Blog Platform Documentation

Project Structure and File Roles

1. Application Core (`/app`)

1.1 Configuration (`/app/config`)
- **config.php**
  - Role: Contains database credentials and application settings
  - Manages environment variables and global configuration
  - Defines constants used throughout the application

- **routes.php**
  - Role: Defines all application routes
  - Maps URLs to controller actions
  - Handles both public and authenticated routes

1.2 Controllers (`/app/controllers`)

Admin Controllers (`/app/controllers/back`)
- **AdminDashboardController.php**
  - Role: Manages admin panel functionality
  - Handles user management (view, activate/deactivate users)
  - Manages article moderation
  - Shows admin statistics and reports

Main Controllers
- **ArticleController.php**
  - Role: Handles article-related operations
  - Lists all articles
  - Shows individual article details
  - Manages article creation, editing, and deletion

- **AuthController.php**
  - Role: Manages user authentication
  - Handles login and registration
  - Manages password reset functionality
  - Controls session management

- **HomeController.php**
  - Role: Manages the homepage
  - Displays featured articles
  - Shows latest content
  - Presents category listings

- **UserDashboardController.php**
  - Role: Manages user dashboard functionality
  - Handles user's article management
  - Shows user statistics
  - Manages user profile settings

1.3 Core Framework (`/app/core`)
- **Controller.php**
  - Role: Base controller class
  - Provides common controller functionality
  - Handles view rendering
  - Manages flash messages

- **Database.php**
  - Role: Database connection management
  - Provides PDO connection instance
  - Handles database configuration
  - Manages connection errors

- **Router.php**
  - Role: URL routing system
  - Parses incoming requests
  - Matches URLs to controllers
  - Handles 404 errors

1.4 Middleware (`/app/middleware`)
- **AuthMiddleware.php**
  - Role: Authentication verification
  - Checks user login status
  - Protects authenticated routes
  - Redirects unauthorized access

- **AdminMiddleware.php**
  - Role: Admin access control
  - Verifies admin privileges
  - Protects admin routes
  - Handles unauthorized admin access

```
#### 1.5 Models (`/app/models`)
- **Article.php**
  - Role: Article data management
  - Handles CRUD operations for articles
  - Manages article relationships
  - Implements article search

- **Category.php**
  - Role: Category management
  - Handles category CRUD operations
  - Manages category-article relationships
  - Provides category listings

- **User.php**
  - Role: User data management
  - Handles user CRUD operations
  - Manages user authentication
  - Handles user roles and permissions

#### 1.6 Views (`/app/views`)

##### Admin Views (`/app/views/admin`)
- **dashboard.twig**
  - Role: Admin dashboard layout
  - Shows admin statistics
  - Lists recent activities
  - Provides quick actions

- **users.twig**
  - Role: User management interface
  - Lists all users
  - Shows user details
  - Provides user action buttons

##### Article Views (`/app/views/articles`)
- **index.twig**
  - Role: Article listing page
  - Shows article grid/list
  - Implements pagination
  - Provides category filters

- **show.twig**
  - Role: Single article display
  - Shows full article content
  - Displays author information
  - Shows related articles

##### Authentication Views (`/app/views/auth`)
- **login.twig**
  - Role: Login form
  - Handles user authentication
  - Shows error messages
  - Provides password reset link

- **register.twig**
  - Role: Registration form
  - Collects user information
  - Validates input
  - Shows registration status

##### Dashboard Views (`/app/views/dashboard`)
- **index.twig**
  - Role: User dashboard
  - Shows user's articles
  - Displays user statistics
  - Lists recent activity

- **create_article.twig**
  - Role: Article creation form
  - Handles article input
  - Manages category selection
  - Provides rich text editor

##### Home Views (`/app/views/home`)
- **index.twig**
  - Role: Homepage layout
  - Shows featured content
  - Displays recent articles
  - Lists popular categories

##### Layout Views (`/app/views/layouts`)
- **app.twig**
  - Role: Main application layout
```

  - Defines page structure
  - Includes common elements
  - Manages navigation

- **dashboard.twig**
  - Role: Dashboard layout
  - Provides dashboard structure
  - Includes navigation menu
  - Shows user information

2. Database (`/migrations`)
- Contains all database migrations
- Manages database schema
- Handles data seeding
- Version controls database changes

3. Public Files (`/public`)
- **index.php**
  - Role: Application entry point
  - Bootstraps the application
  - Handles request routing
  - Manages error handling

- **assets/**
  - Role: Public resources
  - Contains CSS files
  - Manages JavaScript files
  - Stores public images

4. Dependencies (`/vendor`)
- Contains Composer dependencies
- Manages third-party libraries
- Handles autoloading
- Provides external functionality

Key Features

1. **Authentication System**
   - Secure user registration and login
   - Password hashing and verification
   - Session management
   - Role-based access control

2. **Article Management**
   - Create, read, update, delete articles
   - Category organization
   - Rich text editing
   - Image upload support

3. **User Dashboard**
   - Personal article management
   - User statistics
   - Profile settings
   - Activity tracking

4. **Admin Panel**
   - User management
   - Content moderation
   - System statistics
   - Configuration settings

5. **Security Features**
   - CSRF protection
   - XSS prevention
   - SQL injection protection
   - Input validation

Database Schema

Users Table
```sql
CREATE TABLE users (
    id SERIAL PRIMARY KEY,
    username VARCHAR(255) NOT NULL UNIQUE,
    email VARCHAR(255) NOT NULL UNIQUE,
    password VARCHAR(255) NOT NULL,
    role VARCHAR(50) DEFAULT 'user',
    active BOOLEAN DEFAULT true,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Articles Table

```sql
CREATE TABLE articles (
    id SERIAL PRIMARY KEY,
    title VARCHAR(255) NOT NULL,
    content TEXT NOT NULL,
    user_id INTEGER REFERENCES users(id),
    category_id INTEGER REFERENCES categories(id),
    status VARCHAR(50) DEFAULT 'draft',
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Categories Table
```sql
CREATE TABLE categories (
    id SERIAL PRIMARY KEY,
    name VARCHAR(255) NOT NULL UNIQUE,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Development Guidelines

1. **Coding Standards**
   - Follow PSR-4 autoloading
   - Use PSR-12 coding style
   - Implement SOLID principles
   - Write comprehensive comments

2. **Security Practices**
   - Validate all user input
   - Escape output data
   - Use prepared statements
   - Implement CSRF protection

3. **Performance Optimization**
   - Cache frequently accessed data
   - Optimize database queries
   - Minimize HTTP requests
   - Use efficient algorithms

4. **Maintenance**
   - Regular security updates
   - Database backups
   - Log monitoring
   - Performance tracking