Web API Specification

Server URL: https://cloudlockr.herokuapp.com/

Data storage and authentication will be facilitated by a web server. These operations will allow for the phone app as well as DE1 to function. The required APIs are as follows below.

Required APIs for DE1:

1. POST /file/{fileId}/{blobNumber}
    - Stores a new blob of data associated the the fileId for the given blobNumber
    - Request
        - Body: fileData (the binary blob data to store)
    - Returns
        - Error if fileId is not UUID
        - Error if fileId does not correspond to existing file
        - Error if blob number if greater than number of total blobs
        - Otherwise {message: “Stored blob”}

2. GET /file/{fileId}
    - Retrieves the base metadata for file
    - Request
    - Returns
        - Error if fileId is not UUID
        - Error if fileId does not correspond to existing file
        - Otherwise {numBlobs: number of blobs}

3. GET /file/{fileId}/{blobNumber}
    - Retrieves a given blob of data associated the the fileId and blobNumber
    - Request
    - Returns
        - Error if fileId is not UUID
        - Error if fileId does not correspond to existing file
        - Error if blob number if greater than or equal to number of total blobs
        - The fileData associated with file and blob number

Required APIs for App:

4. POST /user/login
    - Authenticates a user to log them in to the app
    - Request
        - Header
            - email
            - password
    - Returns
        - If error: { errors: [error0, error1, ...] }
        - If success:

```
userId?: string;
accessToken?: string;
refreshToken?: string;
token_type?: string;
expires?: number;
message?: string;
```

            - accessToken for authentication
                - Pass in authorization header “token_type accessToken”
                - Expires in 15 minutes
            - refreshToken for acquiring new access token
                - Expires in 1 day

5. POST /user/register
    - Registers a new user
    - Request
        - Header
            - email
            - password
            - password1 (both passwords must match)
    - Returns
        - Same as login

6. POST /user/logout
    - Logout the user
    - Request
        - Header
            - authorization: “token_type accessToken”
            - refreshToken
    - Returns
        - If error: { errors: [error0, …] }
        - If success: { message: “Logged out” }

7. POST /user/refresh
    - Grants new access token
    - userId must be equal to the id of user that the refreshToken is granted to
        - Shouldn't be a problem normally
        - Would only throw error if attacker gained access to refreshToken and tried to get access token without the correct userId
    - Request
        - Header
            - userId
            - refreshToken
    - Returns
        - If error: { errors: [error0, …] }
        - If success: { accessToken, refreshToken, token_type, message: "Refreshed" }

8. DELETE /user
    - Deletes user account and files
    - Request
        - Header
            - authorization: "token_type accessToken"
            - refreshToken
    - Returns
        - If not authenticated: {ok: false, errors: [error0, …]}
        - If authenticated: { ok: true, message: "Account deleted" }

9. GET /user/files
    - Retrieves a list of all of a user's stored files (the metadata)
    - Request
        - Header
            - authorization: "token_type accessToken"
    - Returns
        - If not authenticated: { errors: [error0, …] }
        - Otherwise, array of fileMetadata

10. POST /file
    - Creates the base metadata for a new file
    - Request
        - Header:
            - authorization: "token_type accessToken"
        - Body:
            - fileName (the name to associate to the file)
            - fileType (string)
    - Returns
        - The newly created {fileId, fileName, fileType}

        - Error if fileName or fileType is empty

11. DELETE /file/{fileId}
    - Deletes all data associated to a file from the database
    - Request
        - Header
            - authorization: “token_type accessToken”
    - Returns
        - Error if fileId is not a UUID
        - Otherwise, {message: “Deleted file”}