Group assignment - Advanced JavaScript with React

The deadline for this assignment is **Thursday, May 16, 23:59**.

For this mandatory exercise you should work on **master** branch only.

Preparation

1. Create a new repository on GitHub called **mandatory-advanced-js5**.
2. Follow the instructions that GitHub gives you; Create a local repository and add a remote or clone the newly created repository.

Submission

When you submit the exercise in PingPong, before the deadline, you will enter a link to your repository, such as:

https://github.com/mygithubusername/mandatory-advanced-js5

You also need to supply a link to the deployed application.

The teacher will look in the master branch. If any commits are done to the branch after the deadline, the teacher will look at the last commit before the deadline.

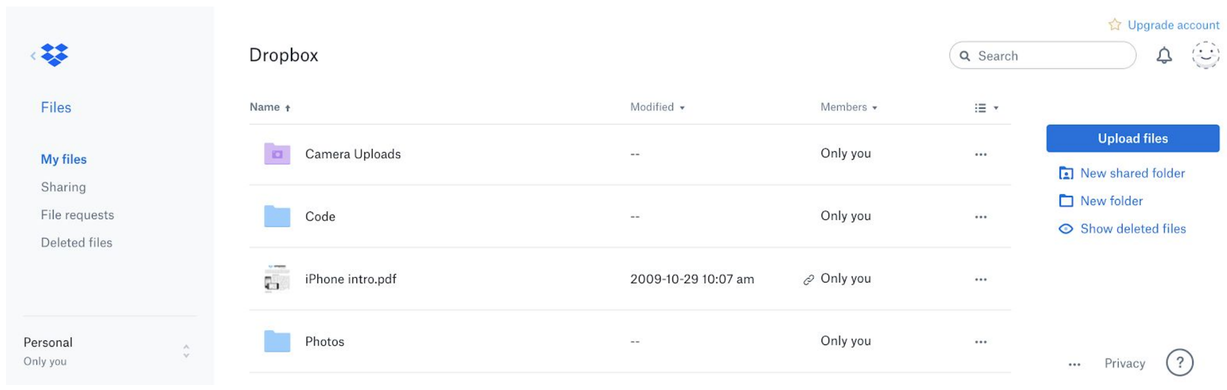
This is a group assignment but it will be presented individually. Each group member must have full knowledge of the application and be ready to answer any questions from the teacher. You will get one of the grades **G**, **VG** or **IG**. The grade is based on both the implementation and the oral presentation.

Instructions

In this project the class will be divided into groups of 3 or 4 members. The aim of this project is to create a web application that uses the Dropbox API. Dropbox is a very popular cloud-based file storage service.

You will create a program that allows users to sign in and view/manage their files, similar to the functionality in the actual Dropbox web service.

If you haven’t used Dropbox already, I recommend you start by creating an account and playing around in the service to get more accustomed to it.
https://www.dropbox.com/



Name	Modified	Members
Camera Uploads	--	Only you
Code	--	Only you
iPhone intro.pdf	2009-10-29 10:07 am	Only you
Photos	--	Only you

Above is a screenshot of a user’s root folder.

The interface contains the following:

- A "main" area showing the files and folders (items) available in the current folder.
- The path of the current folder at the top (Dropbox).
- Metadata about each item in the current folder, including:
  - Whether the item is a file or folder (represented by the corresponding icon).
  - The name of the item.
  - The date and time of the most recent modification to the item.
- A search bar allowing the user to search for items across his or her account
- An option to upload a file to the current folder.

When clicking on a folder in the main area, the interface navigates to and shows the items in that folder and also updates the path of the current folder displayed at the top.

Note: If the user has navigated into a subfolder, the path contains clickable path parts that allow the user to navigate up in the hierarchy. As an example, if the user clicks on the "Photos" folder in the image above, the path will update to "Dropbox / Photos", and the "Dropbox" part will be made clickable to allow navigation back to the root folder

When clicking on a file in the main area, the interface shows a preview (if possible) of the file, with the option to download the file.

As preparation for this project, continue exploring the Dropbox interface and gather inspiration for your project's UI, by using your own Dropbox account!

Requirements

In order to complete the project a number of requirements must be implemented.

For the grade **G** the specification is as follows:

- In order for your web application to access the contents of a user's account, the user must log in, i.e. authenticate against Dropbox and authorize your application to perform actions on his or her behalf.

  Dropbox accomplishes this via the Oauth protocol; the process is described in detail here. https://www.dropbox.com/developers/reference/oauth-guide
- The result of the Oauth flow is an access token which is used in executing calls to the Dropbox API. This access token should be stored (in the client) and reused until the user logs out.
- The application UI must contain
  - A "main" area showing the files and folders in the current folder
  - The complete path of the current folder displayed on the top. If the user is currently viewing a sub-folder, each part in the path should be made clickable to enable navigation to all folders higher in the hierarchy.
  - If a file is an image, a thumbnail of the image should be displayed instead of the regular file icon
  - Metadata for a file should be displayed and must include: filename, size (in a human-readable format), last modified timestamp
- When a user clicks on a file, a download is started. No preview functionality is required.
- A user must be able to upload a file to the current folder
- A user must be able to create a new folder in the current folder
- A user must be able to remove files and folders. Before an item is removed the user should be asked if he/she really wants to remove the item.
- A user must be able to "star" files and folders. The UI must allow the user to view all the items that have been starred.

  This functionality should be implemented on the client. You can use localStorage to remember the starred items.

  Note: If a starred item is removed or moved outside och the application, the file will no longer be accessible in the application. This will cause an error that must be handled.

  Remember to remove the starred items for localStorage when the user signs out.
- Finally, a "go to parent folder" button must be provided so the user can easily navigate to the parent folder

For the grade VG there are some additional requirements:

- A user must be able to search for files and folders by name. Search results should be displayed in the “main” area.

  Clicking on a folder in the search results navigates to that folder. Clicking on a file in the search results starts a download.
- A user must be able to copy files and folders. You can either chose to create the copy in the same folder with a new name, or show a dialog where the user can select a target folder.
- A user must be able to rename files and folders.
- A user must be able to move files and folders. The application should show a dialog where the user selects a target folder.
- When a change happens outside the application the content should automatically be updated. You can implement this either by polling the API or by using webhooks (more on that later)
- You must write tests for at least one React component

Implementation

The application should be implemented as a SPA using React. You need to implement correct routing.

To communicate with the Dropbox API you should use the official Javascript SDK. You can find documentation here:
http://dropbox.github.io/dropbox-sdk-js/

A reference of all the available methods is available here:
https://dropbox.github.io/dropbox-sdk-js/Dropbox.html

To implement the UI you are free to use a CSS framework or write CSS for the components yourself.

Webhooks

For the grade VG the application must automatically update the content if something is changed outside the application. This can happen if you upload or remove a file in another client.
A simple solution is to use polling (make a request every n seconds).

Another (better) way to solve this is to use webhooks. You can configure the Dropbox API to make a request to a URL every time users’ files change on Dropbox. This requires you to create a simple backend that handles the requests from the Dropbox API.

You can read more here:
https://www.dropbox.com/developers/reference/webhooks

There are many ways to handle the webhooks. One way would be to create a server with Socket.io.

Deployment

The application must be deployed so it can be accessed from a publicly accessible URL. You can use GitHub Pages to deploy the application.

Tips

- Before doing any coding, decide on the layout of the application and the views and components that need to be implemented
- Try to split the implementation into independent parts so the members in your group can work on different parts of the application simultaneously