perplexity

Using this project plan give me detailed instructions step by step for an inexperienced dev for Phase 1.1 Environment & Repository Setup (Day 1) in markdown

Certainly! Here are step-by-step instructions for **Phase 1.1: Environment & Repository Setup (Day 1)** from your project plan, written for an inexperienced developer. These instructions will guide you through setting up the GitHub repository, project structure, initializing frontend and backend, and documenting the environment[1].

Phase 1.1: Environment & Repository Setup (Day 1)

Goal:
Set up the basic project structure and environment so both the frontend and backend can be started locally, with the repository organized and documented.

1. Create a GitHub Repository

1. **Go to GitHub:**
   - Visit github.com and log in to your account.
2. **Create a New Repository:**
   - Click the "+" button in the top right and select "New repository."
   - Name it `LifeTriage` (or your preferred name).
   - Add a description if you wish.
   - Choose "Public" or "Private."
   - **Do not** initialize with a README, .gitignore, or license yet (we'll do this manually).
   - Click "Create repository."

2. Set Up the Project Folder Structure

C:\Users\TechFast Australia\Desktop\LifeTriage

1. **Open Your Terminal or Command Prompt:**
   - Navigate to where you want your project to live (e.g., `cd Documents/Projects`).
2. **Create the Main Project Folder:**
   - Run:

```
mkdir LifeTriage
cd LifeTriage
```

3. **Create Subfolders:**
   - Run:
     ```
     mkdir frontend backend docs
     ```
   - This matches the structure shown in your plan.

3. Initialize Frontend with React + Vite

1. **Enter the Frontend Folder:**
   - Run:
     ```
     cd frontend
     ```
2. **Initialize a New React Project with Vite:**
   - Run:
     ```
     npm create vite@latest .
     ```
   - When prompted:
     - Select “React” as the framework.
     - Choose “JavaScript” or “TypeScript” (JavaScript is fine for beginners).
     - Press Enter to confirm the project name (it should be the current folder, `frontend`).
3. **Install Dependencies:**
   - Run:
     ```
     npm install
     ```
4. **Test the React App:**
   - Run:
     ```
     npm run dev
     ```
   - Open the provided local URL in your browser to confirm the app is running.
   - Press `Ctrl+C` in the terminal to stop the server when done.

4. Initialize Backend with Flask

1. **Go Back to the Main Project Folder:**
   - Run:
     ```
     cd ..
     cd backend
     ```
2. **Create a Virtual Environment (Recommended):**
   - Run:
     ```
     python -m venv venv
     ```
   - **Activate the Virtual Environment:**
     - **Windows:**
       ```
       venv\Scripts\activate
       ```
     - **Mac/Linux:**
       ```
       source venv/bin/activate
       ```
3. **Install Flask:**
   - Run:
     ```
     pip install flask
     ```
4. **Create a Basic Flask App:**
   - Create a file named `app.py` in the `backend` folder.
   - Add the following code:
     ```
     from flask import Flask
     app = Flask(__name__)

     @app.route('/')
     def hello():
         return "Hello, LifeTriage!"

     if __name__ == '__main__':
         app.run(debug=True)
     ```
5. **Test the Flask App:**
   - Run:
     ```
     python app.py
     ```
   - Open `http://127.0.0.1:5000` in your browser to confirm the message appears.

- Press `Ctrl+C` in the terminal to stop the server when done.

5. Create a `.gitignore` File

1. **Go Back to the Main Project Folder:**
   - Run:
     ```
     cd ..
     ```
2. **Create a `.gitignore` File:**
   - Run:
     ```
     touch .gitignore
     ```
   - Open the file and add these lines:
     ```
     node_modules/
     venv/
     .env
     __pycache__/
     *.pyc
     *.pyo
     *.pyd
     .DS_Store
     ```
   - This prevents unwanted files from being tracked by Git.

6. Set Up Local Development Environment Variables

1. **Create a `.env` File in the Backend:**
   - In the `backend` folder, run:
     ```
     touch .env
     ```
   - For now, leave it empty. You'll add environment-specific settings later.

7. Create Initial README with Setup Instructions

1. **Create a** README.md **File:**
   - In the main project folder, run:
     ```
     touch README.md
     ```
2. **Open** README.md **and Add Basic Instructions:**
   - Example content:

```
# LifeTriage

A voice-first task management application.

## Setup

### Frontend
1. `cd frontend`
2. `npm install`
3. `npm run dev`

### Backend
1. `cd backend`
2. `python -m venv venv`
3. Activate the virtual environment:
   - **Windows:** `venv\Scripts\activate`
   - **Mac/Linux:** `source venv/bin/activate`
4. `pip install -r requirements.txt` (once you have one)
5. `python app.py`
```

8. Push to GitHub

1. **Initialize Git in the Main Project Folder:**
   - Run:
     ```
     git init
     ```
2. **Add All Files:**
   - Run:
     ```
     git add .
     ```
3. **Commit Your Changes:**
   - Run:
     ```
     git commit -m "Initial project setup"
     ```
4. **Link to Your GitHub Repository:**
   - Go to your GitHub repository and copy the remote URL (looks like `https://github.com/yourusername/LifeTriage.git`).
   - Run:
     ```
     git remote add origin https://github.com/yourusername/LifeTriage.git
     ```
5. **Push to GitHub:**
   - Run:

```
git push -u origin main
```

9. Create Development Environment Documentation

1. **In the `docs` Folder, Create a File:**
   - Run:
     ```
     cd docs
     touch development.md
     ```
2. **Add Notes About Your Setup:**
   - Example content:
     ```
     # Development Environment Setup

     - **Node.js** installed (for frontend)
     - **Python 3.9+** installed (for backend)
     - **Git** installed
     - **Vite** for React frontend
     - **Flask** for Python backend
     - **Virtual environment** for Python dependencies
     ```

Success Criteria

- **Both frontend and backend can be started locally.**
- **Repository is properly organized and documented.**
- **All files are pushed to GitHub.**

You have now completed Day 1 of Phase 1.1 for the LifeTriage project![1]

⁂

1. lifetriage_revised_plan.md