SFU
FACULTY OF
APPLIED SCIENCES

Thunkable Tutorial: My Garden App

Authors:

- Alyssa Rusk, Danny Yu, Eva Yap, and Sunil Mann of Applied Sciences (APSC) Outreach at Simon Fraser University

Versions:

- 1.0 – 21-08-11
- 1.1 – 21-09-03

Attribution:

- This work, “Thunkable Tutorial: My Garden App,” is a derivative of “Thunkable Tutorial Food Court App” by Chris Kerslake of XModus Learning 2020, 2021, used under CC BY.

Table of Content

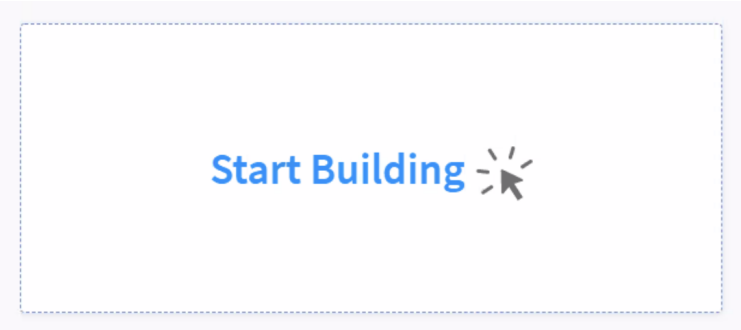
Getting Started

What is Thunkable?

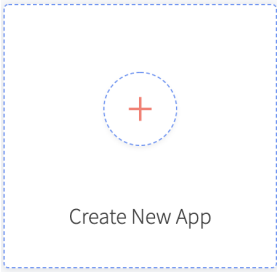
- Thunkable is a drag-and-drop app builder.
- Blocks of code are used to program apps on Thunkable.
- You can create apps using Thunkable and run them on mobile devices.

Create New Project

Once you logged into your Thunkable account, create a new project. If you don't have any existing projects, you should see a prompt that says "Start Building."

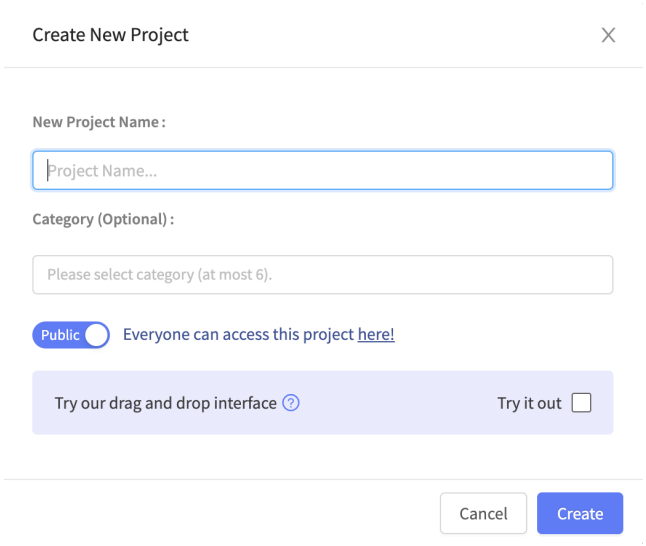


If you have other projects, there should be a button like the one below. Click on it to create a new project.



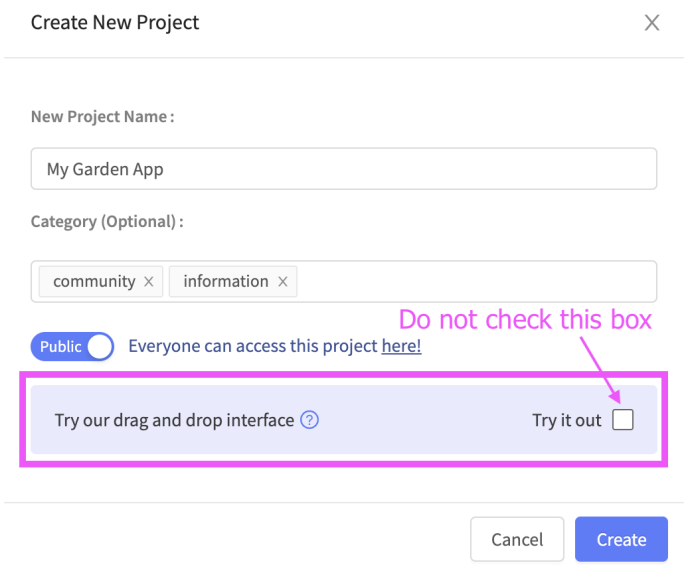
Give your new project a name. You can select a category or categories if you want. This is optional.

Unless you have a paid account, your project will be public. This means that anyone can view and make a copy of your project to remix. Don't worry as they won't be able to make changes to your original project!



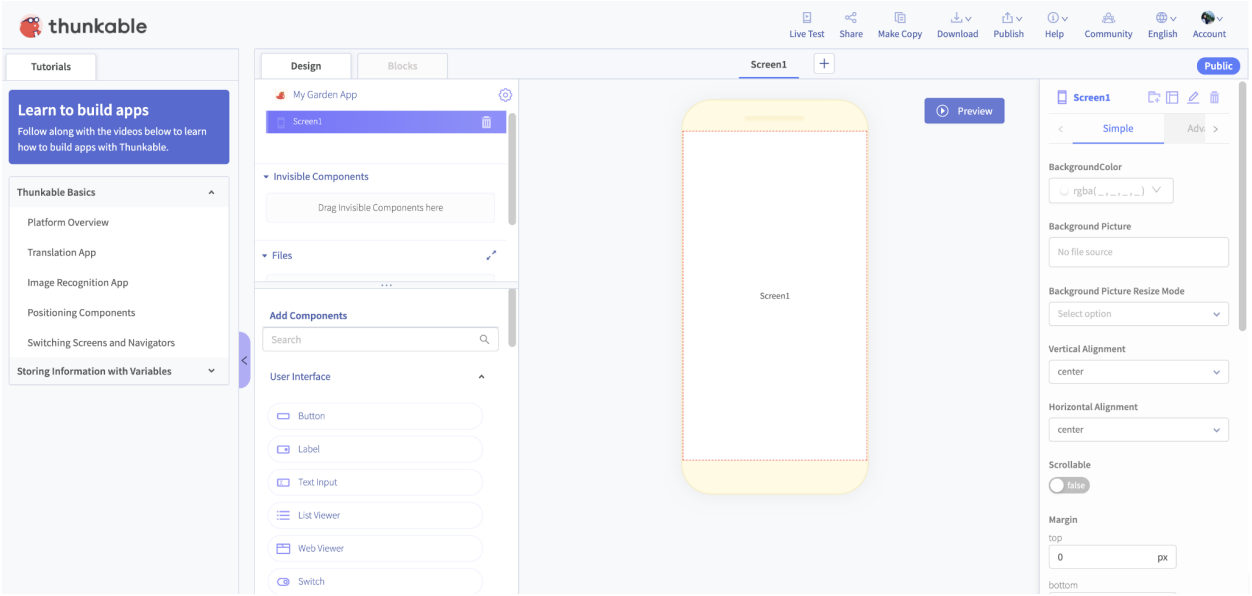
Drag and Drop Interface

This tutorial will not use the drag and drop interface of Thunkable. Please make sure you uncheck this option before moving forward.



