Project Title: LastPass Clone (Give your own title)

Project Description:

The idea is to build a password management tool. Read more about LastPass at https://www.lastpass.com

The app should be able to store user's email and password (mapping them together). It should give it back to the user whenever required/requested. A search functionality should be provided to search by email or title of the stored password. Further, users should be able to share the password with others where the person on the receiving side can only copy the password but cannot view it.

Note: Do not try to clone everything from LastPass. You are expected to use it as a reference to understand the way a password management tool works. Instructions and other information follow:

Basic Features of the App:

1. **User Login:** User should be able to sign up using email. The user should also be able to login using the same ID. Forgot password, change password and change email address options must be provided. Note that is the master password to login to your app.

2. **Vault:** A vault is a place where the user can view all stored passwords. This is where they can search for stored passwords.

3. **Secure Notes:** Here, the user can store their private notes. This can be almost anything like their PAN card number, credit card information, Wi-Fi passwords or private phone numbers.

4. **Shared Passwords:** User should be able to share passwords as mentioned below. This section of the app should display all the passwords he has shared with other and others who have shared with him. The sharing feature should be limited to 3 per user for Free Tier users and unlimited for Premium Users.

Note: Other trivial features that build a meaningful web app have to be assumed and implemented.

Key Features of the App:

1. **Storing:** User will enter their website, email and password in your app. Your app should automatically recognize the website from its URL and should generate a **title** for it. The title and email are stored as they are but should be sorted by title. The sites have to be categorized into their respective categories (user can choose the type of category). Example: Social, Video, News etc., The password, secure notes and private numbers has to be encrypted using a secure hashing algorithm (SHA) and should be stored. If the user doesn't want to enter a password, a password **should be generated** and stored for them. There should be an option to edit and delete the stored password. Every time an edit or a delete operation is requested, the user should be prompted to enter their password again.

   Learn more about SHA and storing of passwords below:
   - https://en.wikipedia.org/wiki/SHA-1
   - https://www.geeksforgeeks.org/store-password-database/
   - https://stackoverflow.com/questions/1054022/best-way-to-store-password-in-database
   - https://bitwiseshiftleft.github.io/sjcl/

2. **Searching and Retrieving:** A search bar should be able to search stored passwords by title or email and should return back the result instantly. The search algorithm should use a binary search algorithm to perform the same (remember that the titles have to be stored in a sorted order). The password has to be hidden and the user should be able to view their password by clicking on a 'view' button. The notes and other private details can be viewed directly.

3. **Sharing**: Every user should be able to share their passwords, secure notes and other information with every other user by entering their email ID. The sending user will send a ‘share’ request and the receiving user will receive a notification saying that a password is being shared with them. The sender should have an option before sending which should permit the receiver to view the password or not. The receiving user has to accept the invitation where the title, email and password will be shared with them. Once shared, the password (only its title) should appear in both sender and receiver side under their shared passwords section. The receiving party can only copy the password but cannot view it (if they have permission to view). There should be a time limit (expiring) to the sharing feature which means that receiver has access to the password for only a certain amount of time.

   Note: If the sending user changes or updates his password from his account, it **should not** be reflected among shared users. The sender has to approve again in order for the receiver to view the password again.

4. **Free/Premium Users:** There can be two types of users. A free user will have all functionality except that they cannot share a single password with more than three users. A free user also cannot share private notes and other information to more than one user. Premium users can share unlimited passwords and notes.

Exceptions and Errors to be handled:

1. Can a shared password be shared with another user? Will implementing this feature an advantage or a disadvantage?
2. If the user uses the same password for multiple websites, they should be promoted with security message saying that it is not secure.
3. What happens to the passwords (in a hypothetical scenario) if a user **dies**?

Note: More points to handling out more exceptions and to making sure that the app performs well in all conditions.

Evaluation Criteria:

- **Code Quality:** Every line of code has to be indented properly using best practices. Add comments wherever necessary. You will be judged on overall code quality, code structuring and modularity. Organized, scalable code architecture will be appreciated. Any extra effort made to look the code more readable will get you extra points.

- **Database Design:** The app's core database architecture and the way attributes, keys are mapped are judged. The database is expected to be designed with scalability in mind.

- **User Interface:** The app should be made keeping in mind it will be used by a layman and not a developer. It should be as intuitive as possible. The app should be as easy to understand as possible. You should think about innovative ways of delivering messages (like errors, failures) to users.

- **Originality:** The code will be checked for plagiarism, it is considered as a minus point if the code is not original, copied/borrowed from a fellow candidate or the internet.

Note: If in doubt, use your best judgement. Also add assumptions, if any, as comments in the code, wherever necessary.

Submission Guidelines

- **Source Code:** The root repository of the project has to be kept track in **git** from the beginning. The project should be uploaded to **GitHub** and a possible link has to be shared of the repository.

- **Documentation:** Include a README in the repository with steps to install/run the application and basic description of design/architecture and approach. More information on how to write a good documentation can be found here: https://github.com/dctacademy/docs-format

- **Deployment:** Extra points for deploying the application on the web. Using Heroku is recommended. If needed, we will provide you help regarding this. A link should be submitted of the deployed application. The app should be in working condition. Please do not submit an incomplete or broken app.