02 - HTML Technology Spike

COMP290 – Large Scale and Open Source Software Development
Dickinson College

Name:

Introduction:

HTML (Hypertext Markup Language) is the fundamental technologies used to structure forms and reports in the FarmData2 front-end. These activities introduce you to just enough essential HTML to get you on the path to doing FarmData2 development. They will also familiarize you with the resources and references that you can return to later to learn more as you need to. Future activities will provide similar introductions to just enough of the other key technologies necessary for FarmData2 front-end development. All of these activities will be done in the context of FarmData2 and will help prepare you to fix bugs and add features.

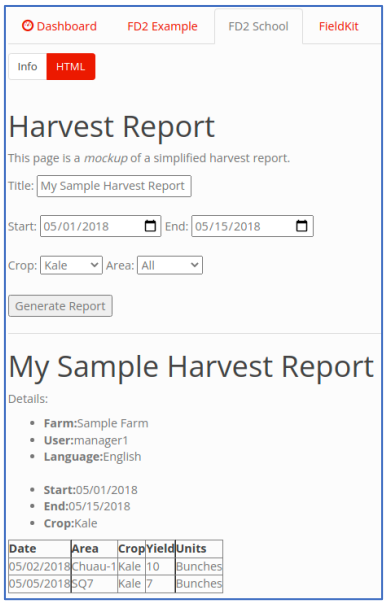
As in the last activity you will be using your fork and clone of the FD2School-FarmData2 repository (rather than the actual FarmData2 upstream repository) for this activity:

- https://github.com/DickinsonCollege/FD2School-FarmData2

The Harvest Report Mockup

At the end of the previous activity, you gained some experience using FarmData2 by looking at the Seeding Report and Seeding Input features. In this activity and the next several, you will learn about FarmData2 development and the technologies by building a *mockup of a Harvest Report feature* for FarmData2. Note that our goal here is not to build the real harvest report feature, but to learn (thus the mockup). The (mockup) report that you build will not have all of the features of an actual Harvest Report and it won't look and feel quite like a full FarmData2 feature. However, by the end of the activities you will be well positioned to begin developing and testing real FarmData2 features.

The initial version of the mockup harvest report that you will build through these activities will look something like the one shown below:



A later activity will then guide you through the process of making the report look more like a full FarmData2 feature (e.g. like the Seeding Report that you used).

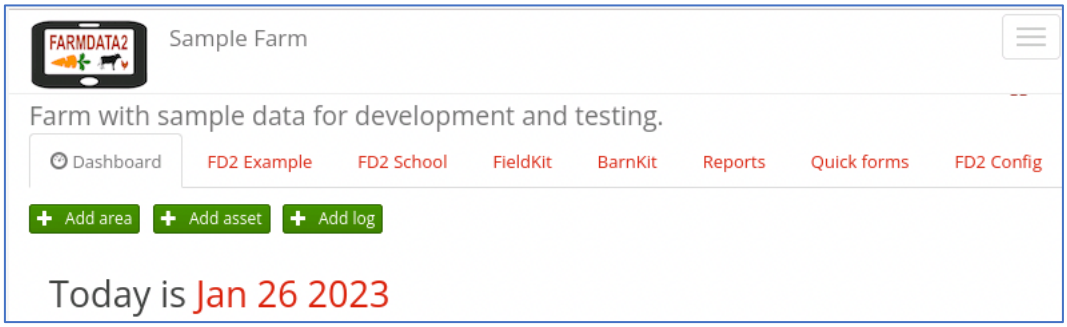
So, let's get started…

Getting Started:

1. Be sure that you have started FarmData2 using the `fd2-up.bash` command and connect to it using the TigerVNC Viewer as you did in the last activity. You might find it helpful to refer back to the INSTALL.md file in the repository if you don't remember how to start and stop FarmData2.