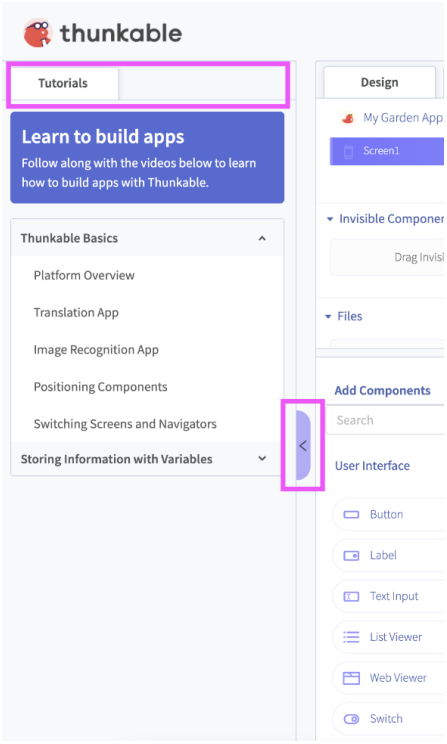
Quick Intro to Thunkable

Once you click on the Create button, you should see the following. Let's have a quick Thunkable intro to familiarize ourselves with this interface.



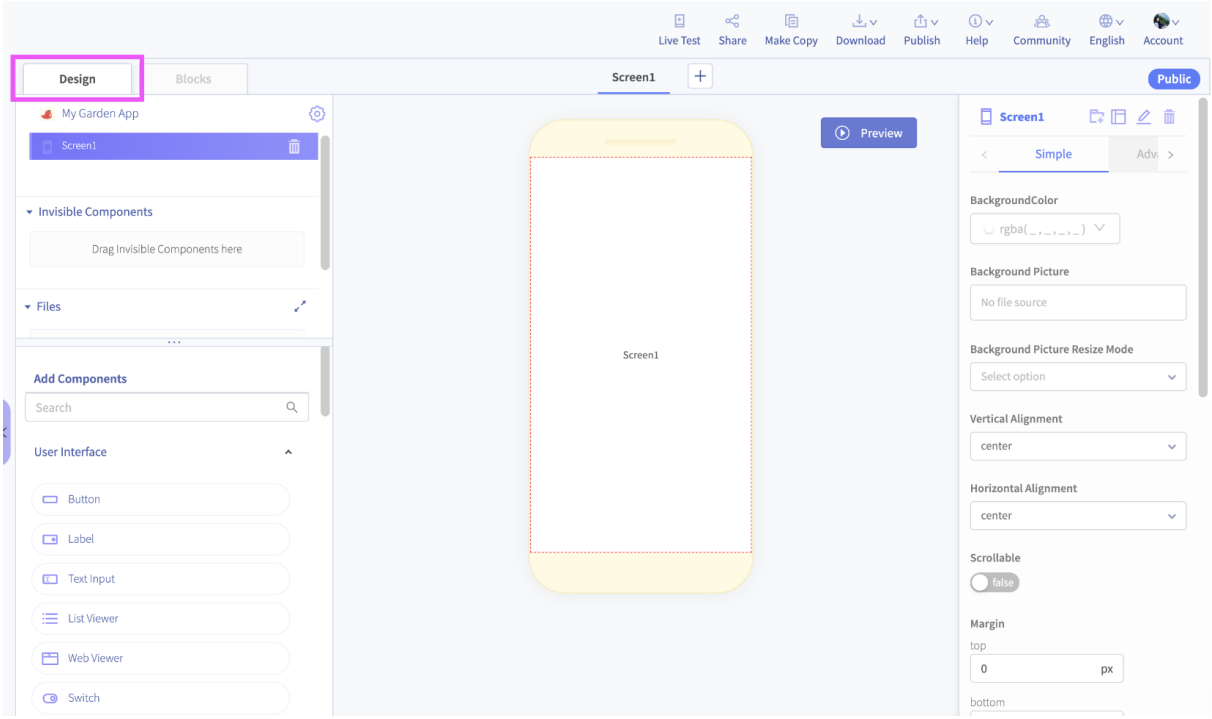
Thunkable Tutorials

You will see some tutorials on your left, which you can use to learn about Thunkable. Click on the purple slider to close the Tutorials tab.

