**Project Architecture Document: DXF Automation and Document Generation System**

**Project Objective:**

To design and develop a web-based application that allows users to input geographical and coordinate data to generate DXF files compatible with CAD software, facilitate planning team reviews, and upon approval, generate associated documentation with preview and download functionality.

**Project Description:**

This project involves creating an integrated system to streamline the process of:

1. Inputting coordinates and location data
2. Generating DXF files for CAD tools
3. Allowing a planning team to review submissions
4. Accepting or rejecting coordinate-based plans
5. Generating formal documentation upon approval, including certificates and supporting documents

**Phase-wise Development Steps**

**Phase 1: DXF File Generation (1 week)**

**Objective:** Capture user input and convert coordinates into a DXF file

**Steps:**

1. Design a form with fields: Location, Address, x1, y1, x2, y2, x3, y3, x4, y4
2. Integrate a client-side DXF generation library (dxf-writer) or call backend DXF creation API
3. Render polygon or shape using the provided coordinates
4. Generate and download the DXF file locally
5. Optionally auto-open DXF file in local CAD application using a backend trigger (native app or shell command)

**Deliverables:**

* Working input form
* DXF file download

**Phase 2: Review System (1 week)**

**Objective:** Enable planning team to accept or reject generated plans

**Steps:**

1. Display each submission in a review panel
2. Provide Accept/Reject buttons
3. Change view based on submission status
4. Trigger document generation flow only when status is "Accepted"
5. Passing the entered values into the document.

**Deliverables:**

* Submission list and review interface
* Accept/Reject workflow

**Phase 3: Document Generation and Preview (1 week)**

**Objective:** Automatically generate and preview documents upon approval

**Steps:**

1. On Accept action, trigger PDF generation scripts
2. Create:
   * Approval Certificate
   * Acknowledgement Form
   * Supporting Document 2
3. Show inline previews of generated PDFs
4. Allow individual and bulk download options

**Deliverables:**

* Automatically generated documents
* Preview and download functionality

**Phase 4: Deployment (1 week)**

**🛠 Tech Stack**

|  |  |
| --- | --- |
| **Layer** | **Technology** |
| Frontend | ReactJS, TailwindCSS, PDF.js |
| Backend | Flask or Node.js |
| DXF Library | dxf-writer (JS) or ezdxf (Python) |
| Document Generator | reportlab/weasyprint (Python) or pdf-lib (JS) |
| Deployment | Local server / Docker / Electron App |

**⏱ Timeline Estimate**

|  |  |  |
| --- | --- | --- |
| Phase | Feature | Estimated Time |
| Phase 1 | DXF Generator + Form | 3–4 days |
| Phase 2 | Review Interface + Status Flow | 3–4 days |
| Phase 3 | Document Generator + Preview + Download | 4–5 days |
| **Total** | Complete MVP | **10–13 working days** |

**🚀 Deployment Options**

* Local machine using npm, flask run, etc.
* Dockerized app for local testing and deployment
* Electron App (optional) for auto-opening DXF in CAD tools

**Detailed Architecture Overview:**

**1. Frontend (ReactJS):**

* Input form for location, address, and coordinates
* DXF generation trigger
* Accept/Reject buttons post-review
* PDF preview and download features

**2. Backend (Flask or Node.js):**

* RESTful APIs to handle form submissions, DXF file creation, status management, and document generation
* File handling and local storage

**3. DXF Generation Module:**

* Library: dxf-writer (client-side JS) or ezdxf (server-side Python)
* Converts coordinates into a structured DXF file format
* Automatically triggers download
* Optionally opens in installed local CAD tool using OS-level command

**4. Document Generation Module:**

* Templated PDF files populated with user data
* Library: reportlab, weasyprint, pdf-lib, etc.
* Preview using embedded PDF viewer