```sql
DROP TABLE IF EXISTS users;

CREATE TABLE users (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    username TEXT UNIQUE NOT NULL,
    first_name TEXT NOT NULL,
    last_name TEXT NOT NULL,
    email_address TEXT UNIQUE NOT NULL,
    user_group TEXT,
    password TEXT NOT NULL,
    salt TEXT NOT NULL
);
```

Database for user microservice including account details, group, etc.

- id: unique identifier for each user
- first_name, last_name: users' information
- username: used for login and unique for users
- email_address: information (unique)
- group: the user's groups
- password: store hashed user's password for security.
- salt: help improve security for hashing password

APIs:

POST /documents:

Description: Checking JWT and return data.

Parameters: {

"jwt_token": <jwt token>

}

Return: {

"user_id": <user's id>,

"username": <username>,

"groups": <groups>

}

POST/get_user:

Description: Retrieves user's data from user's id.

Parameters: user_id: <user's id>

Response:{

"user_id": <user's id>,

"groups": <group>,

"username": <username>

}

```
DROP TABLE IF EXISTS documents;

CREATE TABLE documents (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    filename TEXT UNIQUE NOT NULL,
    body TEXT NOT NULL,
    owner_id INTEGER NOT NULL,
    groups TEXT NOT NULL,
    FOREIGN KEY (owner_id) REFERENCES users (id)
);
```

Database for documents microservice.

- id: unique identifier for each document
- filename: the document's name; it must be unique.
- body: the content of the document.
- owner_id: links the document to its creator (user ID from user.db).
- groups: JSON string specifying groups allowed to access/edit the document.

APIs:

POST/search:

Description: User for searching document data from filename (body, filename, owner_id, groups)

Parameters: { 'filename': <filename>}

Response: {

'filename': <filename>,

'body': <body>,

'owner_id': <owner's id>

'groups': <JSON groups>

}

```sql
DROP TABLE IF EXISTS logs;

CREATE TABLE logs (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    event_type TEXT NOT NULL,
    username TEXT NOT NULL,
    filename TEXT DEFAULT 'NULL'
);
```

This database tracks activity logs across all microservices.

- id: a unique identifier for each log entry (use for activity orders instead of timestamps).
- event_type: the type of action logged (e.g., user_creation, login, document_creation, document_edit, document_search).
- username: the user who performed the action.
- filename: the name of the document involved for file related activities and default is "NULL" for data type related to user.

APIs:

GET/get:

Description: Retrieves user's total and last modifications from filename

Parameters: 'filename': <filename>

Response:{

"total_mod": <total modifications>,

"last_mod": <username's last modification>

}