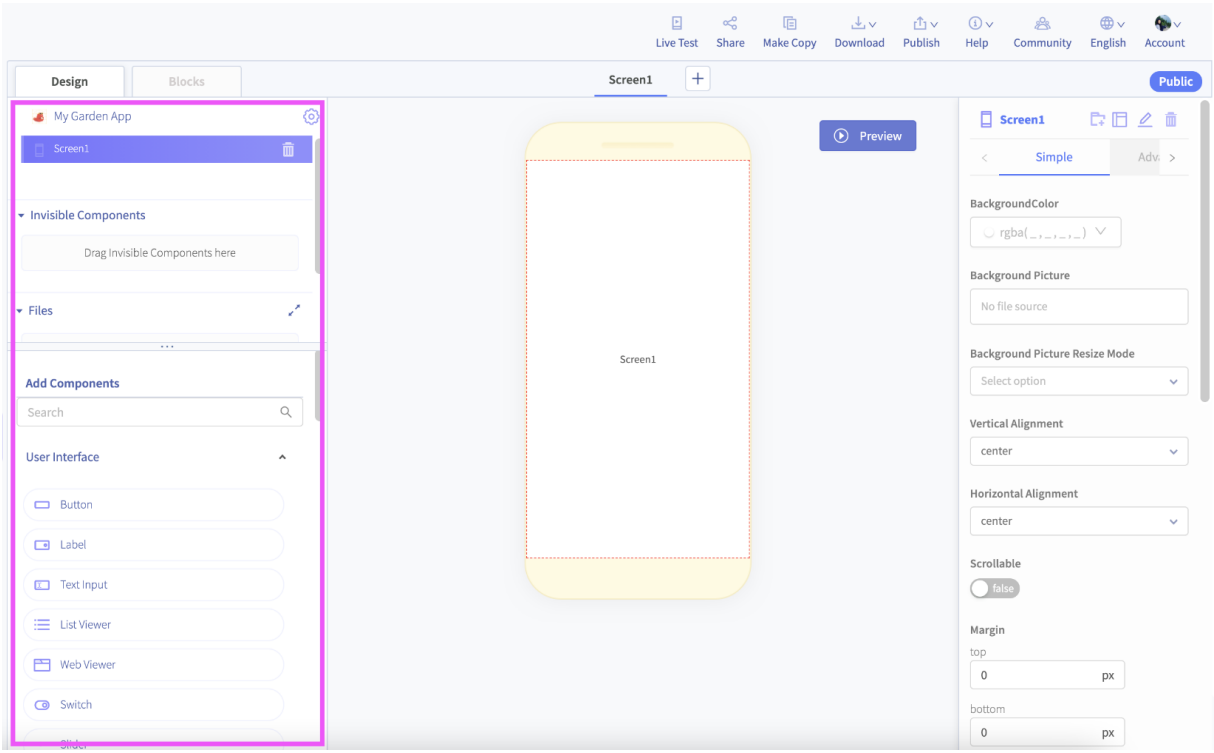


Design tab

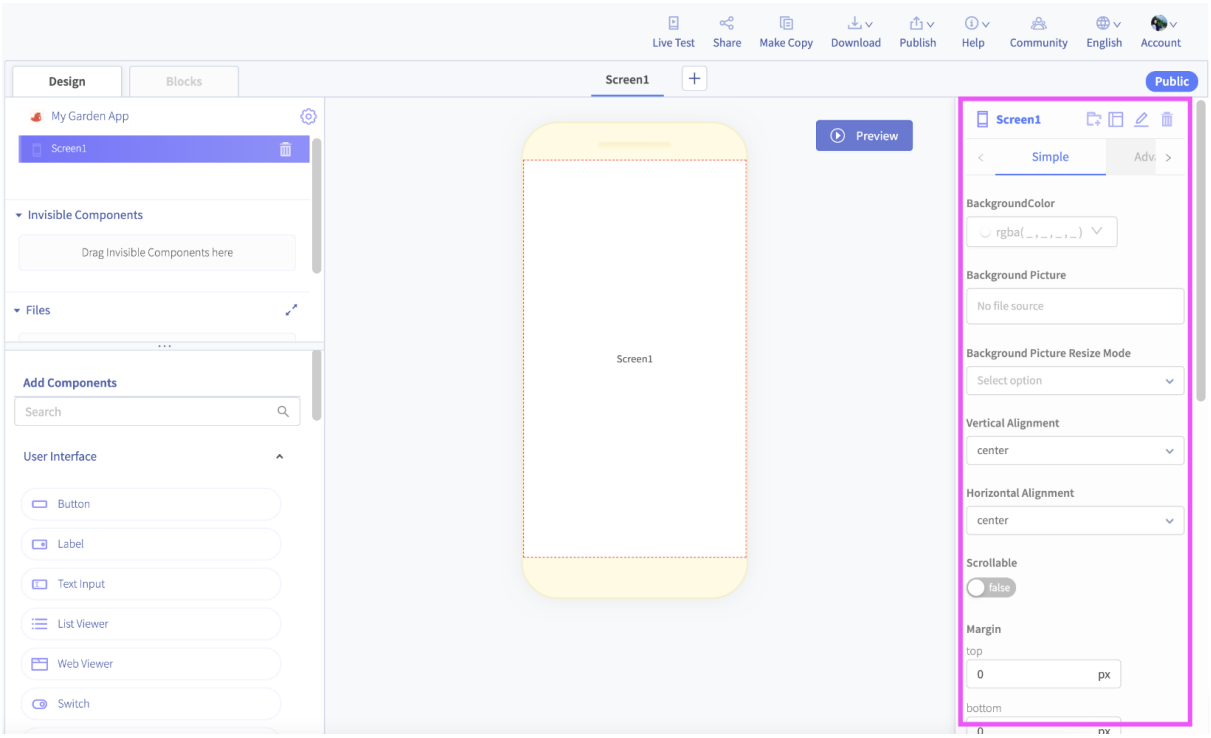
This is your view when you're on the design tab.



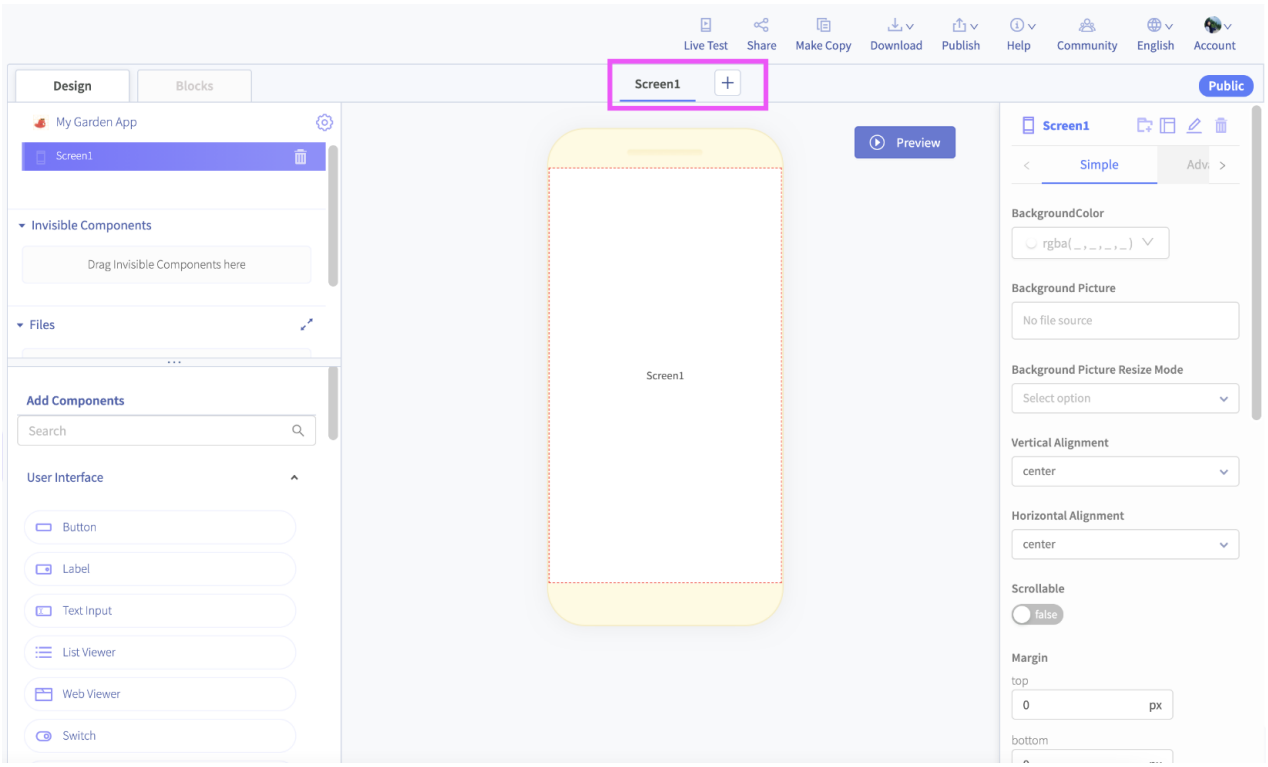
Project components and Add Components are on your left.



Properties/attributes are on your right.

