Step-by-Step Guide - ProcureHub

This guide explains how to set up and run the ProcureHub project on your local machine from scratch.

---

1) Install Node.js v20.12.2

Node.js is required to run the backend (Node.js/Express) and frontend (React) applications.

Instructions:

- Go to the official Node.js website: https://nodejs.org/en/download
- Click on the **"Previous Releases"** section
- Download and install version **v20.12.2** for your operating system (Windows, macOS, Linux)
- During installation, make sure to check the box that says **"Add to PATH"**
- After installation, open your terminal and check the version:

```
node -v
npm -v
```

You should see an output similar to this:

```
v20.12.2
10.5.0
```

---

2) Install PostgreSQL

Install PostgreSQL on your machine and setup pgAdmin

Instructions:

- Go to this website and download the appropriate version of PostgreSQL for your device
- Follow the download wizard instructions and make sure to **remember your password**
- Create a database inside pgAdmin 4 - name of the database is optional but make sure to **remember the name you set**

---

3) Clone the Backend and Frontend Repositories

Clone both repositories from GitHub into a local directory.

Instructions:

Open your terminal and run:

```
git clone https://github.com/Procure-Hub-Org/procure-hub-be.git
git clone https://github.com/Procure-Hub-Org/procure-hub-fe.git
```

This will create two folders:

- `procure-hub-be` (backend)
- `procure-hub-fe` (frontend)

---

4) Run npm `install` in Both Repositories

This command installs all required packages (defined in `package.json`) for both the backend and frontend.

Instructions:

```
cd procure-hub-be
npm install

cd ../procure-hub-fe
npm install
```

You should see a `node_modules` folder created in each project. This means all dependencies are successfully installed.

---

5) Copy `.env.example` to `.env` and Configure It

Environment variables are used to define database credentials, ports, secrets and other values that are dependent on the running environment.

Instructions:

In the **`procure-hub-be` folder**, run:

```
cp .env.example .env
```

Then open the `.env` file in a text editor (like VS Code), and fill in the values. Example:

Required variables:

1. Database:

   ```
   DB_HOST=127.0.0.1        # database host
   DB_PORT=5432             # database port (5432-postgres default)
   DB_NAME=step_2_name      # database name
   DB_USER=postgres         # database user
   DB_PASS=step_2_password  # database password
   ```

   Make sure the values match your local PostgreSQL setup.

2. Frontend application

   ```
   FRONTEND_URL=http://localhost:5173 # url on which your frontend application is running (http://localhost:5173 is for local react environment)
   ```

Optional variables (without these variables the application will work, but without some functionalities):

1. For sending email notifications during live auction:
   ```
   EMAIL_USER=your_email_address
   EMAIL_PASS=your_email_password
   ```

2. For storing files on supabase bucket:
   ```
   SUPABASE_URL=your_project_url
   SUPABASE_SECRET_KEY=your_supabase_secret_key
   SUPABASE_BUCKET_NAME=your_supabase_bucket_name
   ```

These values can typically be found by opening your supabase project and going to **Project Settings** -> **Configuration** -> **Data API** (more about supabase buckets)

3. In-application caching for faster retrieval during live auctions and data from database generally:
   `REDIS_URL=your_redis_url` (more about local redis setup)

In the **`procure-hub-fe` folder**, run:

```
cp .env.example .env
```

Then open the `.env` file in a text editor (like VS Code), and fill in the required values. Example:

```
VITE_API_URL=http://localhost:3000     # url on which your backend
application is running
VITE_WEBSOCKET_URL=ws://localhost:3000 # for websocket communication
```

---

6) Run Migrations: `npx sequelize-cli db:migrate`

This creates the database tables according to Sequelize models. Make sure pgAdmin is running in background.

Instructions:

```
cd procure-hub-be
npx sequelize-cli db:migrate
```

If successful, you'll see a list of applied migrations in the terminal. You can confirm the tables exist using **pgAdmin**.

---

7) Run Seeders: `npx sequelize-cli db:seed:all`

Populate the database with sample data (e.g. categories) from your backend project folder.

Instructions:

```
npx sequelize-cli db:seed:all
```

---

8) Create Admin User: `npm run add-admin`

The app does not have a default admin. You must manually create one using CLI in your backend project folder.

Instructions:

```
npm run add-admin
```

The terminal will prompt you to enter:

- Admin email
- Password
- First and Last Name

This data is stored securely in the database. Using these credentials, you will be able to login in on the page.

---

9) Start Backend: `node index.js`

Now that the backend is configured and the database is ready, start the backend server.

Instructions:

```
node index.js
```

If everything is set up correctly, you'll see something like:

```
Server running on port 3000
```

Keep this terminal open and running.

---

10) Start Frontend: `npm run dev`

Open a new terminal window or tab and start the frontend React app from your frontend project folder.

Instructions:

```
cd procure-hub-fe
npm run dev
```

This will start the application at `http://localhost:5173` by default.

If the backend is also running, the frontend will be able to fetch data and you'll see the landing page.

---

11) Open the App in Your Browser

Navigate to:

```
http://localhost:5173
```

You can now:

- Register as a buyer or seller
- Log in with your admin credentials
- Interact with the system like in production

Final Notes

- Make sure PostgreSQL is running locally.
- Make sure all required environment variables are populated with correct values
- Ports used:
    - Backend: typically `3000`
    - Frontend: Vite default `5173`
- If ports are occupied, update them in `.env` and `vite.config.js`