Project Information:

Website URL: http://csc415-server35.hpc.tcnj.edu:3000

Implementation URL: http://csc415-server35.hpc.tcnj.edu:3000/checklists

VM: student3@csc415-server35.hpc.tcnj.edu

Password: 7iG3f6789

Project Path: /home/student3/CSC415-Growstuff-Watering

Github Project Name: CSC415-Growstuff-Watering

Github: https://github.com/saremshal/CSC415-Growstuff-Watering

Branches:

- Dev
    - Contains Docs
- checklist
    - Contains Docs and Prototype

Notes on Github Issues and Milestones:

When looking at github, you will notice my project does not have any issues or milestones. This is because I forked the project from Growstuff and they have disabled raising issues and milestones within a forked project.

Project Description

The project will focus on adding a watering notification feature to Growstuff!

The features it will include are:

- An Alert System to let you know when to water your crops
- Customizing which crops to receive notifications from
- A new crop page for watering to let you know which crops you should water and to let you check off the ones you have already watered
- Ability to edit when the last time you watered your crop so that you can specify when you watered a crop if it was not on the same day you checked it off
- Ability to customize crop characteristics to set how often you should water them if the default settings are not to your liking

How to to Run Project on VM

1. Login with “ssh student3@csc415-server35.hpc.tcnj.edu”
2. Type the password “7iG3f6789”
3. Go into rails project file “cd CSC415-Growstuff-Watering/”
4. Run the server “rails server --binding 10.18.6.35”
5. Open the webpage http://csc415-server35.hpc.tcnj.edu:3000/

How to Use Application

- Once you have a plant added to a garden, you can click a button on the plant's page to add it to your checklist.
- You can then access your checklist from the profile drop-down menu.
- On your checklist page, it will show you what plants you should water and which plants are coming up to be watered.
- By ticking or unticking a checklist row and saving the row, you can adjust your checklist to indicate which plants you have watered and which plants you need to water.
- You can also edit the day you last watered the plant and the application will calculate when you should water it next and make the appropriate adjustments to your checklist.
- There are a few accounts available for you for testing this app
    - Available Accounts:
        - {User: test1, Password: password1}
        - {User: test2, Password: password2}
        - {User: test3, Password: password3}

Know Bugs/Issues

- The notification mailing is currently not implemented due to TCNJ's firewall and how it blocks outgoing email messages from the server
- Adding/Removing a plant is not fully implemented at the moment due to a known issue with postgresQL, active_median, and growstuff. The effects of this issue prevent any adding, editing, or removing of plants in the database. After attempting to solve the issue,

it's fix seems much more complex than my current skill set would be capable of. The issue has been documented here https://github.com/Growstuff/growstuff/issues/1757 .

    - For testing purposes, I have filled the database with plants to the checklist due to this issue.

- The checklist currently has a 1:1 ratio for saving a checklist tick/untick. This can make larger checklists very annoying to use and is not great UI design. Ideally what I would have liked to done was to have all the ticks for each object saved when the save button is pressed. However doing this seems to be more complicated than anticipated, hence why it is not in this current iteration of the design.
- There is currently no message that shows up when a checklist is empty to let the user know why it is empty
    - Reason being they do not have any crops registered for a checklist

Recommended Testing Outline

1. First log into test3 by clicking the sign in button
    a. Available Accounts:
        i. {User: test1, Password: password1}
        ii. {User: test2, Password: password2}
        iii. {User: test3, Password: password3}
2. Go to “Show Me My Garden” → “Garden” → “Alexanders”
3. You should see a message of “You are currently not tracking the watering days of this plant” and a button to start tracking
    a. When you click the button, it should lead to an exception due to know issues
4. Click the profile name on the top right and open the drop-down menu
5. Then Click Checklist
6. You should see 2 plants and two tables
    a. One table for plants you need to water and another table for plants that are coming up
7. Try checking and unchecking the boxes on the checklist and make sure to save the checkmark
    a. When checking it, it should exist in the “Coming Up” Table
    b. When unchecking it, it should exist in the “Need to Water” Table
    c. You will also see the next watering days change as you do this
8. Try to edit the last day you watered it

   a. Try setting it to a day that has passed. This should send the object to the “Need to Water” Table and adjust the next watering day accordingly
   b. Try doing it to say that is coming up in the future. This should send the object to the “Coming Up” Table and adjust the next watering day accordingly
9. (Optional) Try removing the object
   a. WARNING: Due to the fact that adding objects is not implemented, there is no current method to recover the datas as the application stands (I however have made some loopholes for adding more data, but that is not something a typical user would have access for) So, please do not delete the objects unless absolutely necessary.
10. Logout of the account
   a. You should now see a message asking you to login to view a checklist
11. Log into test1 or test2
   a. You should now see different plants than the plants you had seen before
   b. Feel free to also repeat steps 4-10 with these user accounts as well.
      i. Please note that step 2 and 3 are currently broken on these users due to the active_median issue as mentioned before

