TEAM: ChefBatMan
Group Members:

1. Chenyang Fang(cf2843)
2. Wan Xu(wx2227)
3. Guancheng Ren(gr2625)
4. Tian Niu(tn2415)

Part 0: DATE we met with IA: 10/30/2020

Part 1:

1.What does our project do?

Ans: The goal of our project is to create a website where students of Columbia are able to share their study notes with each other, comment on other study notes and download shared notes. Our project will enable our users to upvote/downvote any study notes or save those as users' favorites or even comment/highlight study notes. Also, users will have credits as rewards by sharing notes .

2. Who will be its users?

Ans: the target users would be the students or even professors of Columbia University. Our current authentication strategy is to allow whoever owns a LionMail affiliated with Columbia University to register/login our website.

3.Demo?

Ans: our demo will be able to show:

1. A user can create an account (only with LionMail ending with @columbia.edu)
2. A registered user can login with email and password
3. A logged-in user can search for notes by specifying department name and course number
4. A user can view previously-uploaded study notes specified by the searching parameters.
5. A user can upload his/her study notes by specifying department name and course number
6. A user can delete the notes uploaded by himself

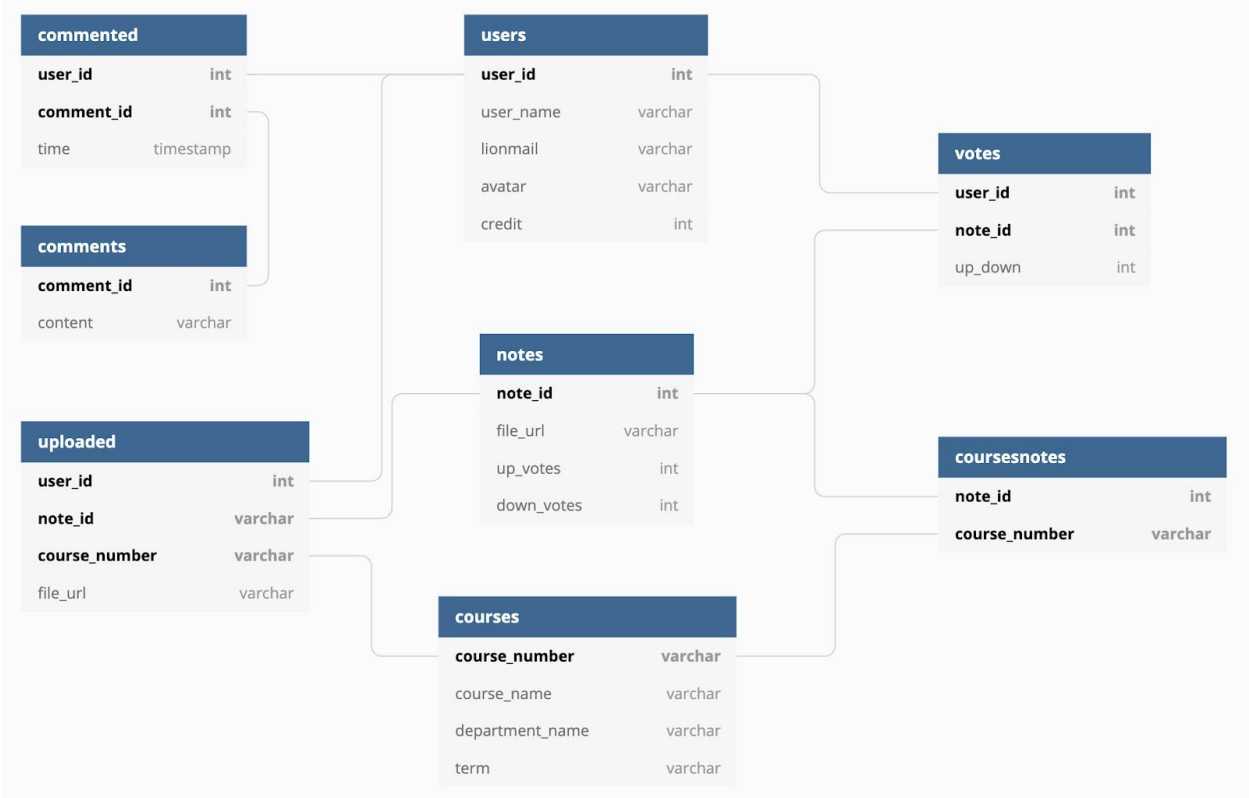
4. What kind of data do you plan to store?

