GITGOOD

Help gitGood, gitBetter

TECHNICAL INSTALLATION

DEMO 4

WEATHER TO WEAR

EPI-USE®

KYLE LIEBENBERG
DIYA BUDHIA
ALISHA PERUMAL

IBRAHIM SAID
TAYLOR SERGEL

INTRODUCTION

Weather to Wear is a full-stack application designed to help users determine appropriate clothing based on current weather conditions. The system consists of two main components that require separate installation, and configuration

SYSTEM ARCHITECTURE OVERVIEW

BACKEND APPLICATION(app-backend/):

- Node.js server with Express.js framework
- PostgreSQL database for data persistence
- RESTful API endpoints for weather data and user preferences
- Docker containerization for consistent deployment
- Jest testing framework for unit and integration tests

FRONTEND APPLICATION(app-mobile/):

- React Native for cross-platform interfaces
- API integration with backend service
- User interface for weather display and clothing recommendations
- Cypress for end-to-end testing

Before proceeding with the installation, you will need to set up both the backend server environment and the mobile development environment. This guide provides step-by-step instructions for Windows, macOS, and Linux operating systems.

PREREQUISITES

REQUIRED SOFTWARE VERSIONS

SOFTWARE	Minimum Version	Recommended Version	Purpose
Node.js	18.0.0	20.9.0 LTS	JavaScript runtime
npm	9.0.0	10.2.0	Package Manager
PostgreSQL	13.0	15.4	Database system
Docker	20.10.0	24.0.6	Containerization
Git	2.30.0	2.42.0	Version Control

DEVELOPED TOOLS

TOOL	Version	Purpose
Visual Studio Code	Latest	Code editor with extensions
Postman	Latest	API testing

INSTALLATION INSTRUCTIONS BY OPERATING SYSTEM

WINDOWS 10/11

STEP 1: INSTALL NODE.JS

- Visit https://nodejs.org/en/download
- Download the Windows Installer (.msi) for Node.js v20.9.0 LTS
- Run the installer and follow the setup wizard
- Verify installation:

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

STEP 2: INSTALL GIT

- Download Git from https://git-scm.com/download/win
- Run the installer with default settings
- Verify installation:

```
git --version
```

STEP 3: INSTALL POSTGRESQL

- Download PostgreSQL 15.4 from https://www.postgresql.org/download/windows/
- Run the installer and remember your superuser password
- Default port: 5432
- Add PostgreSQL bin directory to PATH
- Verify installation:

```
psql --version
```

STEP 4: INSTALL DOCKER DESKTOP

- Download from https://www.docker.com/products/docker-desktop
- Install and restart your computer
- Start Docker Desktop
- Verify installation:

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

MAC OS

STEP 1: INSTALL HOMEBREW

```
/bin/bash -c "$(curl -fsSL
https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

STEP 2: INSTALL NODE.JS

```
brew install node@20
brew link node@20
node --version
npm --version
```

STEP 3: INSTALL POSTGRESQL

```
brew install postgresql@15
brew services start postgresql@15
psql --version
```

STEP 4: INSTALL DOCKER

```
brew install --cask docker
# Start Docker Desktop from Applications
docker --version
docker-compose --version
```

STEP 5: INSTALL GIT

```
brew install git
git --version
```

LINUX (UBUNTU)

STEP 1: UPDATE PACKAGE MANAGER

```
sudo apt update && sudo apt upgrade -y
```

STEP 2: INSTALL NODE.JS

```
curl -fsSL https://deb.nodesource.com/setup_20.x | sudo -E bash -
sudo apt-get install -y nodejs
node --version
npm --version
```

STEP 3: INSTALL POSTGRESQL

```
sudo apt install postgresql-15 postgresql-contrib
sudo systemctl start postgresql
sudo systemctl enable postgresql
sudo -u postgres psql -c "SELECT version();"
```

STEP 4: INSTALL DOCKER

```
sudo apt install apt-transport-https ca-certificates curl software-
properties-common
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key
add -
sudo add-apt-repository "deb [arch=amd64]
https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
sudo apt update
sudo apt install docker-ce docker-ce-cli containerd.io docker-compose-
plugin
sudo usermod -aG docker $USER
# Log out and back in for group changes to take effect
docker --version
docker compose version
```

STEP 5: INSTALL GIT

```
sudo apt install git
git --version
```

INSTALLATION

CLONE THE REPOSITORY

1. Open your terminal/ command prompt.
2. Navigate to your desired project directory.
3. Clone the main branch:

```
git clone https://github.com/your-username/weather-to-wear.git
cd weather-to-wear
```

4. Verify the repository structure:

```
ls -la
```

5. You should see the following structure:

```
weather-to-wear/
├── app-backend/
├── app-mobile/
├── .github/
├── docs/
├── infra/
└── README.md
```

BACKEND INSTALLATION

STEP 1: NAVIGATE TO THE BACKEND DIRECTORY

```
cd app-backend
```

STEP 2: INSTALL DEPENDENCIES

```
npm install
```

This will install all the dependencies listed in package.json, including:

- express@4.18.2
- postgresql (pg@8.11.3)
- jest@29.7.0
- dotenv@16.3.1
- cors@2.8.5

STEP 3: DATABASE SETUP

1. Create Database User (PostgreSQL):

```
# Windows
psql -U postgres

# macOS/Linux
sudo -u postgres psql
```

2. In PostgreSQL console:

```
CREATE USER weather_user WITH PASSWORD 'your_password';
CREATE DATABASE weather_to_wear OWNER weather_user;
GRANT ALL PRIVILEGES ON DATABASE weather_to_wear TO
weather_user;
\q
```

3. Run Database Migrations:

```
npm run migrate
```

STEP 4: VERIFY BACKEND INSTALLATION

```
npm test
```

FRONTEND INSTALLATION

STEP 1: NAVIGATE TO THE FRONTEND DIRECTORY

```
cd ../app-mobile
```

STEP 2: INSTALL DEPENDENCIES

```
npm install
```

STEP 3: VERIFY FRONTEND INSTALLATION

```
npm test
```

DOCKER INSTALLATION

STEP 1: BUILD AND START SERVICES

```
# From project root directory
docker-compose up --build
```

STEP 2: VERIFY SERVICES

```
docker-compose ps
```

You should see:

- weather-backend (Port 3000)
- weather-database (Port 5432)
- weather-frontend (Port 3001)

DEPLOYMENT AND RUNNING

DEVELOPMENT ENVIRONMENT

OPTION 1: MANUAL START

1. Start Backend Server - the backend will start on http://localhost:3001

```
cd app-backend
npx ts-node src/server.ts
```

2. In a new terminal, start the frontend - the frontend will start on http://localhost:3001

```
cd app-mobile
npm start
```

OPTION 2: DOCKER DEVELOPMENT

```
# From project root
docker-compose -f docker-compose.dev.yml up
```

PRODUCTION DEPLOYMENT

STEP 1: BUILD APPLICATIONS

```
# Backend
cd app-backend
npm run build

# Frontend
cd ../app-mobile
npm run build
```

STEP 2: PRODUCTION DOCKER DEPLOYMENT

```
# From project root
docker-compose -f docker-compose.prod.yml up -d
```

RUNNING TESTS

Backend Tests:

```
cd app-backend
npm run test                  # Unit tests
npm run test:integration      # Integration tests
npm run test:coverage         # Coverage report
```

Frontend Tests:

```
cd app-mobile
npm test                    # Unit tests
npm run test:e2e           # End-to-end tests with Cypress
```