1. Design and build one or more web applications to explore and apply the things you have learned thus far.
2. As this is the primary work you will do over weeks 10-14 you should plan on spending at least 30 hours on these challenges. Work with the instructor to manage the scope for your ideas.
3. Once you have settled on an idea or two write it up and submit it here. Because these should be more complex challenges, you should spend more time on this than you may have for the first block to produce a more formal proposal. A sample outline might include the following sections:
4. Purpose
5. Audience
6. Data sources
   1. External API
   2. localStorage
   3. local JSON file.  
      etc.
7. Initial Module list
8. Wireframes for each view of your application
9. Colors/Typography/specific Element styling
10. Schedule to provide yourself mile markers along the way to help you stay on target.

## A Weekly Planner App:

### Purpose and Audience:

Help Students, Teachers, and other Users make personal online schedules for every day/week, for the use of planning School activities, in a Desktop browser. This app should provide adequate measures of time to ensure the user is thinking through their processes.

This can be used by Admin Assistants, Employers, Project Managers, Teachers, Students, Travel Agents, Receptionists, general public, etc., on a per-week basis.

For example:  
 Event A lasts 30 minutes;  
 Event B starts 5 minutes into Event A, and lasts 1 hour and 30 minutes.

This application could provide the user with a *warning box*, asking if the user is sure of the event lineup of Event Bs starting time (is it too soon/consider starting later by 15 to 25 minutes).

This app will also allow the user to print/save the document by the press of a button.

### Data Sources:

1. No external API...? All internal MDN and JS APIs?
2. localStorage
3. local JSON file
4. Uses the *Window.print()* method to print user-generated timesheet. (?)
5. There may be more...

### Initial Module List:

* Event Creator Module - get user input for multiple fields:
  + Date (required)
  + Start time (required)
  + End Time (required)
  + Event Title (required) (can be duplicates by name, but not times during same day)
  + Notes (optional) - These give extra details about the events that are *mini-reminders*, i.e., important micro-details, like *Bring wallet*, *Fill application form*, and others.
* Time Comparison Module - compare events in chronological order:
  + Goes through each event, stepping by (1/2?) and comparing it to the previous event in list (nesting of events?)
    - For Loop of (JSON?) data of events:
      * Looks at *durations* and ***start times***, ensures user is aware of possible *overworking*, and allows for changing to recommendation(s) (defaultInstantChange=false).

### Colors:

* #0076B6 = BYU-I Blue (secondary background) ***<<<Can be changed!***
* #000000 = Black (tertiary background) ***<<<Can be changed!***
* #FFFFFF = White (background and contrast color to blue/black/grays) (primary background, secondary text) ***<<<Can be changed!***
* #DBDBDB = BYU-I button Gray (light on white with dark gray #575252) ***<<<Can be changed!***
* #575252 = BYU-I text Gray (dark on white) (primary text) ***<<<Can be changed!***
* #1C9FEF = icon color (same as the audio headphone for the Immersive Reader button in the top left of the Desktop webpage) ***<<<Can be changed!***

### Typography:

* "Lato Extended", Lato, "Helvetica Neue", Helvetica, Arial, sans-serif; ***<<<Can be changed!***

### Element Styling:

* Each event should be a *div*, with the border of *icon color* (#1C9FEF).

## Schedule:

1. Create Wireframes;
2. Test *Window.print()* functionality;
3. Create mock-up of **one** *event*, for reference, then **two**, and finally, **three** *with overlapping times*;
4. Create Input fields;
5. Populate webpage app content with input values (event creator module);
6. Test and debug event creator module;
7. Code logic of app (time conversion module);
8. Test and debug time conversion module;
9. Run user test(s), debug as needed, correcting errors;
10. Submit final project!