Ans:

1. User information:
    a. username

    b. related lionmail.
    c. the credits the user currently has.
    d. The path to the notes saved to favorite by the user
    e. The path to users' avatar
    f. A list of external links that stores the notes uploaded by the user
2. Course material: lecturer notes and their metadata.
3. Courses info: the department name, course number, course name, and the term in which the course is offered

In detail, we plan to use 8 tables to store the necessary information, the table forms many-to-many relationships and we join the tables when we want some combined information.



The note will be uploaded as a pdf format and stored at amazon S3, some metadata we plan to store are:

- Unique File ID
- File size
- Uploader
- Creation Date
- URL to S3

5. APIs and what for?

1. We use Django as our RESTful service to handle different requests from the users. We use REACT for front-end development.

2. The login part we plan to use Google sign in API for allowing users to do fast logins and fast registrations. And we will also verify that the user logged in with Google has a Columbia email address

3. We use Dicebear Avatars to automatically generate avatars for users.

4. Front-end will adopt React Router to navigate components for our application.

Suggestion from IA: react-pdf-highlighter external api to make the study notes able to be commented/highlighted by the users. However, after discussion, we think our current design has included a great number of functionalities and maybe in later phase we will add this to our plan.

Part 2:

1. As a student of a particular course, I want to view notes uploaded by the professor or other students in this class, so that I can better understand course materials and prepare for homeworks and exams.
    a. Common cases:
        i. A logged-in user can search a class by department name and course number.
        ii. The user can click a particular file that was previously uploaded to view.
    b. Special cases:
        i. A logged-in user enters a department name and course number combo that does not exist. The system pops up an error message.
2. As a student/professor of a particular course, I want to upload my lecture notes to share with other students in this class, so that I can help others to study.
    a. Common cases:
        i. A logged-in user can upload his class notes in pdf and txt file type by indicating which department and course number the file belongs to.
        ii. The uploaded file is then available to all users to view and download.
        iii. The user gets 5 credits after uploading successfully.
    b. Special cases:
        i. A logged-in user enters a department name and course number combo that does not exist. The system pops up an error message.
        ii. A logged-in user enters a valid combo but tries to upload a file that has invalid file type. The system pops up an error message.
3. As a student of a particular course, I want to download notes uploaded by the professor or other students in this class, so that I can view the notes offline.
    a. Common cases:

        i. A logged-in user can search a class by department name and course number.
        ii. The user can click a particular file that was previously uploaded, then choose to download.
    b. Special cases:
        i. A logged-in user enters a department name and course number combo that does not exist. The system pops up an error message.
4. As a student of a particular course, I want to save some files to my “favorites”, so that I can easily retrieve it without searching again and again.
    a. Common cases:
        i. A logged-in user can search a class by department name and course number.
        ii. The user can choose a particular file to save to his “favorite list”.
        iii. This file is then added to the user’s “favorites” list and the user can view the file from the list.
        iv. The files in the “favorites” list can be unsaved. It is then removed from the user’s “favorite” list.
    b. Special cases:
        i. A logged-in user enters a department name and course number combo that does not exist. The system pops up an error message.
        ii. A logged-in user tries to save a file that was already saved before. The system pops up a message telling the user the file was already saved to “favorites”.
        iii. A logged-in user tries to unsave a file that was not saved before. The system pops up a message telling the user the file was never saved before.
5. As a student of a particular course, I want to upvote or downvote a note so that I have a way to tell others students that the file is of high or low quality.
    a. Common cases:
        i. A logged-in user can search a class by department name and course number.
        ii. The user can upvote or downvote this file.
        iii. The upvote and downvote number increments by 1 accordingly.
        iv. The user can undo his action by clicking the button again and the upvote/downvote number decrements by 1 accordingly.
    b. Special cases:
        i. A logged-in user enters a department name and course number combo that does not exist. The system pops up an error message.
        ii. A logged-in user tries to upvote (or downvote) a file that has been downvoted (or upvoted) by himself. The system pops up a message telling the user he has already voted for this file.
6. As a student/professor of a particular course, I want to delete some files uploaded by myself if some mistakes are present in the notes, so that the wrong notes won’t mislead other students.

