Getting Started & Developer Setup Guide

Welcome to the **Paging Polly (Polly Do)** codebase template! This guide will walk you through setting up your environment, running the app locally, compiling it for production, and modifying it using modern "vibe coding" (AI-assisted coding).

🤖 Non-Coder Quick Start: Let AI Guide You!

If you are not a coder and need step-by-step guidance, you can use AI coding assistants like **Antigravity**, **Codex**, or **Claude Desktop** to do the work for you. A cheaper or smaller AI model (such as Gemini Flash or Claude Haiku) is perfectly sufficient for helping you through the initial installation and configuration!

⚠️ IMPORTANT: Install Requirements First!

Before running the setup prompt, please make sure you install the basic requirements (Node.js and VS Code) listed in the sections below! Once they are installed, copy and paste the appropriate setup prompt into your AI sidebar.

💻 Windows Setup Prompt

Simply copy and paste the following prompt into your AI sidebar to get started on Windows:

> ***Hey, can you look at `GETTING_STARTED.md` and help walk me through the entire process of installing requirements and creating the installer. I am on a Windows machine. Please run the environment verification tool using the verify setup skill (`node .agents/skills/dev-setup/scripts/verify-env.js`) first to check if everything is installed correctly. I am not a coder and need full hand holding through this. Can you please specifically tell me what I need to do versus what you can do for me. Thank you!***

🍎 macOS Setup Prompt

Simply copy and paste the following prompt into your AI sidebar to get started on macOS:

> ***Hey, can you look at `GETTING_STARTED.md` and help walk me through the entire process of installing requirements and creating the installer. I am on a macOS machine. Please check if Xcode Command Line Tools are installed, and if not, run the command `xcode-select --install` to start their installation. Also, make sure to make the `.sh` scripts (`run-dev.sh`, `build-app.sh`) executable using `chmod +x`. Please run the environment verification tool using the verify setup skill (`node .agents/skills/dev-setup/scripts/verify-env.js`) to check if everything is installed correctly before trying to build the installer. I am not a coder and need full hand holding through this. Can you please specifically tell me what I need to do versus what you can do for me. Thank you!***

🛠️ Editor Recommendation: VS Code

We recommend using **Visual Studio Code (VS Code)** as your code editor. It is:

- 100% free and cross-platform (Mac & Windows).
- Out-of-the-box support for Javascript, HTML, CSS, and React.
- Home to the best AI plugins for vibe coding.
- Includes an integrated terminal.

> ***Download VS Code:*** ***code.visualstudio.com*** *(https://code.visualstudio.com/)*

⚙️ Operating System Setup

Because this application relies on a local SQLite database (`better-sqlite3`), Node.js needs to compile native C++ database components on your machine. Follow the setup for your OS below.

💻 Windows Setup Steps

1. **Install Node.js LTS (v18 or newer):**
   - Download the installer from **nodejs.org** (https://nodejs.org/).

- **Crucial:** During installation, check the box that says: **"Automatically install the necessary tools. Note that this will also install Chocolatey..."**. This installs Python and compiling tools automatically.

2. **Install C++ Compiler (If dependencies fail to install):**
   - If installing fails with build errors, download the **Visual Studio Build Tools** from **visualstudio.microsoft.com/visual-cpp-build-tools/** (https://visualstudio.microsoft.com/visual-cpp-build-tools/).
   - Select the **Desktop development with C++** workload and install it.

🍎 macOS Setup Steps

1. **Install Node.js LTS (v18 or newer):**
   - Download and run the installer from **nodejs.org** (https://nodejs.org/).

2. **Install Xcode Command Line Tools:**
   - Open the **Terminal** app.
   - Type the following command and press Enter:

     ```
     xcode-select --install
     ```

   - Click **Install** on the pop-up and agree to the license terms.

3. **Make Scripts Executable (CRITICAL for macOS):**
   - Open the **Terminal** app, change directory to this folder, and run the following command to allow macOS to run the build/run scripts:

     ```
     chmod +x run-dev.sh build-app.sh
     ```

🐙 Git & GitHub Setup (For Version Control)

To track changes and upload your code to GitHub:

1. **Download and Install Git:**
   - **Windows:** Download from **git-scm.com/download/win** (https://git-scm.com/download/win) and run the installer (keep the default options).

- **macOS:** Git is installed automatically when you run the Xcode Command Line Tools step above.

