Appsus

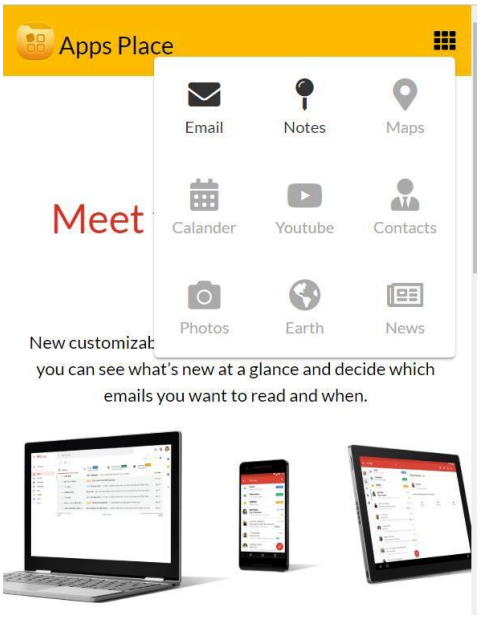
Your favorite apps in one place

Your challenge is to create a single page application with a set of mini apps:

misterEmail, missKeep,
missBooks

The specification for each app is given in a separate document
(missBooks is the app you already have – select one of the team member's projects)

Appsus shall “encapsulate” the different apps and provide a navigation between them. The result should be functional, beautiful and responsive.
Here is an inspiration photo:



Time-Table and Delivery Guidelines

Code shall be delivered to your Dropbox as well as in Git and GitHub Page.
Fill the URL in the submission form.

It is recommended that each team member will be in charge of one of the apps, however, both team members shall be responsible for the successful delivery of the entire project.

Meetings with the team leader

- Wednesday 15:00-17:00 UX and Code Skeleton – First submission 21:00
- Thursday Status meetings – Second submission 21:00
- Saturday – Final submission 22:00

Recommended Development Steps

- Review together our last CRUD project – Books
- Learn the requirements and research the real apps
- Decide how to work together
- Create the app skeleton as described below - so each team member would have a place to work in.
- Setup git and make sure the team can push and pull from the repository

  Commit and Push, see it in Github Pages.

  Note – Git ignores empty folders when committing, so create at least one file in each directory
- Split and work separately and together whenever you see fit.
- Push and pull every few hours to coordinate your code and practice the workflow
- Use this opportunity to improve the code by mutual code reviews

Invest in your demo data

It is required to create some meaningful data, so when the app starts, it shows relevant demo data.

eventBus communication

When components are located at different areas of the DOM we sometimes use an eventBus to communicate. Do note that you cannot communicate with a component that does not exist (e.g. – you cannot communicate with component on different route)

queryString Params

A query string is the part of a uniform resource locator (URL) which assigns values to specified parameters:

`http://example.com/path/to/page?petName=charli&color=black`

(in this case the param **`petName`** gets the value *charli*, and the **`color`** gets *black*)

Query string parameters are a useful way to communicate between routes adding optional parameters. Remember them when integrating the 2 apps.

For example: a link: 'Send as Email' in a <note-preview> component to the <email-compose> component may look like that:

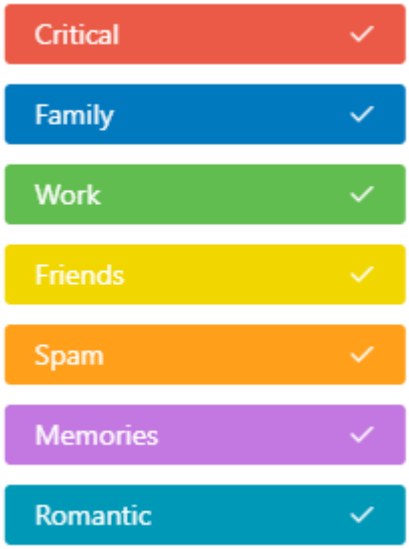
/email/compose?subject=my note&body= note about the rain

Reusable Components

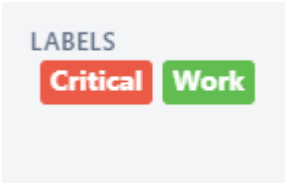
- Use a <long-text> component - gets the text to format as a prop. (this component is used by both apps)
- Use a <user-msg> component for showing success / error messages (this component is used by both apps)

Labels

Code a <label-picker> component and use it in both apps to label an email or a note, the list is fixed, here is the purposed list of labels:



Show the selected labels on the email / note



Folder Structure

Lets review the folder structure for the Appsus app.