    a. Common cases:
        i. A logged-in user can search a class by department name and course number.
        ii. The user can choose the particular file that he wants to delete and then delete the file.
        iii. The deleted file is then removed from the system.
    b. Special cases:
        i. A logged-in user enters a department name and course number combo that does not exist. The system pops up an error message.
7. As a student of a particular course, I want to post comments and read comments posted by other students to a particular file belonging to that class, so that I can ask questions, discuss my thoughts, and answer questions related to the course materials.
    a. Common cases:
        i. A logged-in user can search a class by department name and course number.
        ii. The user can choose a file to view explicitly.
        iii. The user can read comments in the discussion forum of that file.
        iv. The user can post his new comments to the forum.
        v. The post is then available to view for all users.
    b. Special cases:
        i. A logged-in user enters a department name and course number combo that does not exist. The system pops up an error message.
        ii. The user enters a blank comment and tries to post it. The system detects that the post is blank and pops up an error message.

Part 3:

Overall we will conduct the acceptance test in a blackbox testing environment, from the users perspective. Each acceptance test correspond to use cases

1. **View Course Notes**

   Common Case #1:
    a. The user opens the airNote website and login through Google
    b. User searches a course number with a corresponding department name
    c. The course is found and course notes are displayed
    d. User clicks into a note and view it

   Special Case #1 (login through email not from Columbia University):
    a. User enters the homepage of airNote
    b. Login through Google with email not from Columbia University
       Sample input: superman@gmail.com
    c. The system reject the login request

   Special Case #2 (invalid login credential):
    a. (Suppose the user already has an account)

b. User enters airNote and login through Google with wrong password
c. The system reject the request and ask the user to try again

Special Case #3 (course not exist):

a. The user opens the airNote website and login
b. The user searches a course number corresponding to a department name
c. The system notifies the user that the specified course does not exist.

2. **Upload Course Notes**

Common Case #1 (user already have an account):

a. The user opens the airNote website and login
b. The user clicks upload note button and is redirected to file upload page
c. The user upload the note and fills in necessary information about the course
   Sample input: a pdf file / note in plain text
d. The user clicks submit button and the note is available to all users

Special Case #1 (login through email not from Columbia University):

a. User enters the homepage of airNote
b. Login through Google with email not from Columbia University
   Sample input: superman@gmail.com
c. The system reject the login request

Special Case #2 (user didn't fill in all the required information):

a. The user opens the airNote website and login
b. The user clicks upload note button and is redirected to file upload page
c. The user upload the note but fails to fill in some information
   Sample: forget to fill in title, label, course number of the note
d. The system prompt the user to fill in those fields

Special Case #3 (invalid file type):

a. The user opens the airNote website and login
b. The user clicks upload note button and is redirected to file upload page
c. The user uploads the file but the file type is not expected.
   Sample input: exe file, bash file

3. **Download Notes**

Common Case #1:

a. The user opens the airNote website and login through Google
b. User searches a course number with a corresponding department name
c. The course is found and course notes are displayed
d. User clicks into a note
e. A download button is visible on the page and the use clicks it
f. A download window pops up and the user saves the file on his/her own computer

Special Case #1 (login through email not from Columbia University):

a. User enters the homepage of airNote
b. Login through Google with email not from Columbia University
   Sample input: superman@gmail.com
c. The system reject the login request

Special Case #2 (invalid login credential):

a. (Suppose the user already has an account)
b. User enters airNote and login with wrong password
c. The system reject the request and ask the user to try again

Special Case #3 (course does not exist):

a. The user opens the airNote website and login
b. The user searches a course number corresponding to a department name
c. The system notifies the user that the specified course does not exist.

