CIS*2750 Assignment 3 grading scheme

All the grades have the form **N/M**

- **N** is the max grade you get for a fully functional UI connected to a fully functional back-end
- **M** is the max grade you get for a fully functional UI without a connected back-end, but with all stubs that "fake" a functional backend.

To get grades for the UI elements, make sure that they are connected to callback functions. Each callback should provide a visible output, as described below.

Your assignment is graded based on the functionality it displays. A broken server (back-end) route for some functionality is the same as not implementing that functionality. In both cases the grade is zero for that part of the assignment. In fact, a fully functional UI with stubs and without the back-end could be worth more marks than a UI that connects to the back-end incorrectly (or results in a server crash), and has no stubs.

For example:

- Calendar view panel functionality is worth 40 marks if fully implemented (UI and JavaScript/C back-end), and 20 marks if the UI is not connected to the back-end, but has all the stubs - i.e. when the web page loads, a list of dummy Event summaries is displayed.
- If the grader selects a file from the pull-down menu, and nothing happens - e.g. because you've messed up an Ajax call or the C library on the server has crashed - you will get a very low grade for this functionality - much less than grade below.
- On the other hand, if the grader selects a file from the pull-down menu and the UI displays a "dummy" list of card details without connecting to the server, you will get a higher grade - up to the lower of the two marks below, assuming the rest of this functionality is up to spec.

As a result, do not submit an assignment with a completely broken back-end and no UI stubs, and expect a passing grade. You should create stubs for all UI functionality, and only replace stubs with a back-end connection once you have verified that the back-end for this functionality actually works.

Grade breakdown

- **File Log Panel (15 marks / 7.5 marks)**
  - correctly updated when a file is uploaded or created
  - displays all the required details in table format, as specified in Module 1
  - includes a downloadable link for every valid file on the server
  - does not display invalid files

- **Upload a file (10 marks / 5 marks)**
  - Contains the file selection window and a button.

Status Log Panel (10 marks / 5 marks)

- Error messages - e.g. uploading invalid file - are displayed here.
- Stub messages (if any) also go here
- Clear button works

- **Calendar View Panel (15 marks / 7.5 marks)** - extracting alarms and optional properties is graded separately
  - Drop-down list contains all valid files on the server
  - Displays all the required details in table format, as specified in Module 1

- **Show alarms (10 marks / 5 marks)**
  - This is done for a specific event
  - Displays all the required details, as specified in Module 1
  - The text must be formatted for human readability - do not just dump a raw JSON string

- **Extract optional props (10 marks / 5 marks)**
  - This is done for a specific event
  - Displays all the required details, as specified in Module 1
  - The text must be formatted for human readability - do not just dump a raw JSON string

- **Create Calendar (15 marks / 7.5 marks)**
  - Create a new calendar, pass it to the parser, save it to the .ics file
  - User enters all details using forms
  - validate user input and do not allow a user to create an invalid calendar - e.g. a calendar with no events
  - When a new .ics file is created, the File Log Panel and all the file lists must be updated to include it

- **Create Event (15 marks / 7.5 marks)**
  - Add an event to an existing calendar (.ics file)
  - User enters all details using forms
  - validate user input and do not allow a user to create an invalid event - e.g. an event with no dates
  - When a new event is added to an .ics file, the File view panel entry for that file must be updated accordingly
  - If the user requests the details for the updated .ics file, the new event must show up in the Calendar View Panel

- **Total: 100 marks (UI+back-end) / 50 marks (UI+stubs)**

Evaluation procedure and deductions

Your code must compile, run, and implement all functionality listed in Module1 and 2. Your code will be compiled using your Makefile, which must create a shared library (see Module 2 for details) that will be used by `app.js`.

Your backend will be compiled and executed on `cis2750.socs.uoguelph.ca` using a port number of our choice

Your Web front ent will be graded using from Firefox on `linux.socs.uoguelph.ca`. It will be accessed as `cis2750.socs.uoguelph.ca:portNum`

Before submitting the assignment, delete your `node_modules` and compiled shared library (`.so` file), run npm install, recompile your shared library using your Makefile, run your server, and make sure everything works!

If your server fails to execute for any reason when we run `npm run dev portNumber` (with a port number of our choice), your assignment grade will be **zero (0)**.

Make sure you compile and run the assignment on the SoCS Linux server before submitting it - this is where it will be graded. You must test your code on NoMachine, using Firefox installed on the SoCS servers. If your GUI does not work correctly on this browser, you will lose marks - up to and including getting a **zero (0)** in the assignment.

If you do not provide a Makefile, you will lose all the marks for the back-end, and your maximum assignment grade will be **50 marks**.

If you provide a Makefile but it fails to compile, you will lose all the marks for the back-end, and your maximum assignment grade will be **50 marks**.

As stated in Module 2, all your Node.js modules must be automatically downloaded when your code is downloaded on the SoCS server and the grader types “`npm install`”. If your A3 back-end does not run due to missing dependencies when we grade it, you will lose all the marks for the back-end, your maximum assignment grade will be **50 marks -** but it can be **zero (0)** if the missing dependencies prevent your server from executing.

You will lose mark for run-time errors and incorrect functionality.

Additional deductions include:

- Any compiler warnings: **-15 marks**
- Any additional failures to follow assignment requirements: **up to -25 marks**
  - This includes creating an archive in an incorrect format, having your Makefile place the `.so` file in a directory other than `CalendarApp/`, modifying the assignment directory structure, creating additional `.js` and `.html` files, etc.