CSE446: Project Part 2

Deadline: 21/08/25
Title: Disaster Recovery Training Management – DApp Development

In the first part of the project, you developed the smart contract for managing participant data and organizing disaster recovery training sessions. Now, your task is to build a complete decentralized application (DApp) that allows users to interact with your smart contract through an intuitive web-based interface.

There will be **three types of users** in the system:

1. **Admin**
2. **Participant** (Someone attending training)
3. **Trainer** (Conducts training sessions)

Features to Implement

1. User Registration

- Create a registration form for **Admin**, **Participant**, and **Trainer** to register themselves via the smart contract.
- The participant registration must include:
    - ID (auto-generated unique ID)
    - Name
    - Age
    - Gender
    - District
    - Training Interest (must be one of: `first_aid`, `shelter_rebuild`, `food_safety`)

- Has Completed Training (default = `false`)

---

2. Add & Update Participant Data (Admin Only)

- Create an interface where only **Admin** can update participant details:
  - Update `training_interest` (**must be one** of: `first_aid`, `shelter_rebuild`, `food_safety`)
  - Update `has_completed_training` (boolean — once `true`, cannot revert to `false`)
- If invalid values are entered, the update should fail and display an **error message** in the DApp interface (**no terminal checking**).

---

3. Book Training Slot (Participants Only)

- Participants should be able to book a training slot with a trainer.
- **Payment**: A fixed booking fee must be transferred from the **participant's MetaMask** account to **any admin's account**.
- Available slots should be **shown in the DApp before booking**.
- Example slots:
  - Slot 1: 9:00 AM – 9:30 AM
  - Slot 2: 9:31 AM – 10:00 AM
  - Slot 3: 10:01 AM – 10:30 AM
- If a trainer is already booked for a slot, that slot should be unavailable for further booking.
- After a successful transaction, display a **"Booking Successful"** message **without refreshing** the page (must be a real-time update).

---

4. View Training Schedule (All Users)

- There should be a separate section where **all users** can view the complete training schedule for every trainer.

---

5. Search & Sort Participants

- Allow users to search participants by **district** and display the count of participants per district in a **sorted list**.

---

6. Bonus Feature (Optional)
After completing the training, the participant will receive a digital certificate (PDF or image) that will be stored off-chain on IPFS. The smart contract should store the IPFS hash (CID) of the certificate linked to the participant's ID. This will allow anyone to verify and download the certificate through a public IPFS gateway.

Requirements:

1. **Only the admin can upload the certificate for a participant.**
2. **The actual certificate file (PDF/image) must be uploaded to IPFS manually or through the DApp frontend.**
3. **The contract should store only the IPFS CID to reduce on-chain storage costs.**
4. **Any user should be able to retrieve the certificate's CID and view it via a public IPFS gateway.**

Constraints & Tools

- **No external database** (e.g., MongoDB, MySQL) — all data must be stored in the smart contract.
- Must use **Ganache** and **Truffle** for smart contract deployment.
- Frontend: You can use plain **JavaScript**, or if you have prior experience, frameworks like **ReactJS**, **AngularJS**, or **VueJS** are allowed.
- Focus on **functionality over UI design** — no extra marks for appearance.