CPT-200 Team 2 Server/Backend notes

Server will MySQL. Will use MySQL workbench.

Server will be kept in GitHub repo for now, unless someone has a better idea for hosting??

APIs used on backend possibly for key storage:

Google Cloud Secret Manager

Other API

Google Drive for note export

AI APIs (Google?) for either text suggestion or note organization

Preliminary Tables

USERS (stores user account details)

- **Primary Key User\_**ID

- username (would be username or Google API login ID)

- hash of password

NOTES (stores encrypted notes)

- Note\_ID (PK)

- User\_ID (FK)

- title

- content

- create\_date

- update\_date

\*OPTIONAL\* SESSION (could be used to timeout a user if they were logged in too long)

- ID (PK)

- User\_ID

- token

- created\_time

- expiration\_time

If didn’t use API for key storage, would need an EncryptionKeys table, would relate user\_id to the key for that user.

I need someone to draw up the data flow diagram in the presentation. It is fairly simple.

Front End (Django web interface) talks to the Back End (also Django). The Back end communicates with the key management API and the MySQL database. The back end also contains the encryption components with will use Python libraries to complete this.

Some of the database queries needed would include and example commands would be:

(using an example user id of 1 and note ID of 1)

User related,

User registration (preliminary)

INSERT INTO USERS (username, pw) VALUES (‘test\_user’, ‘pw’);

User Login/Authentication (preliminary)

SELECT \* FROM USERS WHERE USERNAME = ‘user’ AND pw = ‘pw’;

User deletion

DELETE FROM NOTES WHERE USER\_ID = 1;

DELETE FROM USERS WHERE USER\_ID = 1;

Note related,

Which notes belong to a user?

SELECT \* FROM NOTES WHERE USER\_ID = 1;

User searches for notes

SELECT \* FROM NOTES WHERE USER\_ID = 1 AND content

Note creation

Note retrieval

Note update

UPDATE NOTES SET TITLE = ‘NEW’, CONTENT = ‘UPDATE’ WHERE ID=1 AND USER\_ID=1;

Note deletion

DELETE FROM NOTES WHERE ID = 1;

Session related \*OPTIONAL\*

Session creation

Session validation

Session expiration

Encryption Key Management if stored in database

Store key

Retrieve User key

Delete Key (if user deleted)