**14 – Bug Reporting (WiD)**

**Activities**

COMP190 – Tools and Techniques for Software Development

Dickinson College

**Name:**

In class we talked about what makes a good bug report (a.k.a. issue ticket, etc.). We saw the elements that every report should include. We learned about the purpose of each and the qualities that make them good. It is not required reading, but you might find a quick read ***How to write effective bug reports*** by Malcom Young a useful review and addition to the class:

* <https://capgemini.github.io/testing/effective-bug-reports>

**Bug Reports:**

1. What are the 5 elements that every good bug report should include?

2. There is an additional element that can be useful in some situations.

a. What is that element?

b. Describe a situation in which that element would be useful.

**Always Search First:**

Ken Fogel in his book Producing Open Source Software says that “Most ticket databases eventually suffer from the same problem: a crushing load of **duplicate** or **invalid tickets** filed by well-meaning but inexperienced or ill-informed users.” To help mitigate the “crushing load” of duplicate tickets you should always search the issue tracker for the bug you plan to report before creating a new ticket.

3. At the bottom of a FarmData2 seeding report there are tables that summarize the Direct Seedings and Tray Seedings in the report. The Direct Seedings Summary looks like:

Graphical user interface

Description automatically generated with low confidence

Most of the information in this table is useful. However, in conversations with Matt (the Farmer who is helping us with FarmData2) indicated that the “Total Row/Bed Planted” does not communicate useful information and should be removed.

a. Use the search bar in the FarmData2 issue tracker to find the ticket that already exists for this bug. Give a search string that matches only the ticket for this issue. Hint: Try a few search strings until you get a set of results that contains the relevant issue. Then modify your search string until it only contains the relevant issue.

Note: By default the search bar in the issue tracker contains the *filters* is:issue and is:open. This indicates that any search will look only through tickets that are open, skipping any closed issues or pull requests. If you are curious, you can look at the filters drop down for more information on filters.

b. What is the issue number and title for the ticket?

c. Read the ticket that you found in part b. Use the criteria from class to assess the quality of this ticket. Write a few sentences answering the following questions. Is it a good ticket overall? What is good about it and why? What could be improved and how?

4. The FieldKit and BarnKit tabs each have sub-tabs for their different features. For example, the FieldKit has tabs for “Info” and the “Seeding Input.” If you look at both the FieldKit and the BarnKit tabs you’ll notice that the ordering of the sub-tabs is inconsistent. The “Info” tab is on the left in the FieldKit and on the right in the BarnKit.

a. Use the search bar in the FarmData2 issue tracker to find the ticket that already exists for this bug. Give a search string that matches only the ticket for this issue.

b. What is the issue number and title for the ticket?

c. Read the ticket that you found in part b. Use the criteria from class to assess the quality of this ticket. Write a few sentences answering the following questions. Is it a good ticket overall? What is good about it and why? What could be improved and how?

**GitHub Markdown:**

You should recall that one of the things that makes a good bug report is to use nice formatting. The ticket that you read in the previous question contained a numbered list that helped organize the steps required to reproduce the bug. Other tickets might use italics, bold face, bulleted lists or links to help in describing the issue. In the GitHub issue tracker (and many others as well) you can use Markdown to format tickets. In fact, GitHub allows you to use Markdown for formatting pretty much anywhere you are asked to write information.

Use the following GitHub Markdown reference to answer the questions below:

* <https://docs.github.com/en/github/writing-on-github/getting-started-with-writing-and-formatting-on-github/basic-writing-and-formatting-syntax>

You can test your work by creating a new issue in one of your own GitHub repos and switching between the Write and Preview tabs, or by using a number of different markdown editors (e.g. <https://jbt.github.io/markdown-editor/>).

5. Give a line of Markdown that will accomplish each of the following tasks.

a. “Another Example” as a third largest heading.

b. Use **Bold** and *italic* text to ***draw attention***.

c. A link with the text Dickinson College that links to the Dickinson home page.

d. A numbered list with your three favorite foods.

e. The following if statement formatted as a block of code:

if (x > 0) {

print “Yep, I’m positive”;

}

**Writing a new Ticket:**

As you know, FarmData2 is under active development. Thus, there will be bugs, issues, missing features, etc… that should be documented in the issue tracker when they are found. Listed below are a few “suspect behaviors” that have been observed, but do not yet have bug reports for them in the issue tracker (I know… I searched for them ☺). Your job in this section will be to write a new bug report ticket for one of these suspect behaviors.