4. **Add Favorite**

Common Case #1 (adding to favorite):

a. The user opens the airNote website and login through Google
b. User searches a course number with a corresponding department name
c. The course is found and course notes are displayed
d. User clicks into a note and view it
e. An “add to favorite” button is visible on the page and user clicks on the button
f. The note is added to user’s favorite list

Common Case #2 (viewing favorite list):

a. The user opens the airNote website and login through Google
b. User clicks on “my favorites” button on the navigation bar
c. A list of user’s previously added favorite notes are displayed

Special Case #1 (items in favorite list is deleted):

a. The other user deletes a note that is in the user’s favorites list
b. The user clicks on “my favorites” button on the navigation bar
c. The corresponding note is also removed from the favorites list

5. **Upvote and Downvote**

Common Case #1 (Upvote):

a. User enters the homepage of airNote with credentials
   Sample credential: cookie, session storage
b. User searches a note and clicks into the preview page
c. User clicks the upvote button
d. The system updates the upvotes by incrementing it by 1

Common Case #2 (downvote):

a. User enters the homepage of airNote with credentials
   Sample credential: cookie, session storage
b. User searches a note and clicks into the preview page
c. User clicks the downvote button
d. The system updates the downvotes by decrementing it by 1

Special Case #1 (the file has already been upvoted/downvoted by user):

a. User enters the homepage of airNote with credentials of non file owner
   Sample credential: cookie, session storage
b. User searches a note that has been upvoted by him/her and clicks into the preview page

c. The user cannot upvote this note, the upvote button will cancel his/her last action of upvoting instead of to upvote again

Special Case #2 (course not exist):

a. The user login into the airNotes with credentials
b. The user searches for specific course
c. The system pops up messages suggesting that course does not exist

6. DELETE

Common Case #1:

a. User enters the homepage of airNote with credentials
   Sample credential: cookie, session storage
b. User searches a note he/she uploaded before and clicks into the preview page
c. User clicks the delete button
d. The system pops up a window with confirmation message
e. The user clicks YES
f. The entry is deleted and other users can't see it

Common Case #2 (user decides not to delete the file):

a. User enters the homepage of airNote with credentials
   Sample credential: cookie, session storage
b. User searches a note he/she uploaded before and clicks into the preview page
c. User clicks the delete button
d. The system pops up a window with confirmation message
e. The user clicks NO
f. The entry is not deleted and the user is redirected to the preview page

Special Case #1 (the file was not uploaded by the user):

a. User enters the homepage of airNote with credentials of non file owner
   Sample credential: cookie, session storage.
b. User searches a note he/she didn't upload and clicks into the preview page
c. There is no delete button

7. Comment

Common Case #1(comment):

a. User enters the homepage of airNote with credentials
   Sample credential: cookie, session storage
b. User searches a note and clicks into the preview page
c. User can view the document and the comments in the discussion forum
d. User clicks 'create comment' and a window pops up so that user can write comment for this note
e. The user clicks upload and the comment is available in the discussion forum

Common Case #2 (user decides to delete comment):

e. User enters the homepage of airNote with credentials
   Sample credential: cookie, session storage
f. User searches a note he/she commented before and clicks into the preview page

g. The user clicks delete button under the comment he/she made
h. The comment is deleted by the system and no longer shown in the forum

Special Case #1 (the user wants to delete others' comments):

d. User enters the homepage of airNote with credentials of non file owner
   Sample credential: cookie, session storage
e. User searches a note he/she did not comment and clicks into the preview page
f. There is no delete button for he/she in the discussion forum

Special Case #2 (user enters empty comment)

a. User enters the homepage of airNote with credentials of non file owner
   Sample credential: cookie, session storage
b. The user searches for some note and clicks into the preview page
c. The user clicks 'create comment' and a window pops up so that user can write comment for this note
d. The user enters nothing and clicks upload
e. The system pops up error message suggesting that comment cannot be empty

Part 4:

IDE: PyCharm
Front End: React HTML JavaScript
Back End: Django
Package Manager: pip/conda
Unit testing tool: unittest within PyCharm integrated tools, jest as unit-test for REACT
Style Checker: pep8 1.7.1/ pylint 2.6.0
Bug finder: pylint 2.6.0
Code Coverage: coverage 5.3
Persistent data store: DynamoDB/ MongoDB