequently queried columns
CREATE INDEX idx_user_email ON users(email);
CREATE INDEX idx_donation_user ON donations(user_id, donation_date);
CREATE INDEX idx_adoption_user ON adoptions(user_id);
CREATE INDEX idx_sanctuary_name ON sanctuaries(name);

-- Optimize queries
EXPLAIN SELECT * FROM donations WHERE user_id = 1; -- Check query performance
```

PHP Optimization

```php
// Use persistent connections
$pdo = new PDO($dsn, $username, $password, [
    PDO::ATTR_PERSISTENT => true
]);

// Cache frequently accessed data
$sanctuaries_cache = apcu_fetch('sanctuaries');
if ($sanctuaries_cache === false) {
    $sanctuaries_cache = fetch_all_sanctuaries();
    apcu_store('sanctuaries', $sanctuaries_cache, 3600); // Cache for 1 hour
}
```

Monitoring & Logging

Error Logging

```php
// includes/error_handler.php
function log_error($message, $level = 'ERROR') {
    $log_file = __DIR__ . '/../logs/error.log';
    $timestamp = date('Y-m-d H:i:s');
    $log_message = "[$timestamp] [$level] $message\n";
    error_log($log_message, 3, $log_file);
}

// Usage
try {
    // Database operation
} catch (PDOException $e) {
    log_error("Database error: " . $e->getMessage());
    // Show user-friendly message
}
```

Activity Logging

```sql
```

```sql
-- Create activity_logs table
CREATE TABLE activity_logs (
    log_id INT PRIMARY KEY AUTO_INCREMENT,
    user_id INT,
    action VARCHAR(100),
    details TEXT,
    ip_address VARCHAR(45),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES users(user_id) ON DELETE SET NULL,
    INDEX idx_user_action (user_id, action),
    INDEX idx_created_at (created_at)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

php

```php
// Log user activity
function log_activity($user_id, $action, $details = null) {
    global $pdo;
    $ip = $_SERVER['REMOTE_ADDR'];
    $stmt = $pdo->prepare("INSERT INTO activity_logs (user_id, action, details, ip_address) VALUES (?, ?, ?, ?)");
    $stmt->execute([$user_id, $action, $details, $ip]);
}

// Usage examples
log_activity($user_id, 'LOGIN', 'User logged in successfully');
log_activity($user_id, 'DONATION', "Donated ₹{$amount} to {$sanctuary}");
log_activity($user_id, 'ADOPTION', "Adopted animal: {$animal_name}");
```

API Response Format

All API endpoints should return consistent JSON responses:

Success Response

json

```
{
  "success": true,
  "data": {
    "user_id": 1,
    "name": "Rajesh Kumar",
    "email": "rajesh@example.com"
  },
  "message": "Data retrieved successfully"
}
```

Error Response

json

```
{
  "success": false,
  "error": {
    "code": "AUTH_FAILED",
    "message": "Invalid credentials"
  }
}
```

Example API Endpoint

php

```php
// api/get_user.php
<?php
session_start();
require_once '../config/database.php';

header('Content-Type: application/json');

try {
    if (!isset($_SESSION['user_id'])) {
        throw new Exception('Not authenticated');
    }

    $user_id = $_SESSION['user_id'];
    $stmt = $pdo->prepare("SELECT user_id, name, email, donation_total, adoptions_count FROM users WHERE user_id = ?
    $stmt->execute([$user_id]);
    $user = $stmt->fetch(PDO::FETCH_ASSOC);

    if (!$user) {
        throw new Exception('User not found');
    }

    echo json_encode([
        'success' => true,
        'data' => $user,
        'message' => 'User data retrieved successfully'
    ]);

} catch (Exception $e) {
    http_response_code(400);
    echo json_encode([
        'success' => false,
        'error' => [
            'code' => 'FETCH_ERROR',
            'message' => $e->getMessage()
        ]
    ]);
}
```

Complete Migration Timeline

Week 1: Database Setup

- Day 1-2: Create database schema, tables, indexes
- Day 3-4: Write seed scripts for sanctuaries and test data
- Day 5: Test database connections and queries

Week 2: Authentication System

- Day 1-2: Create auth.php with all authentication functions
- Day 3: Update signup.php to use MySQL
- Day 4: Create login.php backend
- Day 5: Test registration and login flows

Week 3: Dashboard & Core Features

- Day 1-2: Update dashboard.php with MySQL queries
- Day 3: Create donation processing script
- Day 4-5: Create API endpoints and test

Week 4: Security & Testing

- Day 1-2: Implement CSRF protection, XSS prevention
- Day 3: Create session handler and security features
- Day 4-5: Comprehensive testing and bug fixes

Key Differences: Firebase vs MySQL

Feature	Firebase	MySQL + PHP
Authentication	REST API with API key	PHP sessions with password hashing
Data Storage	NoSQL (Firestore)	Relational (MySQL)
Queries	REST API calls with cURL	SQL queries with PDO
User ID	Firebase UID (string)	Auto-increment INT
Sessions	idToken from Firebase	PHP $_SESSION array
Real-time	Built-in listeners	Manual AJAX polling
Hosting	Firebase Hosting	Apache/Nginx server
Scalability	Auto-scaling	Manual server scaling
Cost	Pay per usage	Server + database hosting