The Suspect Behaviors:

* The main table in the Seeding Report is supposed to be sorted by the values in the date column. It seems to be most times, but sometimes it is not. The different behaviors seem to be dependent on the date range.
* Sometimes in the Tray Seeding Summary table at the bottom of a Seeding Report, the “Total Number of Tray Seeds Planted” and “Average Seeds Planted per Hour” are reported as NaN (Not a Number) instead of the proper values. The different behaviors seem to be dependent on the date range.
* Each row of the main table in the Seeding report has an edit button. Clicking the edit button on some rows seems to cause that row to disappear. The different behaviors seem to be dependent on which row is edited.
* When creating a Seeding Input record, it is possible to enter values that are invalid.

For the behaviors that depend on the date range it will also be useful to know that the sample data provided with FarmData2 includes:

* Jan 1, 2019 - July 15, 2020

If you are interested there is a complete description of the sample data available in the FarmData2 repository in the docker/sampleDB/README.md file. Or you can find it in GitHub at:

* <https://github.com/DickinsonCollege/FarmData2/tree/main/docker/sampleDB>

6. Explore the suspect behaviors described above using your FarmData2 developer install from A13. Pick one of the suspect behaviors and create a new ticket for it in the FarmData2 issue tracker at:

* <https://github.com/DickinsonCollege/FarmData2/issues>

As you do, be sure to follow the guidance on the Process Recap slide from class.

Give the title and URL of the ticket that you created as your answer for this question. Your instructor will review your ticket in the issue tracker.

**Upload to Your WiD Repository:**

COMP 190 is part of the Writing in the Discipline (WiD) thread that runs through the computer science major. In each course on the WiD thread you will complete a writing assignment and add it to your WiD Repository on GitHub. The bug report that you have written is the Writing in the Discipline (WiD) assignment for this course.

7. A WiD Repository has been created for you on GitHub. If this is your first WiD course in computer science you will have received an e-mail inviting you to join that repository. Please accept that invitation. Log into GitHub and then visit the Dickinson-COMP-WiD organization on GitHub to find your WiD repository:

* <https://github.com/Dickinson-COMP-WiD>

If you do not see your repository in that organization, check your e-mail (including your spam folder) for the invitation. If you still cannot find the invitation, get in touch with your instructor as soon as possible to resolve the situation.

Give the URL of your WiD repository on GitHub.

8. The following set of instructions will guide you through the process of uploading a pdf of your ticket to your WiD repository:

* Log into GitHub with your username and password.
* Navigate to your WiD repository in the Dickinson-COMP-WiD Organization using the link below:
  + <https://github.com/Dickinson-COMP-WiD>
* Create a folder named COMP190 for this course in your repository. See the link below for detailed instructions:
  + <https://github.com/KirstieJane/STEMMRoleModels/wiki/Creating-new-folders-in-GitHub-repository-via-the-browser>
* Print the ticket that you created for this assignment to a pdf file.
* Drag and drop your pdf file into the COMP190 folder that you created. See the link below for detailed instructions:
  + <https://help.github.com/en/github/managing-files-in-a-repository/adding-a-file-to-a-repository>

Note: Your WiD portfolio is a git repository stored on GitHub. So, if you prefer, you can also manage your WiD repository on your local machine using git. Simply clone your WiD repository to your local machine, make changes there (i.e. create the COMP190 directory and copy your pdf to it), stage and commit the changes, and then push main branch back to your WiD repo on GitHub. NOTE: In this case it is okay to commit your changes to the main branch.

**Optional Extras:**

The following activities are optional, but may be of interest if you would like to engage more deeply with the FarmData2 project. Feel free to do them in either order, or just do one or the other. Any efforts to contribute to the improvement of FarmData2 are welcome!

9. Document any other suspect behaviors you have observed in FarmData2, or any thoughts on ways that its existing features could be improved in the issue tracker. Be sure to search for existing tickets first. If you find one, add your thoughts to that ticket using the comments. If you do not find a ticket, then create a new one. Give the URL’s for any tickets that you

10. Search through the issue tracker for “good first issue” tickets. Read a few of these and see if you can solve one. If you do work on the code, be sure to do your work on a feature branch as we learned about earlier in the semester. Push your branch to your origin on GitHub and create a “draft” Pull Request linking to the ticket in the issue tracker. The FarmData2 project managers will then be able to review and comment on your work. Give the URL for any pull requests that you create.

**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.