NOTE: Follow the project structure and instructions carefully, BUT you DON'T need to complete the entire project.

🎯 HandsOn – A Community-Driven Social Volunteering Platform

📖 Project Description

HandsOn is a community-driven social volunteering platform that connects individuals with meaningful social impact opportunities. Users can discover and join **volunteer-driven events**, post **requests for community help**, **form teams** for large-scale initiatives, and **track their impact** with contributions logged on a personal and team level.

This platform is designed to **encourage social responsibility, community collaboration, and proactive engagement** in volunteer work. It also aims to **reward participation**, making volunteering more structured, engaging, and impactful.

💡 **Think of it as a "GitHub for social work"—where people contribute their time instead of code, building real-world impact together.**

🎯 MVP Features (Minimum Viable Product)

Each feature below is explained in **detail** to ensure clarity and ease of implementation.

1️⃣User Registration & Profile Management

📌 Core Requirements:

- Users must be able to **sign up and log in** securely with an email and password.
- Each user profile should include **basic information**, their **skills**, and **causes they support** (e.g., environmental, education, healthcare).
- Users can edit their profile and view their **volunteer history & contributions**.

🎯 Expected Outcome:

- A user-friendly **profile dashboard** that encourages active participation in social causes.

2️⃣Discover & Join Volunteer Events

📌 Core Requirements:

- 📅 **Event Creation** – Users or organizations can create volunteer events by providing details like **title, description, date, time, location, and category**.

- 🔍 **Event Listing & Filters** – A **public event feed** allows users to browse **upcoming volunteer events** and filter them by **category, location, and availability**.
- ✅ **One-Click Registration** – Users can **join an event** instantly with a **"Join Event"** button, which adds them to the attendee list.

- 📌 **Event & Post Differentiation** –
  - **Events** are structured with a **fixed time and date** for users to participate.
  - **Community Help Posts** are **open-ended** requests for ongoing support (e.g., "Need volunteers to tutor weekly").

🎯 **Expected Outcome:**
Users can **easily find, register, and participate** in social events while differentiating between **scheduled events** and **ongoing community help posts**. 🚀

3️⃣Community Help Requests

📌 **Core Requirements:**

- Any user or organization should be able to **post a request for help** (e.g., “We need volunteers to distribute winter clothes to homeless people.”)
- Other users should be able to **offer help and coordinate through comments or private messaging**.
- Requests should have an **urgency level** (e.g., **low, medium, urgent**) to help prioritize volunteer responses.

🎯 **Expected Outcome:**

- A dynamic help request board where community members can **proactively offer assistance**.

4️⃣Form Teams & Group Initiatives - (BONUS POINT)

📌 **Core Requirements:**

- Users should be able to **form teams** to organize long-term initiatives (e.g., “Team Green Warriors” for tree planting).
- Teams can have **private or public** membership.

  In the **HandsOn** platform, when users form teams for volunteering projects, they can choose between two types of memberships:

  - 🔒 **Private Teams:** Only invited members can join. The team and its activities are **not visible** to others.

        - Example: A small group of friends forming a **private team** to help a specific charity.
    - 🌍 **Public Teams:** Open for **any user** to join. The team and its projects are **visible** on the platform.
        - Example: A group creating a **public team** for city-wide environmental cleanups, allowing anyone to participate.
- Each team should have a **dashboard** displaying team members, events, and achievements.
- A **leaderboard** showcasing the most active teams.

🎯 **Expected Outcome:**

- A **collaborative space** for individuals to build **sustainable volunteering communities**.

5️⃣ Impact Tracking & Social Recognition - (BONUS POINT)

📌 **Core Requirements:**

- 🕒 **Log Volunteer Hours** – Users can log hours after attending an event by clicking **"Log Hours"** and entering time spent.
- ✅ **Auto & Peer Verification** – Hours are **auto-verified** for platform-created events, while other logs require **2-3 peer verifications**.
- 🎯 **Point-Based System** – Users earn **5 points per hour** volunteered, updated in real-time.
- 🏅 **Auto-Generated Certificates** – Certificates are **automatically generated** when users **reach milestones** (e.g., 20, 50, 100 hours).
- 📊 **Public Leaderboard** – A **leaderboard ranks** the most active volunteers based on verified hours.

🎯 **Expected Outcome:**
Users can **track and showcase** their impact, making their efforts **visible, rewarding, and engaging** without needing an admin. 🚀

🔧 Tech Stack

This project will be built using the following technologies:

- **Frontend:** React.js
- **Backend:** Node.js (Express.js) or Python or Anything you prefer
- **Database:** PostgreSQL or MySql or MongoDB
- **Authentication:** JWT-based auth
- **API Communication:** REST API

📜 Submission Guidelines (GitHub Repository)

To ensure a **structured and professional** submission, follow the **best engineering practices** outlined below.

1️⃣ Create a GitHub Repository

- **Repository Name:** Name your repository clearly and professionally (e.g., `hands-on-volunteering-platform`).
- Ensure the repository is **public** for review.
- Use **README.md** for documentation.

2️⃣ Follow Commit Guidelines

Adopt **clear, structured commit messages** following **Conventional Commits**:

✨ feat: add event listing API
🐛 fix: resolve event registration bug
📝 chore: update README with new setup instructions
🔧 refactor: optimize database queries for better performance

📌 Commit Best Practices:

- Write **small, incremental commits** instead of **large, monolithic ones**.
- Use **imperative mood** (e.g., “Add event filters” instead of “Added event filters”).

3️⃣ Use a Branching Strategy

- Use `master` **only for stable code**.
- Create **feature branches** (e.g., `feature/add-event-listing`).
- Merge feature branches into `master` via **pull requests (PRs)**.

4️⃣ Update the README File

Your **README.md** should include the following:

📌 **1. Project Overview** – A short summary of what the project does.
📌 **2. Technologies Used** – A list of the stack (Node.js, PostgreSQL, React, etc.).
📌 **3. Features** – A bullet-point breakdown of key features.
📌 **4. Database Schema** – A visual representation of the database structure.
📌 **5. Setup Instructions** – How to install dependencies, configure the environment, and run the server.
📌 **6. API Documentation** – List of **API endpoints**, their parameters, and expected responses.
📌 **7. Running the Project** – How to run the project **locally and in production**.

📢 **Final Submission Step**

Once your project is completed and pushed to GitHub:

✅ **Submit your GitHub repository link via the official Google Form** (Link: https://docs.google.com/forms/d/e/1FAIpQLSfyrb2rOvcLggNM64KAIUUmBZQjLWMZSaCysTsrGAyIDhogfg/viewform?usp=header).

Failure to submit via the form will result in disqualification.

Cheers! Enjoy the task.