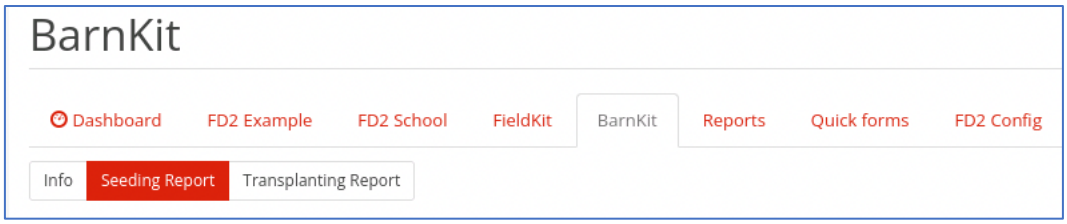
FarmData2 Tabs and Sub-tabs:

When you launch FarmData2 after installation you should see five tabs that are FarmData2 specific: *FD2 Example*, *FD2 School*, *FieldKit*, *BarnKit* and *FD2 Config* as shown below.



The *FD2 School* tab is where you will complete this and future FarmData2 School activities.

The *FieldKit and BarnKit* tabs are where the majority of FarmData2 features live. Each feature within a tab lives in what we will call a *sub-tab*. For example, the BarnKit tab has 3 sub-tabs - Info, Seeding Report and Transplanting report as shown below:



2. The *FD2 Example tab* and its sub-tabs provide a set of examples that illustrate the common UI elements and operations used in FarmData2. These examples are useful references when adding new features to FarmData2 and will be something we return to in later activities.

What sub-tabs are there in the FD2 Example Tab?

Tabs, Sub-tabs and FarmData2 Modules:

FarmData2 features such as the Seeding Report and Seeding Input run within the farmOS application (https://farmos.org/). In turn, farmOS is built on top of the Drupal Content Management System (https://www.drupal.org/). This is relevant here mostly for vocabulary, as each tab in FarmData2 appears in the farmOS interface and is implemented as a *module* in Drupal. This section will familiarize you with the location and organization of the FarmData2 code that creates these modules and sub-tabs. The sections following this will walk you through the process of creating a new sub-tab in the module for FD2School.

3. The code for all of the FarmData2 modules is contained in the `farmdata2/farmdata2_modules` directory and its sub-directories.

a. What subdirectories exist in the `farmdata2/farmdata2_modules` directory? Examine the main tabs in your running instance of FarmData2. How do these sub-directories seem to be related to the tabs that you see in the running FarmData2?

b. What subdirectories exist in the `farmdata2/farmdata2_modules/fd2_example` directory? Examine the sub-tabs of the FD2 Example tab in your running instance of FarmData2. How do these sub-directories seem to be related to the sub-tabs of the FD2 Example tab that you see in the running FarmData2?

Adding a Sub-Tab to the FD2School Module:

Now that you know where the code for the FarmData2 modules lives and a little bit about how it is organized, you'll learn how to add a sub-tab to the FD2School tab. As you work through this and subsequent FD2School activities you will be creating a new sub-tab for each activity.

4. Use the following steps to get ready to add a sub-tab to the FD2School tab.

a. Open a terminal in the FarmData2 development environment.

b. Create a feature branch from your FD2School-FarmData2 `main` branch. Give your new branch the name `<name>-02-HTML` where `<name>` is your name. Also be sure to switch to your new branch. You will do all of your work for this activity in this feature branch.

c. Find the `README.md` file in the `farmdata2/farmdata2_modules/fd2_school` directory. You can open this in the VS Codium editor, or find it in your browser on GitHub. If you opened the file in VSCodium, you can right click its name in the EXPLORER pane to open a "preview" that will show a nicely formatted version of the markdown.

d. For context, read through the README document down to the "The FarmData2 School Module" section.

