1. Role Assignment

Team Member	Role	Responsibilities
Shalon	**Senior Architect, Full Stack Lead**	- Design architecture - Build backend API - Implement critical logic (map, AI, SLUDI) - Code review - Guide juniors - Final integration
Gaindu and Shalon	**Mobile App Developer**	- Build React Native UI screens - Integrate map SDK - SOS button, report forms - Use mock API
Lehan and Pavith	**Web Dashboard Developer**	- Implement dashboard UI in React - Build forms for disaster management - Display data using mock API
All	**Documentation & Tester**	- Write `README.md`, `problem.md`, `solution.md` - Test mobile/web features - Assist with dummy data, screenshots, feedback

---

2. GitHub Monorepo Structure

```
national-disaster-platform/ #Need to come up with a name
├── mobile-app/                  # React Native
│   ├── src/
│   ├── assets/
│   ├── components/
│   └── ...
├── web-dashboard/              # MERN stack
│   ├── client/                 # React frontend
│   └── server/                 # Express backend API
├── mock-data/                  # JSON or CSV for dummy disasters, users
├── docs/
│   ├── problem.md
│   ├── solution.md
│   └── architecture.md
├── ci-cd/                      # GitHub Actions / Dockerfiles if needed
├── .gitignore
├── README.md
└── package.json (workspace root if using npm workspaces)
```

---

3. MongoDB Schema Design

`users` (citizens, responders, admins)

```
{
  "_id": "ObjectId",
  "name": "string",
  "email": "string",
  "role": "citizen | responder | admin",
  "phone": "string",
  "location": {
    "lat": "number",
    "lng": "number"
  },
  "is_verified": "boolean"
}
```

disasters

```
{
  "_id": "ObjectId",
  "type": "flood | landslide | cyclone",
  "severity": "low | medium | high",
  "description": "string",
  "location": {
    "lat": "number",
    "lng": "number"
  },
  "timestamp": "ISODate",
  "status": "active | resolved"
}
```

reports

```
{
  "_id": "ObjectId",
  "user_id": "ObjectId",
  "disaster_id": "ObjectId",
  "type": "food | shelter | danger | medical",
  "description": "string",
  "image_url": "string",
  "status": "pending | addressed",
  "timestamp": "ISODate"
}
```

sos_signals

```
{
  "_id": "ObjectId",
  "user_id": "ObjectId",
  "location": {
    "lat": "number",
    "lng": "number"
  },
  "message": "string",
  "timestamp": "ISODate",
  "priority": "low | medium | high" }
```

resources

```
{
  "_id": "ObjectId",
  "type": "medicine | food | shelter",
  "quantity": "number",
  "status": "available | dispatched | depleted",
  "location": {
    "lat": "number",
    "lng": "number"
  }
}
```

chat_logs

```
{
  "_id": "ObjectId",
  "user_id": "ObjectId",
  "query": "string",
  "response": "string",
  "timestamp": "ISODate"
}
```

---

4. GitHub Issues & Task Breakdown

Issue	Description	Difficulty	Label
#1 Setup GitHub Monorepo & Folders	Init structure, add README	Easy	setup
#2 Mobile App: Risk Map UI	Color-coded disaster map	Medium	mobile
#3 Mobile App: SOS Button + Geolocation	Send SOS with GPS	Medium	mobile
#4 Mobile App: Report Form (text/photo)	UI for need reporting	Medium	mobile
#5 Mobile App: AI Chatbot UI	Integrate dummy Rasa/Dialogflow chat	Medium	mobile, ai
#6 Web Dashboard: Disaster Input Form	Admin adds/edits disaster zones	Medium	web
#7 Web Dashboard: View Reports Map	Needs heatmap with filtering	Hard	web
#8 Web Dashboard: SOS Feed	Table of incoming SOS alerts	Medium	web
#9 Backend: REST API Endpoints	CRUD for disasters, SOS, reports	Hard	backend
#10 Backend: WebSockets for Alerts	Push real-time disaster updates	Hard	backend
#11 Backend: Mock NDX & SLUDI	JSON mocks for login/data fetch	Medium	backend
#12 AI: Dialogflow/Rasa Setup	Deploy Q&A for safety questions	Medium	ai
#13 Auth: SLUDI Login Mock	Citizen/admin login simulation	Medium	auth
#14 Docs: Problem/Solution/Setup	Write markdown files + screenshots	Easy	docs

---

5. Task Assignment Strategy (Week-wise)

Phase 1 (July 31 – Aug 2)

- Shalon: Set up backend folder/API base, define DB schema
- Shalon and Gaindu : Mobile UI screens (map, SOS, reports) – static only
- Lehan and Pavith : Web dashboard UI (form, disaster table)
- All : Start `README.md`, mock-data population

Phase 2 (Aug 3 – Aug 5)

- Shalon: Connect backend to MongoDB, implement REST APIs
- Shalon and Gaindu: Integrate SOS/report post via API, dummy chatbot
- Lehan and Pavith: Connect web forms to backend, show dummy data
- All : Finish docs, test all endpoints/screens

Final Days (Aug 6 – Aug 7)

- All: AI integration, finalize NDX/SLUDI mocks, fix bugs
- All: Final testing, screenshots, submit demo + docs

---

6. Collaboration & DevOps Best Practices

- **Branch Naming**:
  `feature/mobile-sos-button`, `fix/dashboard-map-crash`, `docs/readme-update`
- **Pull Requests**:
  Use PR templates, review teammate PRs before merging, tag you for final review
- **Daily Check-Ins**:
  Use a WhatsApp/Discord group + 10AM and 9PM updates (blockers, progress)
- **Testing Locally**:
  - Mobile: Expo or Android Studio emulator
  - Web: React + Node on localhost:3000 / 5000
- **Offline-first**:
  - Mobile app caches map tiles/data with local storage or SQLite
  - Use `NetInfo` to detect connection for sync alerts
- **Mocking DPI**:
  - SLUDI: mock OAuth token + JSON user info
  - NDX: static JSON with disasters, resources
  - PayDPI: simulate success with a button

Updated Technical Architecture