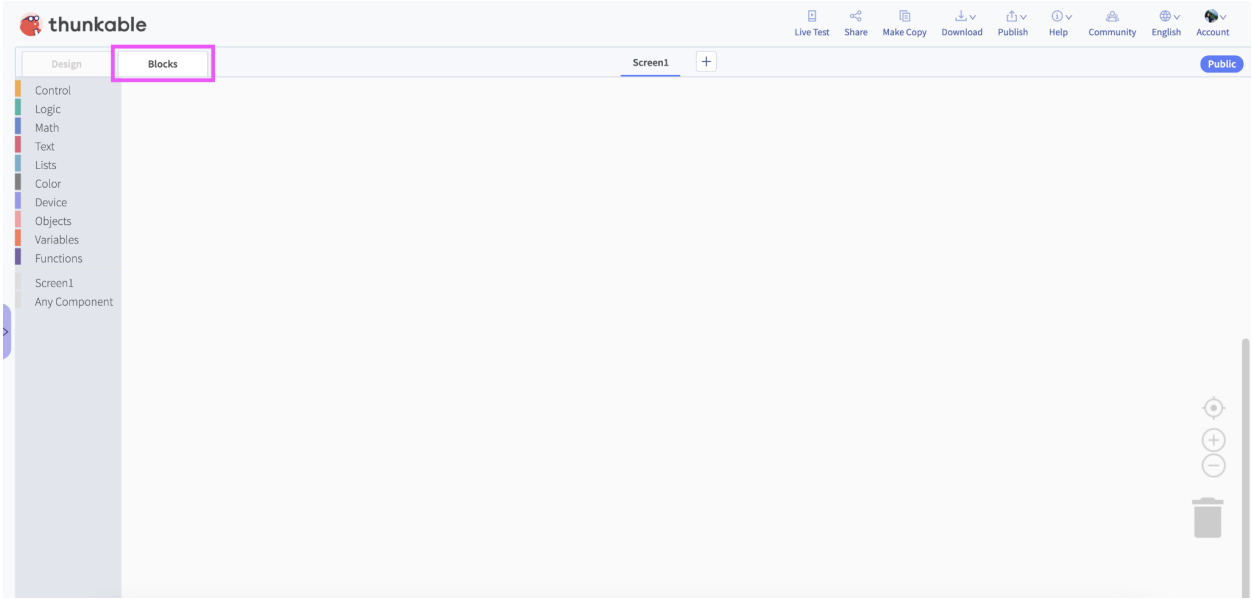


You'll find the list of screens in the middle. To add more screens to your app, click on the plus button.



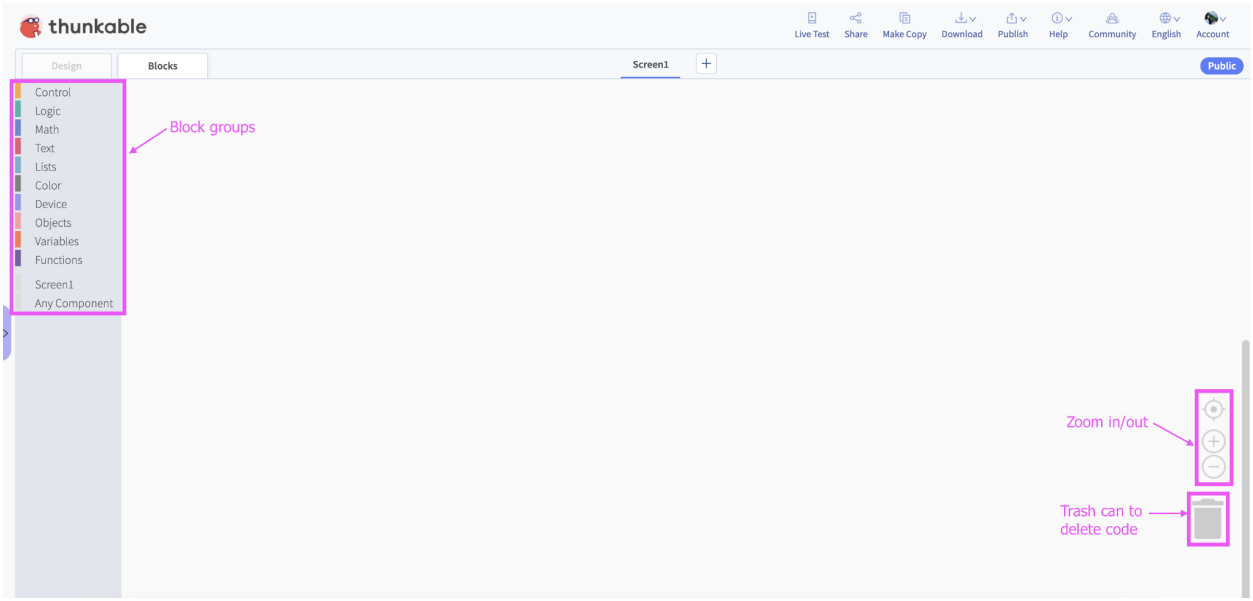
Blocks tab

This is your view when you're on the blocks tab.



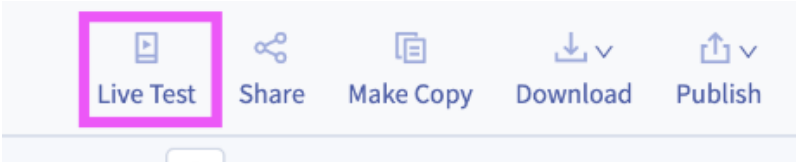
You'll find block groups on your left. On the right, you'll find the zoom in/out buttons and the trash can to delete your code.

You can also delete your code using the delete button (Mac) or backspace button (Windows). Another way to delete your code is to click on the block you want to delete, right click, and select Delete Block from the list of options.



Live Test Your App

You can live test your app in the web browser by clicking on the Live Test button on the top.

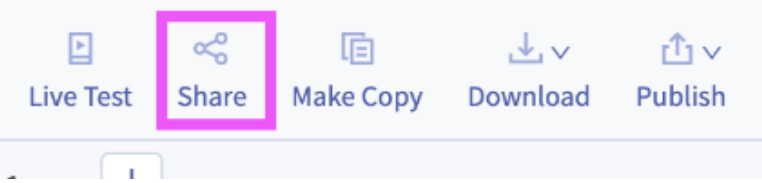


Mobile device and tablet

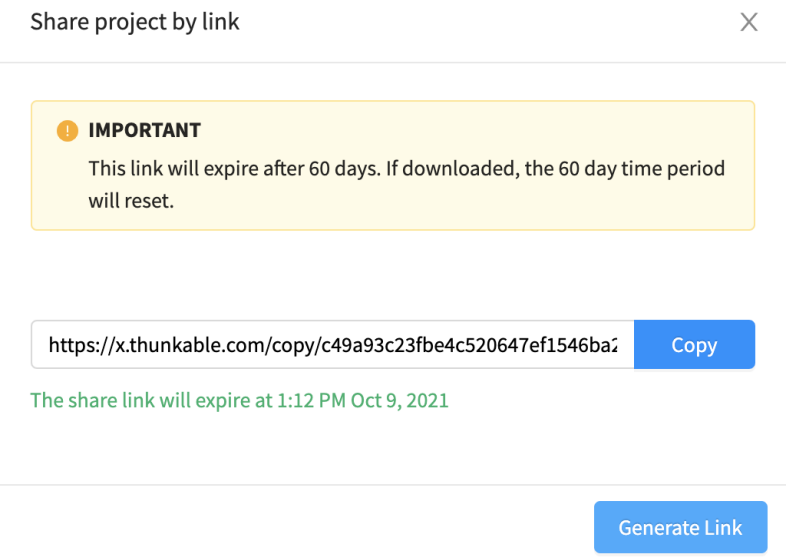
You can also live test your app on a mobile device or a tablet but you will need to install the Thunkable Live app. This app is free and you can download them from the Play store and the Apple store.

Share Your Project

Click on the share button to share your project with someone.



This will prompt you to generate a link for your project, which will be valid for 60 days.