5. Now, follow the steps listed in the "The FarmData2 School Module" to add a new sub-tab to the `fd2_school` module.

- Your new tab should have the name "HTML"
- The contents of the tab should be contained in a file named `html.html`.
- For now the html.html file should contain just your name

Take a screen shot of the FD2 School tab with your HTML sub-tab open and paste it below:

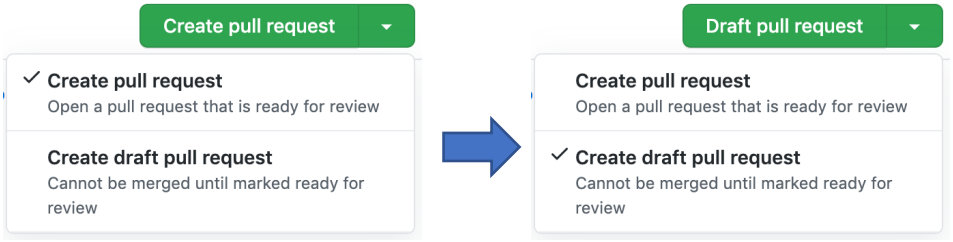
6. As you work through the remainder of this activity, you will be adding to your new HTML sub-tab. As you finish particular questions, you'll be asked to make a commit to your repository with a message about what you have done. This the first time you are being asked to do so.

Stage and commit the changes you have made thus far to your feature branch.

- **Be sure to include a meaningful commit message that describes what you have done** (e.g. Use something descriptive like "Added HTML sub-tab to FD2 School Tab" not something like "Did question 5" or "done" or "edited")**.**
- You can use the `-m` flag on your `git commit` command to add the message. Or omit the `-m`, which will cause the nano editor to open, allowing you to type a longer commit message.

7. Push your feature branch to your origin.

8. On GitHub create a *Draft Pull Request* for your feature branch to the upstream FD2School-FarmData2 repository. You create a Draft Pull Request just like you create a normal Pull Request, but use the drop down on the green "Create pull request" button and choose "Create draft pull request" as shown below:



A Draft Pull Request is just like a regular pull request but is designated as not being ready to be merged into the `main` branch. Since the work you do in these activities is for learning the technologies and will not become part of FarmData2 using a Draft Pull Request here makes good sense.

Learning Some HTML:

The basic structure of all of the FarmData2 specific pages is specified using HTML. These activities will introduce you to the basics of HTML and how it is used to define the elements that make up the content on a page.

Use the Mozilla Developer Network (MDN) HTML Basics guide linked below to complete the following activities:

- https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/HTML_basics

You should read through the guide understanding the contents of each section. There are a few questions below drawn from this guide that are included to ensure that you get the key vocabulary and concepts. Once you have those, you'll add some content to your new HTML sub-tab.

9. Consider the following *strikethrough* (<s>) *HTML element*, which is similar to the paragraph (<p>) element example in the guide except that the text will be ~~rendered with a strikethrough~~:

```
<s>This text has a strikethrough</s>
```

Name the three main parts of an HTML element and identify each in the above element.

10. Consider the following HTML element

```
<p draggable="true">this paragraph is draggable</p>
```

a. What is the *name* of the attribute on this paragraph element?

b. What is the *value* of the attribute you identified in b?

11. Write HTML for a well-formed strikethrough element that has the content "My Old Book Title" and two attributes named *class* and *id*, with the values *revised* and *book-title* respectively.

12. Give properly nested HTML code for a strikethrough element that is strongly emphasized and is part of a paragraph.

13. Rearrange the tags in your previous answer so that they are not properly nested.

Starting the Harvest Report Spike:

14. Let's put the content of the HTML basics guide to use… Using what you have learned about HTML thus far modify the `html.html` file that appears in your HTML sub-tab so that:

- "Harvest Report" and "My Sample Harvest Report" are displayed as level 1 headings.
- The formatting of the bulleted list (including boldfacing) matches the image below.
- You can ignore the italics for "mockup" for now since we haven't learned about that yet, but all other formatting should be the same.
- Be sure that the HTML source code that you write is nicely formatted (e.g. indent it similar to what is shown in the HTML basics guide).

15. Stage and commit to your feature branch the changes you have made to your Harvest Report mockup. **Please include a meaningful commit message that describes what you have done.** Then push your feature branch to your origin.

