Velox

August 2025

Marito - Technical Installation Manual

Prepared for

Data Science for Social Impact(DSFSI)

Team Contact

veloxcapstone@gmail.com

Presented by

Zainab Abdulrasaq
Hayley Dodkins
Arnaud Strydom
Palesa Thabane
Eugen Vosloo

Contents

Introduction

This technical installation manual provides comprehensive instructions for setting up the Marito system on your system for local development. The manual outlines the installation process. The installation starts with system requirements and prerequisite software installation for both Linux and Windows platforms. You will then need to clone the project repository, configure the backend services including database setup and Google Cloud integration, and finally set up the frontend development environment. Each section includes detailed commands.

Requirements

- Git (version 2.34.1 or higher)
- Node.js (version 18.x or higher)
- PostgreSQL (version 14.0 or higher)
- Docker (version 24.0.0 or higher)
- Docker Compose (version 2.21.0 or higher)

Requirement Installation - Linux

1. Install Git

Reference Installation Guide: Click Here
Run the following in your terminal:

```
sudo apt update
sudo apt install git
```

2. Install Node.js

Reference Installation Guide: Click Here
Run the following in your terminal:

```
curl -fsSL https://deb.nodesource.com/setup_18.x | sudo -E bash -
sudo apt-get install -y nodejs
```

3. Install PostgreSQL

Reference Installation Guide: Click Here
Run the following in your terminal:

```
sudo apt update
sudo apt install postgresql postgresql-contrib
sudo systemctl start postgresql
sudo systemctl enable postgresql
```

Requirement Installation - Linux

4. Install Docker and Docker Compose

Reference Installation Guide: Click Here
Run the following in your terminal:

```
sudo apt update
sudo apt install ca-certificates curl gnupg lsb-release
sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | \
  sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg]
   https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee
  /etc/apt/sources.list.d/docker.list > /dev/null
 sudo apt update
sudo apt install docker-ce docker-ce-cli containerd.io docker-compose-plugin
```

Add your user to the docker group:

```
sudo usermod -aG docker $USER
newgrp docker
```

Requirement Installation - Windows

1. Install Git

Reference Installation Guide: Click Here

- Download and run the Git installer for Windows:
- Download from : https ://git-scm.com/downloads/win
- Run the installer with default settings
- Verify installation in PowerShell :

```
git -- version
```

2. Install Node.js

Reference Installation Guide: Click Here

- Download from : https://nodejs.org/en/download/
- Run the Windows Installer (.msi)
- Verify installation in PowerShell :

```
node -- version
npm -- version
```

3. Install PostgreSQL

Reference Installation Guide: Click Here

- Download from : https://www.postgresql.org/download/windows/
- Run the Windows installer
- Remember the password you set for the postgres user
- Verify installation in PowerShell :

```
psql -- version
```

Requirement Installation - Windows

4. Install Docker Desktop And Docker Compose

Reference Installation Guide: Click Here

- Download from : https://docs.docker.com/desktop/install/windows-install/
- Run Docker Desktop Installer.exe
- Restart your computer when prompted
- Start Docker Desktop from Start Menu
- Verify installation in PowerShell :

```
docker -- version
docker compose version
```

Cloning the Repository

1. Choose Installation Directory

Navigate to the directory where you want to install Marito:

```
cd /path/to/your/desired/directory
```

2. Clone the Repository

```
git clone git@github.com:COS301-SE-2025/Mavito.git
```

3. Navigate To Project Directory

```
cd Mavito
```

Backend Setup

1. Configure Google Cloud Service Account for Bucket Access

To enable your backend to access Google Cloud Storage buckets, follow these steps to use a Service Account Key JSON:

- Go to: Google Cloud Console IAM Service Accounts
- Select your project
- Click **Create Service Account**
- Give it a name (e.g., marito-backend-sa)
- Grant it the role: Storage Admin (or more limited roles like Storage Object Viewer as needed).
- Click **Create Key** , Select , JSON Download the file
- Place the downloaded .json file in your backend directory (/backend)