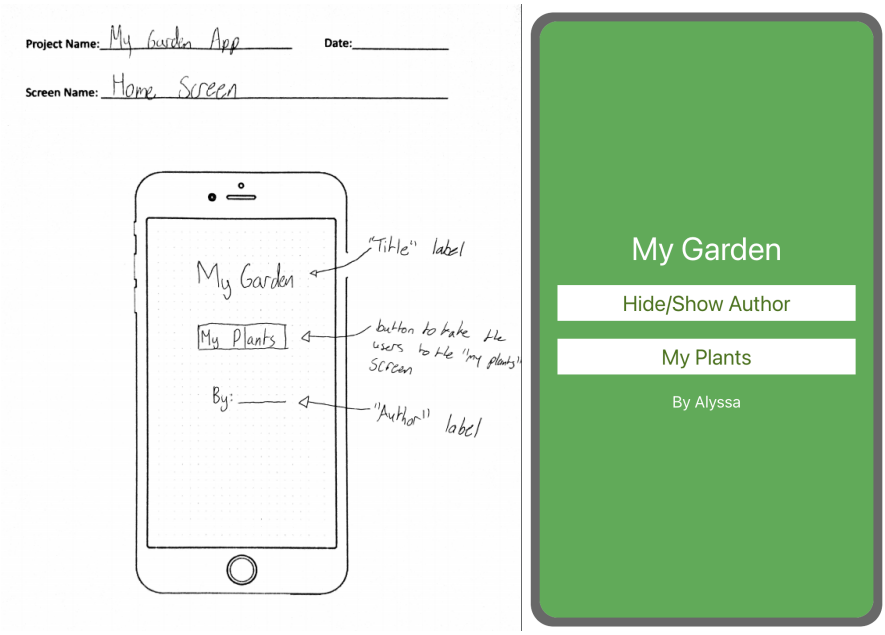


My Garden App

The purpose of this app is to help users keep track of plants near their area. This could be their own garden, a community garden, or a garden at their school. The app will also allow users to rate how good the plants are for local pollinators, which will encourage users to plant pollinator-friendly gardens and raise awareness about the need for pollinator conservation.

Home Screen

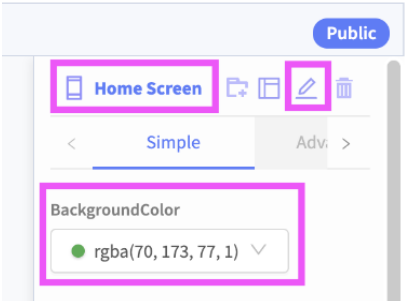
Home Screen Wireframe



Step: Create Home Screen

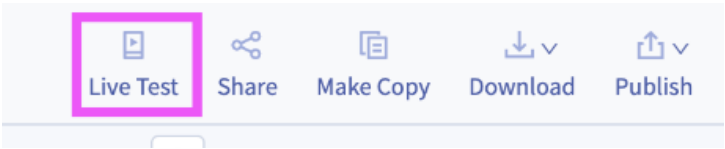
This is the main screen of the app. On this screen, you will add the hamburger button for the drawer navigator, app title, and your name.

1. Rename your screen from the default Screen1 to Home Screen. You can do this by clicking on the edit icon (this icon looks like *a pencil on a line or a piece of paper*) on your right, which is the properties/attribute section.
2. Let's also change the background colour of the Home Screen. I'm going to pick green.

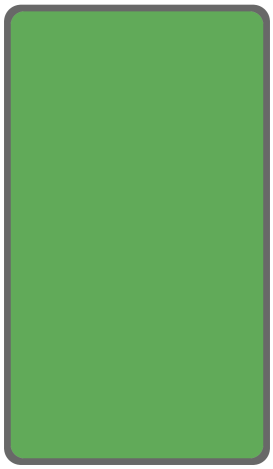


Live Test

Do a quick live test by clicking on the Live Test button.



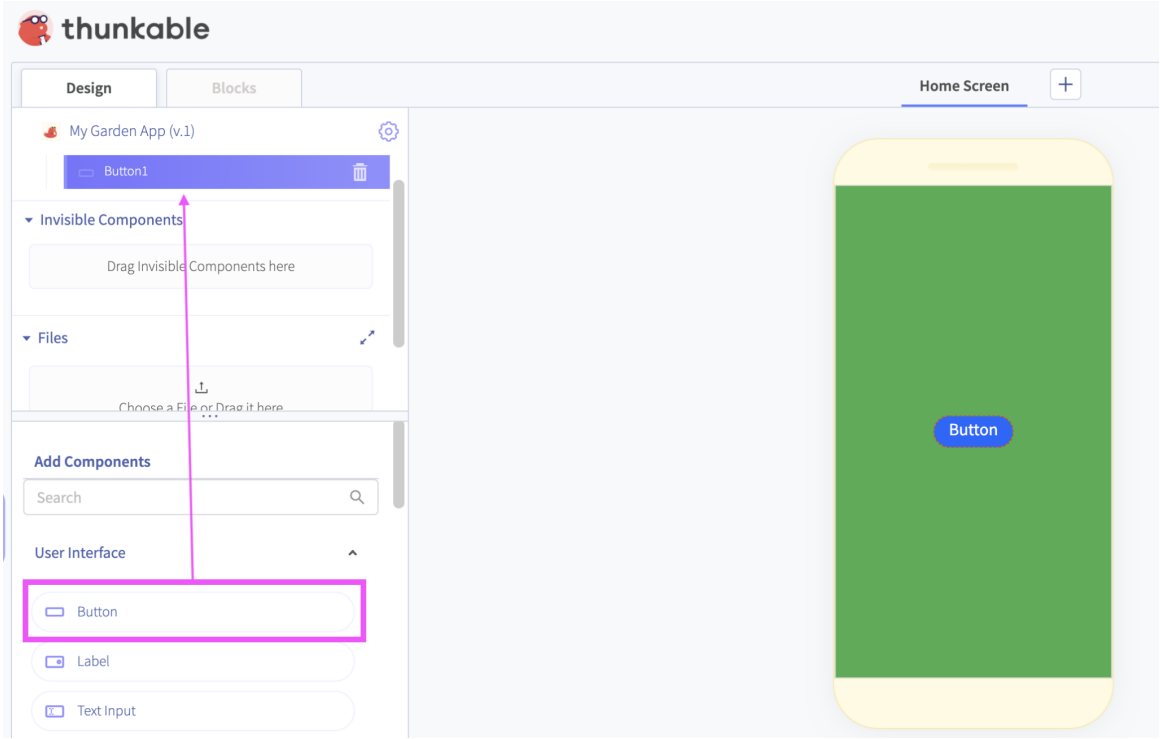
You should see a blank screen with the colour you picked.