Note that pushing a branch for which a pull request has been made will automatically update the pull request. Thus, the pull request at the upstream will now contain two commits, one for when you created the new sub-tab and one for your start on the Harvest Report.

More Basic HTML:

HTML defines many elements besides those that were described in the guide. You can find a complete reference to all of the HTML tags in the HTML elements reference:

- https://developer.mozilla.org/en-US/docs/Web/HTML/Element

16. Use the HTML elements reference to answer the following questions. Write your answers in your own words.

a. What does the
 tag do?

b. What does the <hr> tag do? Hint: The description here is not very helpful. Clicking on the tag name will take you to a page that describes that tag and gives examples.

c. What tag should be used to offset idiomatic or technical terms on a page? How is text tagged in this way typically displayed? Hint: Try using the browser's search function to search the reference page to find the appropriate tag. Then click thorough to its page and read more about it.

Adding to the Harvest Report Spike 1:

17. Improve your Harvest Report page by:
    a. Making the term *mockup* into italics.
    b. Adding the horizontal line between the Harvest Report Form at the top and the actual report at the bottom.
    c. Adding line breaks at appropriate places to space out the content.

18. Stage and commit to your feature branch the changes you have made to your Harvest Report mockup. **Please include a meaningful commit message that describes what you have done.** Then push your feature branch to your origin.

HTML Form Elements:

FarmData2 will need to accept user input in a variety of situations. When requesting a harvest report the user will need to enter things such as the date range and the crop or the field (or both) to generate the report. When performing a harvest, the user will have to enter information including the field and crop being harvested and the quantity that was harvested. HTML form elements provide the mechanisms by which users provide these types of inputs.

19. Use the MDN Basic native form controls guide linked below to complete the following activities:
    - https://developer.mozilla.org/en-US/docs/Learn/Forms/Basic_native_form_controls

a. What HTML code would you add to your page to produce the following form element? You can test it in your sub-tab to be sure it works, but you should then remove it.

b. What HTML code would you add to your page to produce the following set of radio buttons? Ensure that only one of the buttons can be selected at a time. Be sure to pay attention to the `for` and `id` attributes in the documentation. You can test it in your sub-tab to be sure it works, but you should then remove it.

20. Notice that the text field does not have a label the way that each radio button does. Using the radio buttons as a model, give the HTML to add the label "Comment:" to your text field above. Note that this approach can also be used to label most form elements. You can test it in your sub-tab to be sure it works, but you should then remove it. Hint: Figure out what *attribute* connects the label to the element!

Adding to the Harvest Report Spike 2:

21. Improve your Harvest Report page by:
    a. Adding the text field for the title with the "Title" label.
    b. Adding the "Generate Report" button as an anonymous button.

22. Stage and commit to your feature branch the changes you have made to your Harvest Report mockup. **Please include a meaningful commit message that describes what you have done.** Then push your feature branch to your origin.

Additional Input Types:

The `<input>` form element is very versatile and what type of form element it displays is controlled by its `type` attribute. The `<input>`: The Input (Form Input) element reference

linked below provides a convenient list of all of the different type attributes that can be applied to the `<input>` element:

- https://developer.mozilla.org/en-US/docs/Web/HTML/Element/input

23. Use the `<input>` element reference to answer the following questions:

a. What `type` of `input` element should be used for entering numeric values?

b. Why do you think it would it be better to use that type instead of just a `text` type?

c. What `type` of `input` element should be used so that the user can enter the start and end dates of the Harvest Report?

d. Note that not all input types are supported by all versions of all browsers. You can click the `type` name in the reference to get a page with information and examples for that `type`. At the bottom of that page there is a "Browser Compatibility" table.

At which version did the firefox browser begin to support the input type that you found for part c?

Additional Form Elements:

In addition to the basic native form controls provided by the `<input>` element there are many additional form elements available in HTML. The MDN Other form controls guide provides an introduction to a number of these additional HTML form elements.

