



Marsa Maroc Port Management



Introduction

Marsa Maroc Port Management is a comprehensive web application designed to revolutionize port operations and ship activity management. Built on the powerful combination of Laravel and TailwindCSS, this platform offers an integrated solution for handling daily reports, ship information, personnel management, and equipment operations within the dynamic port environment.



Key Features



User Authentication

Our secure authentication system provides role-based access control, ensuring that sensitive information remains protected while maintaining operational efficiency.



Daily Reporting System

Create, manage, and search through daily operational reports with our intuitive date-based search functionality, making historical data readily accessible.



Ship Management

Comprehensive ship information tracking including:

- Operational schedules
- Vessel specifications
- Current status monitoring
- Historical operation records



Personnel Management

Efficiently manage your workforce with features for:

- Staff information tracking
- Ship assignments
- Equipment operation assignments
- Shift scheduling



Equipment Management

Keep track of all port equipment with:

- Real-time operational status
- Maintenance schedules
- Ship assignment records
- Usage history

Stoppage Management

Monitor and manage operational interruptions with detailed tracking of:

- Equipment stoppages
- Ship delays
- Maintenance periods
- Resolution tracking

Installation Guide

Prerequisites

Before beginning the installation, ensure you have:

- PHP 8.0 or higher
- Composer package manager
- Node.js installation
- MySQL database server

Setup Steps

1. **Repository Clone:**

```
git clone https://github.com/your-username/your-repo.git
cd your-repo
```

2. **Dependency Installation:**

```
composer install
npm install
```

3. **Environment Setup:**

```
cp .env.example .env
php artisan key:generate
```

Remember to update your database credentials in the `.env` file.

4. **Database Configuration:**

```
php artisan migrate
```

Optional: Add `--seed` flag for sample data population.

5. Asset Compilation:

```
npm run dev
```



6. Launch Development Server:

```
php artisan serve
```



Access your application at <http://localhost:8000>

Login Credentials

Admin Access:






-  Email: admin@marsa.com
-  Password: admin

User Access:

-  Email: user@marsa.com
-  Password: user






Project Architecture

Core Components:

1.  **Controllers:** Request handling and response management
2.  **Models:** Database interaction layer
3.  **Views:** Blade template rendering
4.  **Routes:** Application endpoint definitions
5.  **Seeders:** Initial data population

Technology Stack

Our application leverages modern technologies including:

-  Laravel: Robust PHP framework
-  TailwindCSS: Modern utility-first styling
-  MySQL: Reliable data storage
-  Laravel Breeze: Streamlined authentication
-  UML: Database architecture design

Development Challenges

Solo Development Journey

The project faced significant challenges due to its solo development nature. While this resulted in some incomplete features and technical debt, it provided valuable lessons about:

- The importance of team collaboration
- Technical architecture planning
- Resource management
- Future scalability considerations

These insights will guide future improvements and expansions of other projects.