Common Migration Pitfalls & Solutions

Pitfall 1: Hardcoded Firebase URLs

Problem: Firebase URLs scattered throughout code **Solution**: Use centralized database connection file

Pitfall 2: Different Data Types

Problem: Firestore uses nested objects, MySQL uses flat tables **Solution**: Restructure data into normalized tables with relationships

Pitfall 3: Authentication Token Confusion

Problem: Code expects Firebase idToken **Solution**: Replace with PHP session checks everywhere

Pitfall 4: Async/Promise Patterns

Problem: Firebase uses promises, PHP is synchronous **Solution**: Remove async/await patterns, use direct function calls

Pitfall 5: Missing Foreign Keys

Problem: Data integrity issues without proper relationships **Solution**: Use FOREIGN KEY constraints and ON DELETE rules

---

Testing Checklist for AI Implementation

After the AI completes the migration, verify:

Database Tests

- [ ] All 5 tables created successfully
- [ ] Foreign key constraints working
- [ ] Indexes created properly
- [ ] 52 sanctuaries inserted
- [ ] Test users can be inserted
- [ ] Test donations linked to users

Authentication Tests

- [ ] User registration creates database entry
- [ ] Password is hashed (not plain text)

- [ ] Login validates credentials correctly
- [ ] Session is created on successful login
- [ ] Logout destroys session
- [ ] Duplicate email registration blocked

Dashboard Tests

- [ ] User profile displays correctly
- [ ] Recent donations show (max 5)
- [ ] Recent adoptions display
- [ ] Recommended sanctuaries appear (3)
- [ ] Donation total calculates correctly
- [ ] Adoptions count is accurate

Donation Tests

- [ ] Form submits to process_donation.php
- [ ] Data validates before insertion
- [ ] Donation inserts into donations table
- [ ] User donation_total updates
- [ ] Redirects to thank you page
- [ ] Recurring type saves correctly

Security Tests

- [ ] SQL injection attempts blocked
- [ ] XSS attempts sanitized
- [ ] CSRF tokens validate
- [ ] Passwords not visible in database
- [ ] Sessions expire after inactivity
- [ ] Unauthorized access blocked

API Tests

- [ ] get_user.php returns user data
- [ ] get_donations.php returns user donations
- [ ] get_sanctuaries.php returns all sanctuaries
- [ ] search_sanctuary.php finds by name
- [ ] All endpoints check authentication
- [ ] All endpoints return valid JSON

Future Enhancements (Post-Migration)

Phase 1: Payment Gateway Integration

php

```php
// integration/razorpay.php
require_once 'vendor/autoload.php';

use Razorpay\Api\Api;

$api = new Api($keyId, $keySecret);

$orderData = [
    'receipt' => 'order_' . time(),
    'amount' => $amount * 100, // Amount in paise
    'currency' => 'INR',
    'payment_capture' => 1
];

$razorpayOrder = $api->order->create($orderData);
```

Phase 2: Email Notifications

php

```php
// includes/mailer.php
use PHPMailer\PHPMailer\PHPMailer;

function send_donation_receipt($email, $name, $amount, $sanctuary) {
    $mail = new PHPMailer(true);
    $mail->setFrom('noreply@sancturia.org', 'Sancturia Wildlife');
    $mail->addAddress($email, $name);
    $mail->Subject = 'Thank You for Your Donation';
    $mail->Body = "Dear $name, Thank you for donating ₹$amount to $sanctuary...";
    $mail->send();
}
```

Phase 3: Admin Dashboard

sql

```
-- Create admins table
CREATE TABLE admins (
    admin_id INT PRIMARY KEY AUTO_INCREMENT,
    user_id INT UNIQUE,
    role ENUM('super_admin', 'admin', 'moderator'),
    permissions JSON,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES users(user_id) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

Phase 4: Analytics Dashboard

php

```
// pages/admin/analytics.php
// Total donations by sanctuary
$stmt = $pdo->query("
    SELECT sanctuary_name, SUM(amount) as total, COUNT(*) as count
    FROM donations
    GROUP BY sanctuary_name
    ORDER BY total DESC
    LIMIT 10
");

// Monthly donation trends
$stmt = $pdo->query("
    SELECT DATE_FORMAT(donation_date, '%Y-%m') as month, SUM(amount) as total
    FROM donations
    GROUP BY month
    ORDER BY month DESC
    LIMIT 12
");
```

Backup & Recovery

Database Backup Script

bash

```
#!/bin/bash
# backup.sh

DATE=$(date +%Y%m%d_%H%M%S)
BACKUP_DIR="/var/backups/sancturia"
DB_NAME="sancturia_wildlife"

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

# Compress
gzip $BACKUP_DIR/backup_$DATE.sql

# Delete backups older than 30 days
find $BACKUP_DIR -name "backup_*.sql.gz" -mtime +30 -delete

echo "Backup completed: backup_$DATE.sql.gz"
```

Restore from Backup

bash

```
# Decompress
gunzip backup_20241023_120000.sql.gz

# Restore
mysql -u root -p sancturia_wildlife < backup_20241023_120000.sql
```

Deployment Instructions

Local Development Setup

bash