- https://developer.mozilla.org/en-US/docs/Learn/Forms/Other_form_controls

24. What HTML element would be used to create the Crop and Field options?

Adding to the Harvest Report Spike 3:

25. Using the Mock Harvest Report image at the top of this document as a guide, improve your Harvest Report page by:

a. Adding the "Crop" element:

i. This element should have "Broccoli", "Kale" and "Peas" as the possible crops. Make "Kale" selected by default. Use a label to make "Crop:" appear next to this element.

ii. When your element works correctly. Stage and commit the changes to your feature branch with **a meaningful commit message that describes what you have done.**

b. Adding the "Field" element:

i. This element should have "All", "Chuau-1" and "SQ7" as the possible fields. Make "All" selected by default and use a label to make "Field:" appear next to this element.

ii. When your element works correctly. Stage and commit the changes to your feature branch with **a meaningful commit message that describes what you have done.**

c. Adding the "Date" elements:

i. Add the elements for the Start and End date using the `date` input type. The start date should range from 01/01/2014 to 01/01/2022 and have the selected date of 05/05/2020. The end date should range from 05/05/2020 (i.e. not before the start date) up to 01/01/2022 and have the selected date of 05/15/2020.

ii. When your element works correctly. Stage and commit the changes to your feature branch with **a meaningful commit message that describes what you have done.**

In question #25 you made a number of changes and made a commit for each "nameable unit of work." This is a good practice that makes it easy to identify when a particular feature or bug fix was added to the project. Notice though that it is not necessary to push to your origin after each commit. You can make multiple commits and then push to your origin at a later time. Though, there is no harm in pushing after each commit.

26. If you have not pushed your commits from question #25, push your feature branch to your origin now.

HTML Tables:

Many of the reports that are generated by FarmData2 will contain tables of data. For example, in a harvest report there will be one row for each harvesting of a particular vegetable during a

time period. Each row will show information about that harvest (e.g. date, field, quantity, worker, etc…). Similarly, an animal health report might show one row for each health check that an animal has had, with each row showing information about that health check.

This section introduces you to the HTML elements that are useful for creating tables.

Use the MDN HTML table basics guide linked below to complete the following activities:

- https://developer.mozilla.org/en-US/docs/Learn/HTML/Tables/Basics

You should skim quickly down through the guide until you reach the *Active learning: Creating your first table* section. The details for how to create tables will be in the *Active learning: Creating your first table* section and the Adding headers with `<th>` elements section. You can stop before the Allowing cells to span multiple rows and columns section. If the need arises to learn how to do that, you can always come back to it later.

27. Answer the following questions based on the HTML table basics guide:

a. What HTML start and end tags surrounds all of the table content?

b. What HTML start and end tags surround all of the content of a row in the table?

c. What HTML start and end tags are used to surround the content of each cell in a row?

d. The headers for a table are contained in the first row but require different tags. What HTML start and end tags are used to surround the header for each column in the table?

Adding to the Harvest Report Spike 4:

28. Improve your Harvest Report page by adding the HTML table as shown in the Mock Harvest Report image at the top of these activities. You may populate the table with the data shown or other data of your invention. Hint: You can add the attribute `border=1` to the `table` tag to

get the borders to appear. When you complete this exercise, your HTML sub-tab should look like the one at the top of these activities.

29. Stage and commit to your feature branch the changes you have made to your Harvest Report mockup. **Please include a meaningful commit message that describes what you have done.** Then push your feature branch to your origin.

30. When your work on a Pull Request is complete you should convert it from a draft pull request to a regular pull request. This indicates to the project maintainers that your work is ready for review. Go to your pull request on GitHub and convert your Pull Request from a Draft to a regular pull request.

31. In addition to converting your pull request from Draft to regular, many projects will require that you request a review from one of the project maintainers. If you are doing this activity as part of a class, go to your pull request on GitHub and request a review from your instructor.

Optional: To help us improve and scope these activities for future semesters please consider providing the following feedback.

a. Approximately how much time did you spend on this activity outside of class time?

b. Please comment on any particular challenges you faced in completing this activity.