Step: Add button to Home Screen

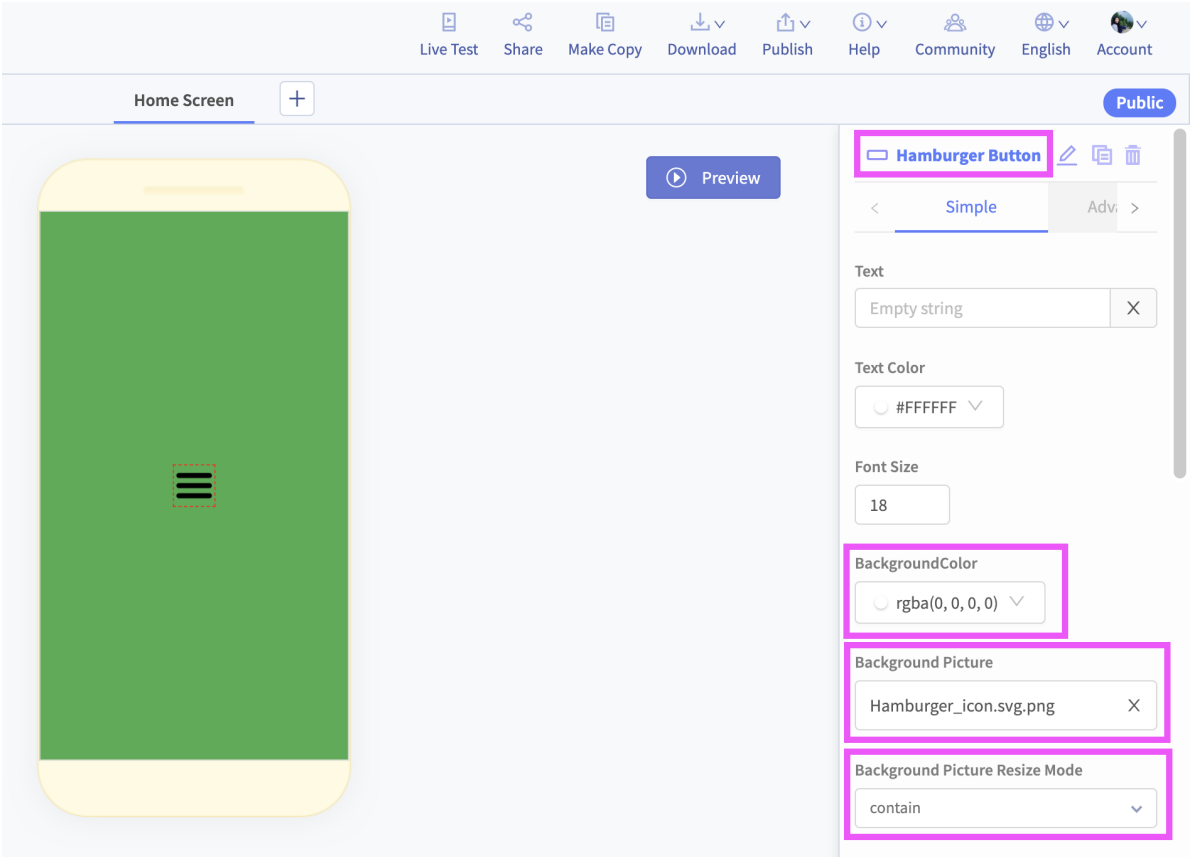
Let's add the hamburger button so we can set up the drawer navigator in our app.

1. Locate the Button component under the Add Components section.
2. Drag and drop this to the project component section on the top. Alternatively, you could also drag and drop the component on the screen. I prefer to drag and drop components to the project component section. It's less laggy that way.

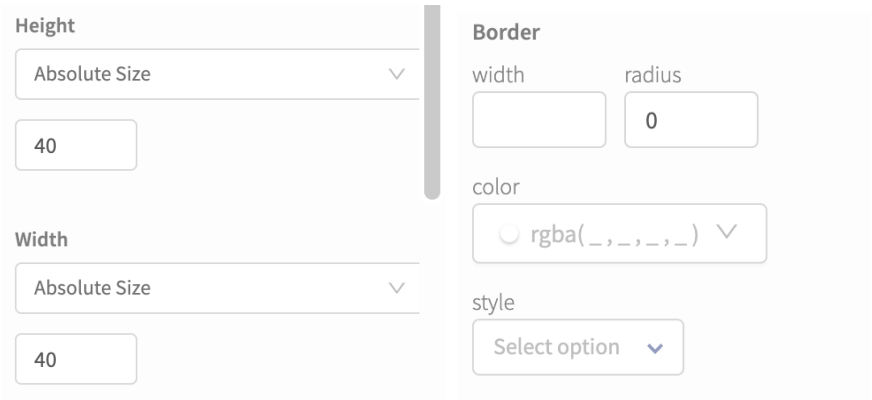


3. Rename the button. I renamed mine to Hamburger Button.

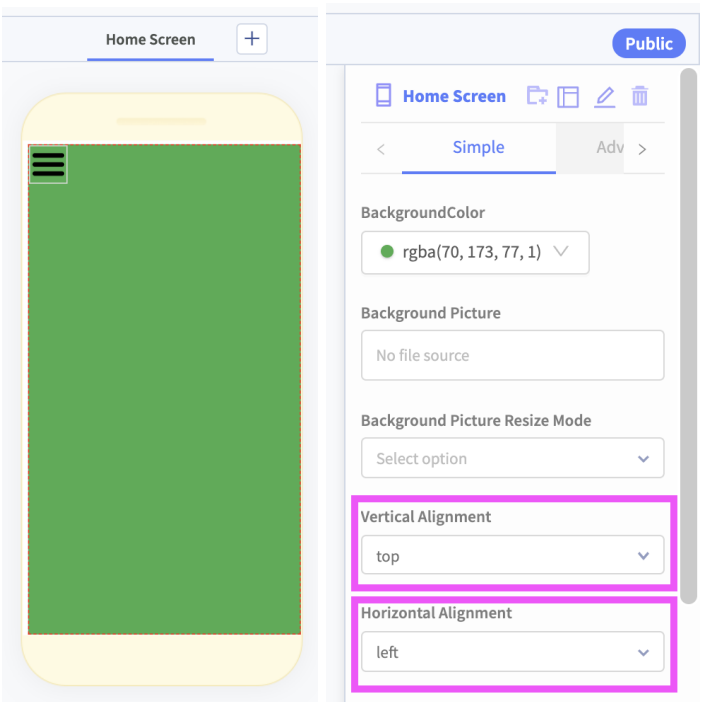
4. Change the background color to transparent, upload the hamburger icon asset, and set background picture resize mode to contain.



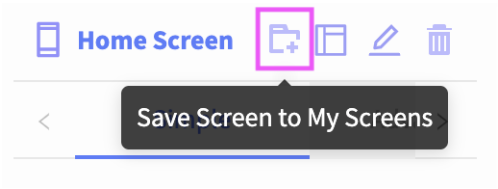
5. Next, set height and width to absolute size = 40, border radius = 0, and 5 px for the top and left margins of the Hamburger button.



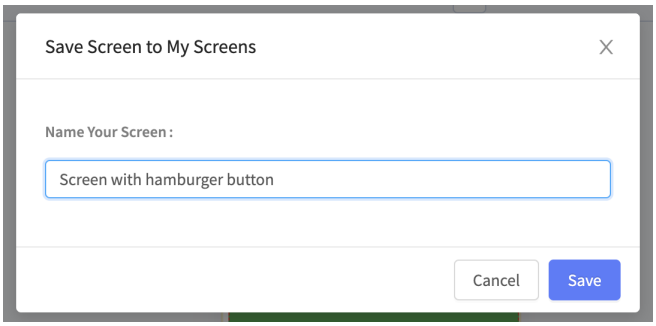
6. You need to play around with the Home Screen properties to move the button to the top left. Select the Home Screen component. Set vertical alignment to top and horizontal alignment to left.



7. Click on the Save Screen button. This will allow you to save the Home Screen so you won't have to keep adding the hamburger buttons in the other screens of your app.