Test Cases

GREEN - Fully Functional

YELLOW - UI is Implemented

RED - Not Implemented yet

Table 1. Test Case Table

Functionality Tested	Inputs	Expected Output	Actual Output
Select Crop to get Notification From [Use Case] - Should be able to check off a box to enable receiving notifications.	The user clicks a button (enable tracking).	Planting's Notification Attribute should be true and it should have a valid Next_watered date value. Member's planting notification count goes up by one.	There is a nonfunctional button to add a crop from the checklist.
Select Crop to get Notification From [Exception 1] - The user no longer wants to receive notifications on a planting.	The user clicks the button (disables tracking).	Planting's Notification Attribute should be false. Member's planting notification count goes down by one.	There is a nonfunctional button to remove a crop from the checklist.
Select Crop to get Notification From [Exception 2] - The user deletes the crop from their garden.	The user deletes the planting with notifications enabled.	Member's planting notification count goes down by one. Notifications for the planting should no longer be received.	Instead of having an automated way of deleting, it can be manually deleted from the checklist menu.
Check off Which Crops Were Watered Today [Use Case] - The user should be able to check off any crops they have watered today	The user checks off one of their crops.	The next day value for the crop should update to "today's date + the watering frequency days" and the old value of next day should be saved into the Last_watered variable.	When checked off and the save button is pressed, it updates the values appropriately as specified in the expected outputs and switches the plant object over to the

and its next watering day value should update		The next watering day should also update on the checklist page.	coming up to water table
Check off Which Crops Were Watered Today [Exception 1] - The gardener wants to undo a checkoff.	The user unchecks off one of their crops.	The next day value is replaced with the Last_watered value. The next watering day value updates on the checklist to show the Last_watered value.	When unchecked off and the save button is pressed, it updates the values appropriately as specified in the expected outputs and switches the plant object over to the coming up to water table
Check off Which Crops Were Watered Today [Exception 2] - The gardener has no plants registered.	The user opens their checklist.	The page shows a message on it that lets the user know they currently do not have any plants registered.	Currently if nothing is registered, both of the tables will show up and not be filled with any values.
Edit What Day the Crop Was Last Watered [Use Case] - The User should be able to adjust what was the last day they watered a plant	The user has a new “last watered day” selection	The last watered day should update with the user’s selection and the next watered day should update with it.	When the user updates the value, it will correctly edit the last day watered and update the next day watered and make these updates on the checklist table as well.
Edit What Day the Crop Was Last Watered [Exception 1] - The user no longer wants to change what was the last day they watered	The user clicks the back button to stop the “last waerted day” selection	The user should be sent back to the checklist screen and the plant they were going to edit should not be changed at all.	When the user wants to not update it anymore, they can click a back button and it will take them to the checklist screen and the plant will have the same values as before

Open Source Maintenance and Communication

- Where do they organize issues and bugs?
    - All issues and bugs should be addressed on the main growstuff github page. The developers have provided an array of tags for you to appropriately tag the issue you want to raise and they encourage all discussion to be done here as well.
    - https://github.com/Growstuff/growstuff/issues
- Where do they discuss ideas for the project?
    - All feature discussion should be posted on the main growstuff github page under issues. The developers have provided a tag for “feature_requests”. If you would like to suggest a feature, please raise an issue and use this tag.
    - https://github.com/Growstuff/growstuff/issues
- What processes do they require you to follow in order to contribute to their project?
    - The growstuff community encourages using extreme programming when developing a feature for the growstuff application. They recommend being involved in the community and working with someone else in order to ensure the quality of the code.
    - They also have a few notes on some coding styles for controllers and views in order to help ensure the application is consistent. This are mostly small things like prevent spaghetti code and utilizing feature tests
        - https://github.com/Growstuff/growstuff/wiki/Coding-style-guide

- When a developer has finished their feature, they should make a pull request to the “dev” branch which will be analyzed and given feedback from the community within the next few days.

- How do they determine which code is accepted?
  - When a developer has submitted a pull request, the community will analyze their code. From there the community will give their feedback if they see any glaring issues. Once the code has been reviewed and no issues have been discovered, then the pull request will be accepted and merged with the “dev” branch.
- Steps to take in order to have code accepted (Summary)
  1. Utilize Extreme Programming and get a partner
  2. Interact with the Community
  3. Raise an issue that you are going to work on a feature
  4. Fork the Project
  5. Develop your feature
  6. Make a pull request to “dev”
     a. Case 1) Pull Request is given feedback
        i. Add feedback to project
        ii. Repeat step 6
     b. Case 2) Pull Request is accepted
        i. Feature is merged with dev
        ii. Congratulations, you successfully contributed to the application!

Licensing

- Growstuff uses AGPL-3.0 licensing.
- Included Features
    - Commercial IUse
    - Modification
    - Distribution
    - Patent Use
    - Private Use
- Unincluded Features
    - Liability
    - Warranty