```
# 1. Clone repository
git clone https://github.com/yourorg/sancturia-wildlife.git
cd sancturia-wildlife

# 2. Create database
mysql -u root -p -e "CREATE DATABASE sancturia_wildlife CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;

# 3. Import schema
mysql -u root -p sancturia_wildlife < database/schema.sql

# 4. Import seed data
mysql -u root -p sancturia_wildlife < database/seeds.sql

# 5. Configure database connection
cp config/database.example.php config/database.php
# Edit database.php with your credentials

# 6. Set permissions
chmod 755 -R .
chmod 777 -R logs/
chmod 777 -R uploads/

# 7. Start local server
php -S localhost:8000
```

Production Deployment

bash

```
# 1. Upload files via FTP/SFTP to server

# 2. Create production database
mysql -u username -p -e "CREATE DATABASE sancturia_wildlife_prod;"

# 3. Import schema and data
mysql -u username -p sancturia_wildlife_prod < schema.sql

# 4. Update config/database.php with production credentials

# 5. Set proper permissions
chmod 755 -R /var/www/html/sancturia-wildlife
chmod 600 config/database.php

# 6. Enable HTTPS in .htaccess (uncomment RewriteRule)

# 7. Test all functionality

# 8. Set up cron jobs for backups
crontab -e
# Add: 0 2 * * * /path/to/backup.sh
```

Maintenance Schedule

Daily Tasks

- Monitor error logs
- Check database connections
- Review failed login attempts

Weekly Tasks

- Backup database
- Review user activity logs
- Check donation statistics
- Update sanctuary information if needed

Monthly Tasks

- Database optimization (OPTIMIZE TABLE)

- Security audit
- Performance analysis
- Update dependencies

Quarterly Tasks

- Full security assessment
- Code review
- User feedback analysis
- Feature prioritization

Support & Documentation

For Developers

- **API Documentation**: `/docs/api.md`
- **Database Schema**: `/docs/schema.md`
- **Coding Standards**: `/docs/standards.md`
- **Contributing Guide**: `/docs/CONTRIBUTING.md`

For Users

- **User Guide**: `/docs/user-guide.md`
- **FAQ**: `/docs/faq.md`
- **Contact**: support@sancturia.org

For Administrators

- **Admin Manual**: `/docs/admin-manual.md`
- **Deployment Guide**: `/docs/deployment.md`
- **Troubleshooting**: `/docs/troubleshooting.md`

Conclusion

This comprehensive documentation provides everything needed to migrate the Sancturia Wildlife project from

Firebase to MySQL + PHP. The migration maintains all existing frontend functionality while replacing the backend infrastructure with a traditional LAMP stack.

Key Migration Benefits:

1. **Full Control**: Own your data and infrastructure
2. **Cost Effective**: No per-usage Firebase fees
3. **Better Performance**: Optimized SQL queries
4. **Enhanced Security**: Custom authentication and authorization
5. **Flexibility**: Easy to add custom features
6. **Scalability**: Can optimize for specific needs

What Stays the Same:

- All HTML/CSS/JavaScript frontend code
- User interface and experience
- Bootstrap styling
- LocalStorage functionality
- 52 sanctuaries data
- Donation tiers and features

What Changes:

- Backend from Firebase to MySQL
- Authentication from REST API to PHP Sessions
- Database queries from Firestore to SQL
- User management system
- Session handling
- Security implementation

Quick Reference Commands

Database

```bash
```

```
# Connect to database
mysql -u root -p sancturia_wildlife

# Show all tables
SHOW TABLES;

# Describe table structure
DESCRIBE users;

# Count records
SELECT COUNT(*) FROM users;
```

Git

bash

```
# Create new branch for migration
git checkout -b mysql-migration

# Commit changes
git add .
git commit -m "Migrated from Firebase to MySQL"

# Push to remote
git push origin mysql-migration
```

PHP

bash

```
# Check PHP version
php -v

# Check installed extensions
php -m

# Run PHP built-in server
php -S localhost:8000 -t public/
```

Final Notes for AI

When implementing this migration, please:

1. **Preserve all frontend code** - Only change backend files
2. **Use consistent naming** - Follow existing conventions
3. **Add comprehensive comments** - Explain complex logic
4. **Include error handling** - Every database operation needs try-catch
5. **Follow security best practices** - Use prepared statements, sanitize inputs
6. **Test thoroughly** - Provide testing checklist results
7. **Document everything** - Create README_MIGRATION.md with setup instructions
8. **Maintain backward compatibility** - Same user experience after migration

IMPORTANT: The goal is to create a seamless transition where users don't notice any difference in functionality, only backend improvements in performance, security, and maintainability.

END OF COMPLETE DOCUMENTATION

Total Pages: ~50 pages
Total Words: ~15,000 words
Last Updated: 2024
Version: 1.0 (MySQL Migration Edition)