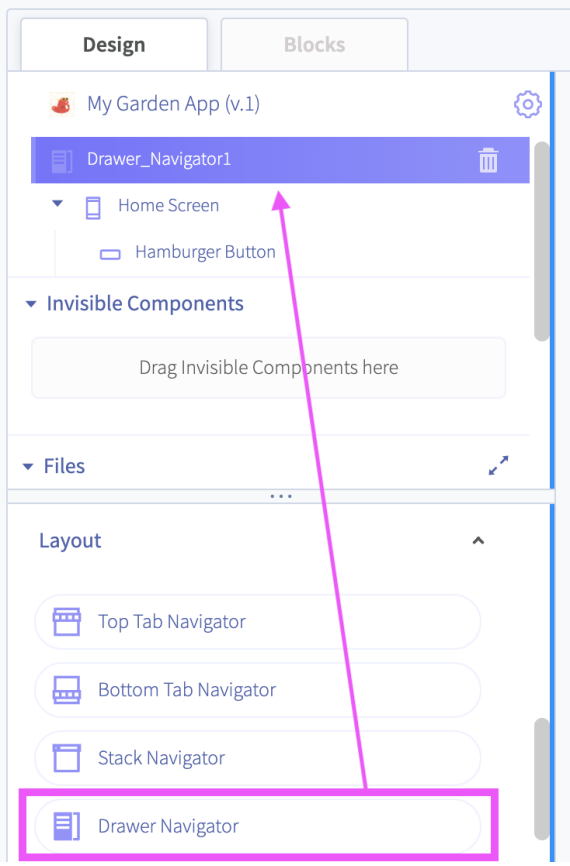
8. I saved the Home Screen as Screen with hamburger button.



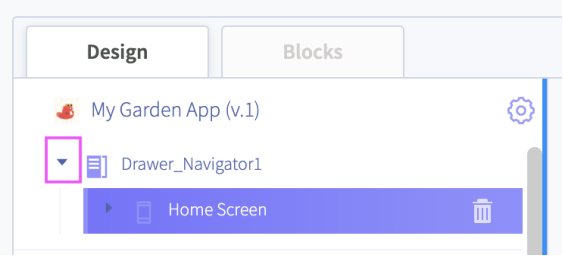
Step: Add drawer navigator

The drawer navigator allows users to navigate to different screens. When users click on the hamburger button, a clickable drawer will slide out.

1. Locate the Drawer Navigator component under the Add Components section.
2. Drag and drop this to the project component section on the top.



3. Now drag the Home Screen under the Drawer Navigator component. You will see an arrow on the left of the Drawer Navigator component.



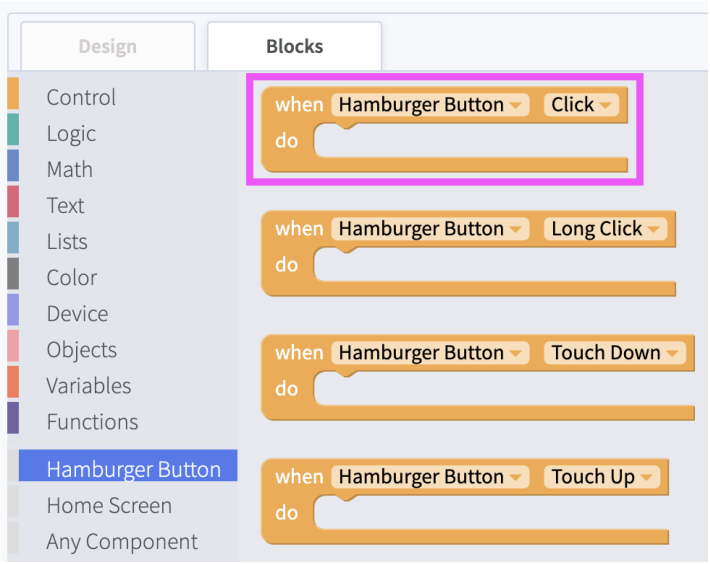
Live Test

Do a quick live test and try clicking on the Hamburger Button. You'll notice that it doesn't work yet so let's work on this.

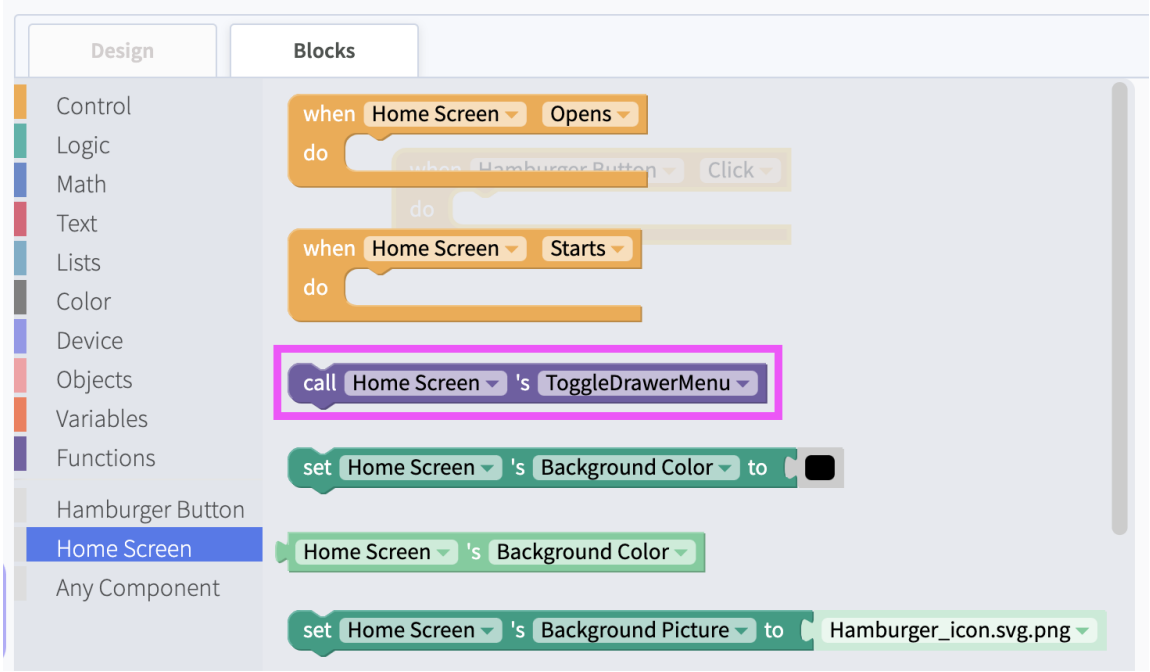
Step: Make the drawer navigator slide out when the Hamburger Button is clicked

Now that you have your hamburger button and drawer navigator. Let's add some code to make the drawer navigator slide out when the Hamburger Button is clicked. To do this, we need to switch to the blocks tab.

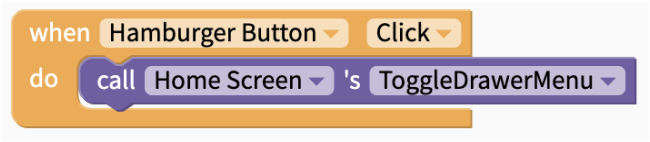
1. Find the Hamburger Button on the left. This is why it's important to rename your components. If you left it as the default Button1 and added multiple buttons without renaming, you'll have issues identifying which button is which.
2. Choose the When Hamburger Button Click block and drag it to the workspace.



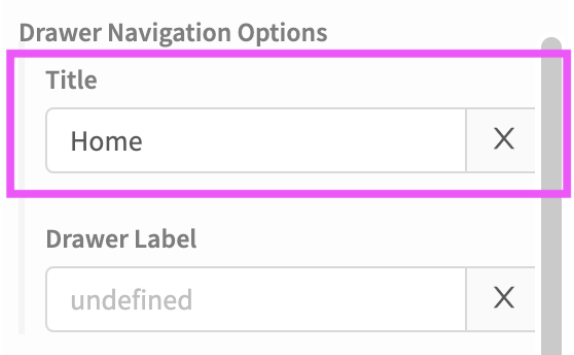
3. Click on the Home Screen component and choose the call Home Screen's Toggle Drawer Menu block.



