CU CHANDIGARH UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

PROJECT REPORT

- **Project Name:**

  Note Sharing Website – A Full Stack Java Application using Spring Boot and MySQL

- **Reference Websites:**

  1. https://evernote.com/
  2. https://www.onenote.com/

- **Project Description:**

  The Note Sharing Website is a full-stack web application developed using Java, Spring Boot, React, and MS SQL Server. The system allows users to securely register, log in, and **contribute notes to a public community feed**. Users can create, view, edit, and delete their own notes, while also browsing and learning from notes shared by all other users.

  The project follows a client-server model. The frontend, built with React, provides a responsive and interactive single-page application (SPA). It handles user authentication and provides a two-column layout for creating new notes and viewing all **publicly shared** notes in a scrollable card-based format.

  The backend, built with Spring Boot, serves a RESTful API to handle request processing, user authentication (using JWT), database connectivity, and business logic. Data is stored in an MS SQL Server database, managed via Spring Data JPA. When a note is created, the author's name is automatically associated with it, allowing for proper attribution in the community feed.

  The application demonstrates end-to-end integration of a modern frontend (React) with a robust backend (Spring Boot), highlighting core web development concepts such as RESTful APIs, token-based security, and database CRUD operations in a **community-focused** application.

- **Problem Statement:**

  In today's digital environment, valuable knowledge and learning resources are often siloed. Students, professionals, and hobbyists create useful notes, guides, and summaries, but these are typically stored privately. This makes it difficult for a community to discover and benefit from this shared knowledge.

  There is a need for a centralized, community-driven platform where users can not only create and manage their notes but also easily share them with the public. This system must allow users to contribute their insights for others to learn from, fostering a collaborative educational environment, while also allowing them to manage their own contributions.

  The Note Sharing Website addresses this problem by offering a secure and user-friendly platform for both creating and publicly sharing notes. It uses Spring Boot and SQL Server to securely store user accounts and their contributions. The React frontend provides a simple interface for users to create notes (which are visible to everyone) and for the community to browse and learn from the "All Notes" feed, which displays contributions from all users.

- **High-Level Design:**
  1. **Frontend (React):**
     - User interface for Signup (`Signup.js`), Login (`Login.js`), and Note Management (`Notes.js`).
     - Fetches and sends data to backend REST APIs using the `fetch` API.
     - Stores the JWT token in `localStorage` after login and attaches it to the `Authorization` header for all secure requests.
     - Displays notes in a two-column, scrollable card layout with a dark, modern theme.
  2. **Backend (Spring Boot – Java):**
     - Handles REST API requests via `AuthController` (for `/auth/signup`, `/auth/login`) and `NoteController` (for `/api/notes`).
     - Implements user registration (with password hashing) and login logic using `UserService`.
     - Manages JWT token generation (`JwtUtil.java`) and validation (`JwtAuthFilter.java`, `SecurityConfig.java`).
     - Performs CRUD operations for notes, automatically attaching the logged-in user's name (`authorName`) upon creation.
  3. **Database (MS SQL Server via Spring Data JPA):**
     - Stores user information (ID, name, email, hashed password) in the `Users` table (mapped from `User.java`).
     - Stores note information (ID, title, subject, content, authorName, createdAt) in the `Note` table (mapped from `Note.java`).
     - Repositories (`UserRepository`, `NoteRepository`) handle all database queries.
  4. **Workflow:**
     - User signs up with their name, email, and password. The password is hashed and stored.
     - User logs in. The backend validates credentials and returns a JWT token.
     - The React frontend stores this token and navigates to the main `Notes` page.
     - When the `Notes` page loads, it sends a `GET` request to `/api/notes` (with the token) to fetch all public notes.
     - User creates a new note using the left-hand form. A `POST` request is sent to `/api/notes`. The backend verifies the token, retrieves the user's name from the database, saves it to the `note.authorName` field, and saves the note.
     - The "All Notes" section refreshes, displaying the new note (visible to all users) with the author's name.

- **Future Scope:**
  1. Private Notes: Implement functionality for a user to create private notes, visible only to them, in addition to the public community notes.
  2. Rich Text Editor: Replace the standard <textarea> with a rich text editor (e.g., ReactQuill) to allow for formatting (bold, italics, lists).
  3. Search and Filtering: Add a search bar to filter notes in the "All Notes" section by title, subject, content, or author.
  4. Tagging/Categories: Allow users to organize notes by adding tags or creating different "notebooks" or categories.
  5. Cloud Deployment: Deploy the full-stack application to a cloud platform like AWS or Azure for public accessibility and scalability.