Ensure your backend's Docker service has access to the key by mounting it. In your `docker-compose.yml:`

```
backend-service:
  volumes:
   -./keys/your-key-file.json:/app/keys/service-account.json
  environment :
   GOOGLE_APPLICATION_CREDENTIALS : /app/keys/service-account.json
```

2. Create .env File for Local Development

The backend requires a .env file in the /backend directory to run properly in local environments. This file holds sensitive configuration values such as database connection info, secret keys, and CORS settings.

```
cd backend/
```

Backend Setup

You can create the file manually or use a text editor to paste the following contents: Create a .env file with the following content:

```
# Application Settings
PROJECT_NAME = "Mavito Local"
SECRET_KEY = " ChangeThisKeyForProduction "
ACCESS_TOKEN_EXPIRE_MINUTES =10080

# Database Configuration for Local PostgreSQL
DB_USER = "mavito_dev_user"
DB_PASSWORD = "mavito_dev_password"
DB_NAME = "mavito_local_dev_db"
DB_HOST = "db"
DB_PORT = "5432"

# CORS Configuration
BACKEND_CORS_ORIGINS = http://localhost:5173, http://localhost:3000

# Admin Account Defaults ( used for first seed )
ADMIN_EMAIL = admin@example . com
ADMIN_PASSWORD = ChangeThisInProduction123 !
ADMIN_FIRST_NAME = Alice
ADMIN_LAST_NAME = Root
```

Notes:

- Values such as PROJECT_NAME, SECRET_KEY, DB_USER, and DB_PASSWORD can be customized to suit your local environment.
- The SECRET_KEY is used for cryptographic operations such as signing JWT tokens. It should be unique and kept secret.
- ADMIN_EMAIL and ADMIN_PASSWORD are used to create the default admin account during database seeding. These credentials can be used to log into the system with admin privileges after setup.
- BACKEND_CORS_ORIGINS specifies which frontend URLs are allowed to interact with the backend. You can add additional URLs as needed.

Backend Setup

3. Stop All Services and Delete Database Volume

Navigate to your backend directory and run these commands:

```
docker-compose down
```

WARNING: Only run this once to rebuild the database for new setup - This erases your local development database:

```
docker volume rm backend_postgres_data
```

4. Start Database Service

Start only the database container first::

```
docker-compose up --build -d db
```

5. Apply Database Migrations

Run the database migrations to create all tables::

```
docker-compose run --rm alembic-service alembic -c migrations/alembic.ini upgrade head
```

6. Start All Services

Start all remaining services:

```
docker-compose up -d
```

7. Seed The Database

Only run once initially to populate the tables :

```
docker-compose exec search-service python scripts/ingest_data.py
docker-compose exec auth-service python scripts/ingest_data.py
```

Frontend Setup

1. Create Environment Configuration

- Navigate to your frontend directory and create a .env file
- Create a .env file with the following content:

```
VITE_AUTH_SERVICE_URL=http://localhost:8001
VITE_SEARCH_SERVICE_URL=http://localhost:8002
VITE_ANALYTICS_SERVICE_URL=http://localhost:8003
VITE_LINGUIST_APP_SERVICE_URL=http://localhost:8004
```

2. Install Frontend Dependencies

Install the required dependencies:

```
npm install
```

3. Start Development Server

Run the development server:

```
npm run dev
```

4. Access the Application

Open your web browser and navigate to:
URL: http://localhost:5173
The frontend application should now be running and accessible in your browser.

Using the System

1 .User Manual

For detailed instructions on how to use all features of the system, please refer to the complete user manual: User Manual: Click Here to open the User Manual

2. Quick Start

After installation, follow these steps to begin using the system:

1. Access the application at http://localhost:5173/Marito
2. Create a user account or log in with existing credentials
3. Refer to the user manual for detailed guidance on specific functions