Full-Stack Task: One-Time Secret Sharing App with Auth, Password Protection, and Access Logging (DB)

Goal:

Build a secure system where users can:

- Register/login
- Create password-protected, one-time-view secrets
- Share a secure link
- Viewers must enter the correct password to view the secret
- After viewing, the secret is **permanently deleted**
- All access attempts are **logged into a database**

Tech Stack:

- **Frontend**: React.js
- **Backend**: Node.js (Express.js)
- **Database**: MySQL **or** MongoDB (choose one)
- **Auth**: JWT (recommended) or sessions
- **Password Hashing**: bcrypt

Functional Requirements:

User System

- **POST /api/signup**
    - Register a new user with an email and a password
    - Store the hashed password

- **POST /api/login**
    - Return JWT token on success

- **GET /api/me**
    - Return current user (auth-protected)

Secret Creation

- **POST /api/secret**
    - Auth required
    - Input: { message, password }
    - Stores:
        - Token (unique)
        - Hashed message password
        - Message text
        - Creator user ID
        - Created timestamp
    - Returns shareable URL: /secret/:token

Secret Viewing

- **GET /api/secret/:token**
    - Returns: If token exists → ask for password; if not → "Secret unavailable"

- **POST /api/secret/:token/view**
    - Input: { password }
    - If correct:
        - Return the message
        - Delete the secret from the DB
        - Log the access (see below)
    - If incorrect:
        - Return error
        - Still log the access

Frontend Features (React)

1. **Login/Register Pages**

2. **Dashboard**
    - Form to create a secret: message + password
    - Shows the generated link

3. **Secret Viewer Page**
    - Input field to enter the password
    - Show secret if valid
    - Show error if not
    - Always delete after success

Security Considerations:

- JWT token required for all secret creation routes
- Passwords (user & secret) must be hashed (bcrypt)
- Rate-limit secret view requests to avoid brute-force attacks

Bonus Features (Optional):

- Auto-expire secret after 30 minutes
- "View my sent secrets" dashboard (with status: viewed/not viewed)
- Admin route: GET /api/access-logs (auth-protected) to see logs

Deliverables:

- GitHub repo
- Working app with test users
- Include setup instructions for DB + .env