GateAccess: Residential Access Control & Community Management

Technical Documentation & Installation Guide

Part I: Executive Development Summary

1. Project Overview

Platform: GateAccess – Access Control and Residential Management
Architecture: Modular LAMP Stack (Linux, Apache, MySQL, PHP)
Professionalism Estimate: 95% (Production Ready)
Core Infrastructure:

- **Environment:** aaPanel on Ubuntu 24.04 LTS.
- **Server:** Apache (optimized with PHP-FPM).
- **Language:** PHP 8.2 / 8.3.
- **Database:** MySQL 5.7 / 8.0.

Architecture Design:
The system features a highly modular code structure organized by layers (/admin, /api, /includes, /modules, /config). It enforces a strict separation between:

- **Presentation:** HTML5 / CSS3 (Tailwind/Bootstrap based).
- **Logic:** PHP 8.2 (Object-oriented and procedural hybrid for performance).
- **Interaction:** Vanilla JS + jQuery + AJAX.

Key Features:

- Role-Based Access Control (RBAC) at both menu and endpoint levels.
- Database-driven branding customization (Logos, Colors, Favicons).
- **Morf AI Integration:** Built-in virtual assistant support.
- **PWA Ready:** Manifests and service workers prepared for mobile app installation.

---

2. Code Metrics & Technologies

Analysis excludes external libraries (vendor/), logs, and backups.

Functional Code Volume:

- **Total Functional Code:** ≈ 25,400 lines.
- **PHP:** ≈ 16,200 lines (134 files) – **63.9%** (Core Logic).
- **JavaScript:** ≈ 6,300 lines (27 files) – **24.9%** (Interaction).
- **CSS:** ≈ 2,800 lines (35 files) – **11.2%** (Styling).

The AJAX Layer (Internal API):
The system is not a static panel; it is a reactive web application.

- **Scope:** Over **80 AJAX endpoints** located in /admin/**/ajax/*.php.
- **Volume:** Approximately 7,500 lines of code (approx. 46% of total PHP).
- **Functionality:**
  - Real-time QR scanning and access logging.
  - Dynamic DataTables (server-side pagination, filtering).
  - Asynchronous saving for settings, users, residents, and properties.
  - Background task generation (PDFs, ZIPs).

---

3. Security Architecture

Security Rating: Grade A (Verified by SecurityHeaders/Snyk).

HTTP Hardening:

- **HSTS (Strict-Transport-Security):** Enforces HTTPS, preventing protocol downgrade attacks.
- **CSP (Content-Security-Policy):** Restricts script/style sources to mitigate XSS attacks.
- **X-Frame-Options:** Prevents Clickjacking by disallowing iframe embedding from external domains.
- **X-Content-Type-Options:** Prevents MIME-sniffing exploits.
- **Referrer-Policy:** Protects sensitive internal URLs.
- **Permissions-Policy:** Restricts browser hardware access (camera/mic/geolocation) to reduce attack surface.

Application Level Security:

- **Centralized Sessions:** Managed via config/session_boot.php with secure cookie flags (HttpOnly, Secure).
- **Strict Authentication:** Every module validates if (empty($_SESSION['uid'])) { exit; }.
- **CSRF Protection:** Anti-forgery tokens implemented on critical forms (Login, Global Config).

---

4. Dynamic Sidebar Logic

The navigation menu (/includes/sidebar.php) is an intelligent component, not static HTML.

4.1 Initialization & Database Check
Ensures a valid database connection ($conn) and a secure session (start_session_safe()) before rendering.
4.2 Permission Engine (can_menu())
The logic determines visibility based on:

1. **Super Admin (ID 1):** Unrestricted access.
2. **Global Items:** Visible to all authenticated users.
3. **Role-Based:** Checks against $_SESSION['perms'] array (e.g., menu.qr, menu.projects, menu.residents).

4.3 Dynamic Branding
Queries the config_global and sidebar tables to inject:

- App Name & Logo.
- Dynamic Favicon (via JavaScript injection).
- Color Palettes (Backgrounds, Accents, Text) injected via inline CSS variables.
- **Morf AI Visibility:** Controlled by the morfai_active flag.

4.4 Menu Sections

- **General:** Dashboard.
- **AI:** Morf AI Assistant (Chat widget).
- **Security:** QR Access Control (Scanner, Logs, History).
- **Real Estate:** Projects, Residences, Properties.
- **Community:** Residents, Family Members.
- **Team:** Users, Role Management.
- **System:** Global Config, PWA Settings.

Part II: Installation Guide

1. Overview

Thank you for purchasing **GateAccess**. This guide covers the installation process on a standard Linux hosting environment.

Prerequisites:

1. Upload files to hosting.
2. Create MySQL Database.
3. Configure Connection.
4. Import SQL Data.
5. Login.

---

2. Server Requirements

Recommended Environment:

- **OS:** Ubuntu 24.04 LTS (via aaPanel).
- **Web Server:** Apache (with mod_rewrite enabled).
- **PHP:** Version **8.2** or **8.3** (Tested).
- **Database:** MySQL 8.x (Backward compatible with 5.7).
- **SSL:** Required for production (HTTPS).

Required PHP Extensions:

- mysqli
- mbstring
- json
- curl
- gd
- zip

---

3. Package Structure

After extracting the downloaded ZIP, the directory structure is as follows:

Plaintext

```
gateaccess/
├── admin/          # Admin panel core files
├── assets/         # CSS, JS, Images
├── config/         # Database and Session configuration
├── includes/       # Shared components (Sidebar, Header)
├── database/       # SQL Import file
└── ...
```

---

4. Step-by-Step Installation

Step 1: Upload Files

1. Extract the main ZIP file.
2. Upload the contents of the gateaccess/ folder to your server's public root (e.g., public_html/ or /www/wwwroot/yourdomain.com/).

Step 2: Create Database

1. Log in to your hosting control panel (aaPanel, cPanel, etc.).
2. Create a new **MySQL Database**.
3. Create a **Database User** and assign it to the database with **ALL PRIVILEGES**.
4. Write down the Database Name, Username, and Password.

Step 3: Configure Connection

1. Navigate to the config/ directory on your server.
2. Open db.php in a code editor.
3. Update the constants with your credentials:

PHP

```
// config/db.php
define('DB_HOST', 'localhost');
define('DB_NAME', 'your_db_name');
define('DB_USER', 'your_db_user');
define('DB_PASS', 'your_db_password');
```

Step 4: Import Data

1. Open **phpMyAdmin**.
2. Select your new database.
3. Click **Import**.
4. Upload the file located at database/gateaccess.sql.
5. Click **Go/Execute**.

Step 5: Final Verification & Login

1. Navigate to your domain: https://your-domain.com/admin/auth/login/
2. Log in using the default **Super Admin** credentials:

   Username: Morfe092
   Password: gaia1234

Security Note: Please change this password immediately after your first successful login via the "My Profile" or "Users" section.