Assignment: Contract Management Platform (Full Stack)

Objective

Build an end-to-end Contract Management Platform that demonstrates your ability to design and implement a **complete full-stack system**, including frontend UI, backend APIs, data modeling, and controlled workflows.

This assignment evaluates product thinking, backend design, API structure, frontend integration, and overall system quality.

Deadline

Monday, 11:59 PM IST
Late submissions will not be evaluated.

Deliverables

1. Public GitHub repository
2. Running frontend + backend application
3. `README.md` including:
    - Setup instructions (frontend + backend)
    - Architecture overview
    - API design summary
    - Assumptions and trade-offs

Functional Requirements

1. Blueprint Management (Backend + Frontend)

A **Blueprint** is a reusable contract template.

Features

- Create a blueprint with configurable fields
- Supported field types:
    - Text
    - Date
    - Signature
    - Checkbox
- Each field must store:
    - Field type
    - Label

    - Position (x/y or grid)
- Persist blueprint data in a database

2. Contract Creation from Blueprint

- Select an existing blueprint
- Generate a contract instance from it
- Contract should inherit all blueprint fields
- Allow entering values for contract fields
- Persist contract data

3. Contract Lifecycle Management

Each contract must follow this lifecycle:

Created → Approved → Sent → Signed → Locked
Revoked (allowed after creation or sending)

Rules

- Lifecycle transitions must be enforced on the backend
- Invalid transitions should be rejected via API
- Locked contracts are immutable
- Revoked contracts cannot move forward
- Frontend must reflect current status and allowed actions

4. Contract Listing & Dashboard

Create a dashboard showing contracts in a table.

Table fields

- Contract name
- Blueprint name
- Current status
- Created date
- Actions (view, change state)

Contracts should be filterable or grouped by:

- Active
- Pending
- Signed

Backend Requirements

- RESTful APIs for:
    - Blueprint CRUD
    - Contract creation and retrieval
    - Contract lifecycle transitions

- Proper request validation
- Clear data models
- Meaningful error handling
- Any backend language/framework is allowed

Database:

- Any relational or NoSQL database is acceptable
- Schema design must be explained in README

Authentication is optional (can be mocked).

Frontend Requirements

- Consume backend APIs
- Create UI for:
    - Blueprint creation
    - Contract creation
    - Lifecycle actions
    - Contract listing dashboard
- UI design is not provided and must be created by the candidate
- Focus on clarity and usability over visual polish

Tech Stack

You may choose any stack. Justify your choices.

Preferred (not mandatory):

- React / Next.js with TypeScript
- REST APIs
- Proper state management
- Clean separation between frontend and backend
- Environment-based configuration

Evaluation Criteria

- Backend architecture and API design
- Data modeling and lifecycle enforcement
- Frontend–backend integration
- Code structure and readability
- UI clarity and workflow logic
- Git commit quality and documentation

Restrictions

- No copied code without understanding
- No hardcoded lifecycle bypasses

- Missing README will lead to rejection
- Incomplete backend logic will not be evaluated

Optional Enhancements

- API documentation (Swagger / Postman collection)
- Role-based actions (approver vs signer)
- Status timeline view
- Basic unit or integration tests
- Docker setup

Submission Instructions

Submit your application at: https://forms.gle/5SCUfjVjyQvzcffn9