2. **Create a GitHub Account:** Sign up at **github.com** (https://github.com/) if you haven't already.
3. **Log in to GitHub on your PC:** Open your terminal/VS Code command prompt and run:

```
git config --global user.name "Your Name"
git config --global user.email "your.email@example.com"
```

🚀 Running & Building the Application

1. Install Dependencies

Before running or building the application, install the required dependencies by running this command in your terminal within the `electron-app` directory:

```
npm install
```

2. Verify Environment & Installation

After installing the dependencies, you should verify that everything was installed correctly and there are no native module errors. Run this command from the project root directory:

```
node .agents/skills/dev-setup/scripts/verify-env.js
```

This script will test if Node, Git, `node_modules`, and the native C++ database components (`better-sqlite3`) are correctly compiled and integrated. If there are any errors, it will let you know how to fix them!

3. Run in Development Mode

This will start the live-reload development server.

- **Windows:** Double-click `run-dev.bat`.

- **macOS:** Right-click `run-dev.sh` -> Open With -> Terminal (or run `./run-dev.sh` in your terminal).

4. Build Your Installer / Executable

This packages your source code, assets, and compiled SQLite database into a production-ready application installer.

- **Windows:** Double-click `build-app.bat` (Generates a Windows `.exe` setup file in the `release/` folder).
- **macOS:** Run `./build-app.sh` in the terminal (Generates a macOS `.dmg` and `.zip` file in the `release/` folder).

🔮 Vibe Coding Guide (AI-Assisted Coding)

"Vibe Coding" means writing prompts describing what you want instead of writing lines of code yourself. Let the AI write, edit, debug, and build the program for you.

Recommended VS Code Extensions

Install these AI tools inside VS Code:

1. **Cline:** A highly capable AI agent built directly inside VS Code. (We recommend Cline specifically over Roo Code or Claude Dev).
2. **OpenAI Codex:** OpenAI's official coding helper.

Note: We do not recommend Cursor or GitHub Copilot for this specific workflow.

How to Reference Files

- **Use the `@` Symbol:** Inside extensions like **Cline**, typing `@` allows you to select and attach a specific file (e.g. `@package.json` or `@App.jsx`) to focus the AI's search.
- **No Coding Knowledge Required:** If you don't know where a file is or how the code works, don't worry! You can simply describe the change you want (e.g., "*Change the layout of the project screen to have a green title bar*"), and the AI will find the correct files for you.

Tips for Successful Vibe Coding

- **Work incrementally:** Do not ask the AI to build 10 features at once. Ask for one small change, test it using your local dev script, then proceed to the next change.
- **Let the AI read files first:** Ask the AI: "*Analyze the structure of the database files in* `electron/database/` *and tell me how projects are stored before editing.*"

🛠️ Debugging & Fixing Errors with AI

If something breaks, doesn't load, or behaves unexpectedly:

1. **Send Error Messages to the AI:** If the terminal or setup script shows a failure, copy the entire output and paste it directly to the AI, asking: "*What does this error mean and how can we fix it?*"
2. **Take and Upload Screenshots:** Visualizing the UI issue is extremely helpful. Take a screenshot of the broken interface and upload it to your AI agent.
3. **Inspect the Console (Toggle Developer Tools):**
   - While running the app, locate the top program menu bar.
   - Go to **View** -> click **Toggle Developer Tools** (you may need to click it a few times to open).
   - Look at the **Console** tab that slides open.
   - **Crucial:** Look for any **red text** (error logs). Copy and paste this red text directly into your AI assistant.

⚙️ Automation with AI Skills

This workspace has pre-defined automation rules (**AI Skills**). You do not need to memorize commands! You can simply ask the AI in plain English to perform these tasks:

- **To Verify Setup:** Just ask the AI to "*Run the dev setup check*" or "*Verify my environment*". It will run the environment verifier to ensure there are no build errors.
- **To Build the App:** Just type "*Build the app*" or "*Create the installer*" (works for both Windows & Mac). The AI will automatically handle the build skill.

- **To Push to GitHub:** Once you have provided your GitHub URL, just type "*Push my changes to GitHub*". The AI will automatically update the changelog, make logical commits, and push.
- **To Run the Dev Server:** After making any feature changes, you can ask the AI to "*Start development mode*" or "*Run the app in dev*". Use this tool to verify your changes work correctly before building the final installer.