4. Snap this in the When Hamburger Button Click block.



5. Switch back to the design tab. In the properties section on your right, find the Drawer Navigation Options and change the title to Home.



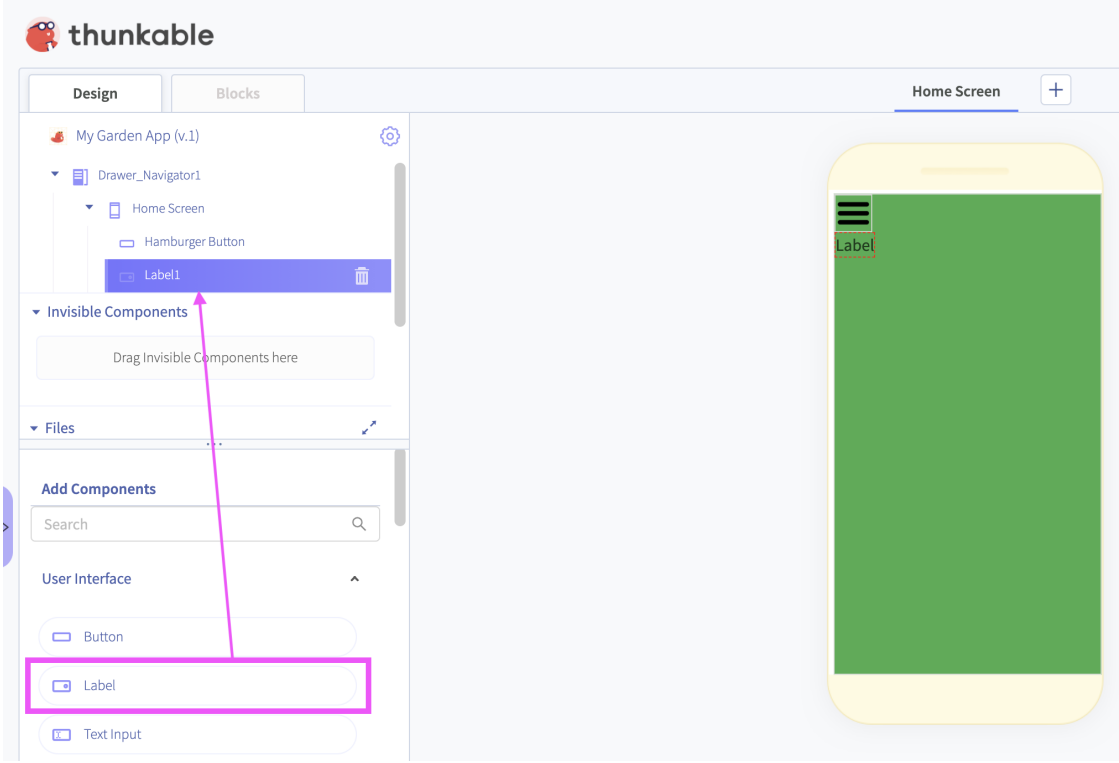
Live Test

Click on the Hamburger Button. The drawer navigator should slide out and you should see the Home option. We will be using this to navigate to different screens of our app.

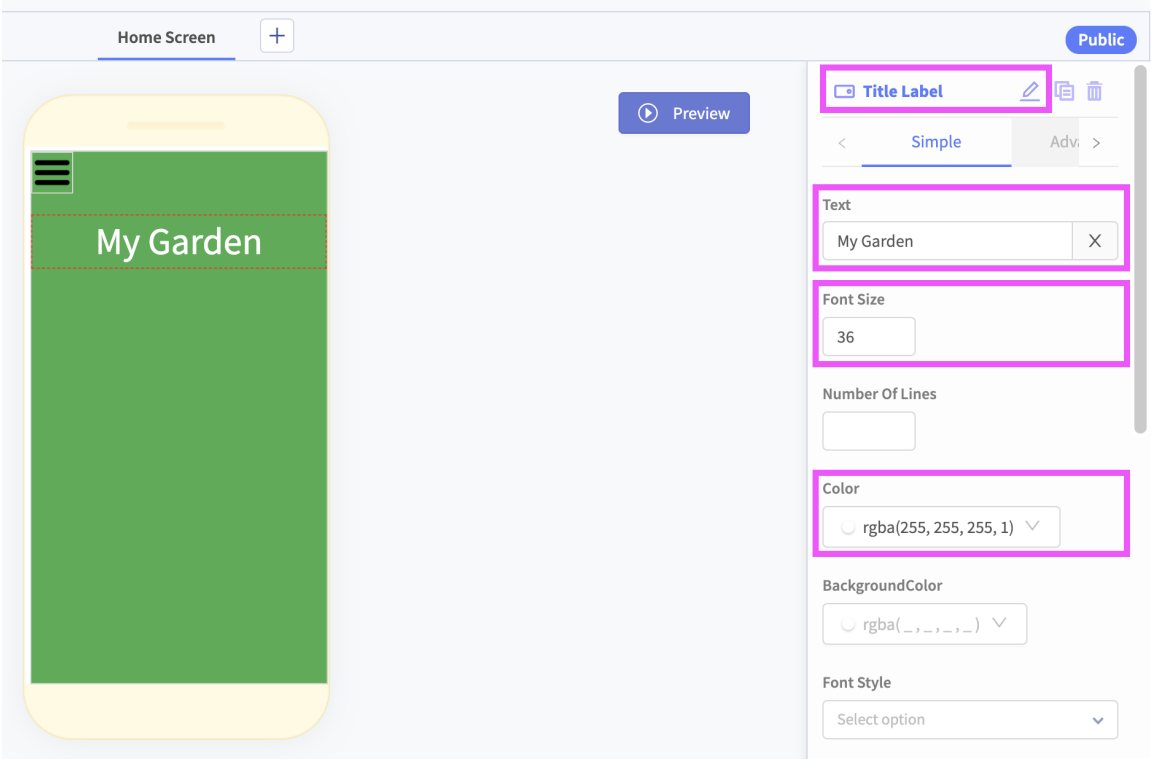
Step: Add label to Home Screen

Let's add the app title to our Home Screen.

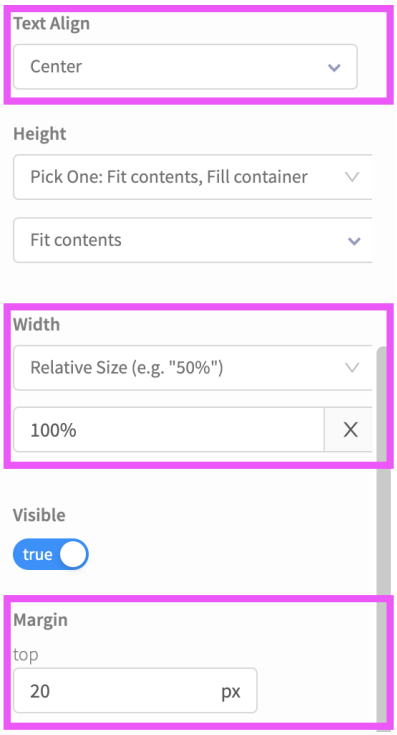
1. Locate the Label component under the Add Components section.
2. Drag and drop this to the project component section on the top.



3. Rename the label. I renamed mine to Title Label. I also changed the text to My Garden, increased the font size to 36, and changed the text colour to white.



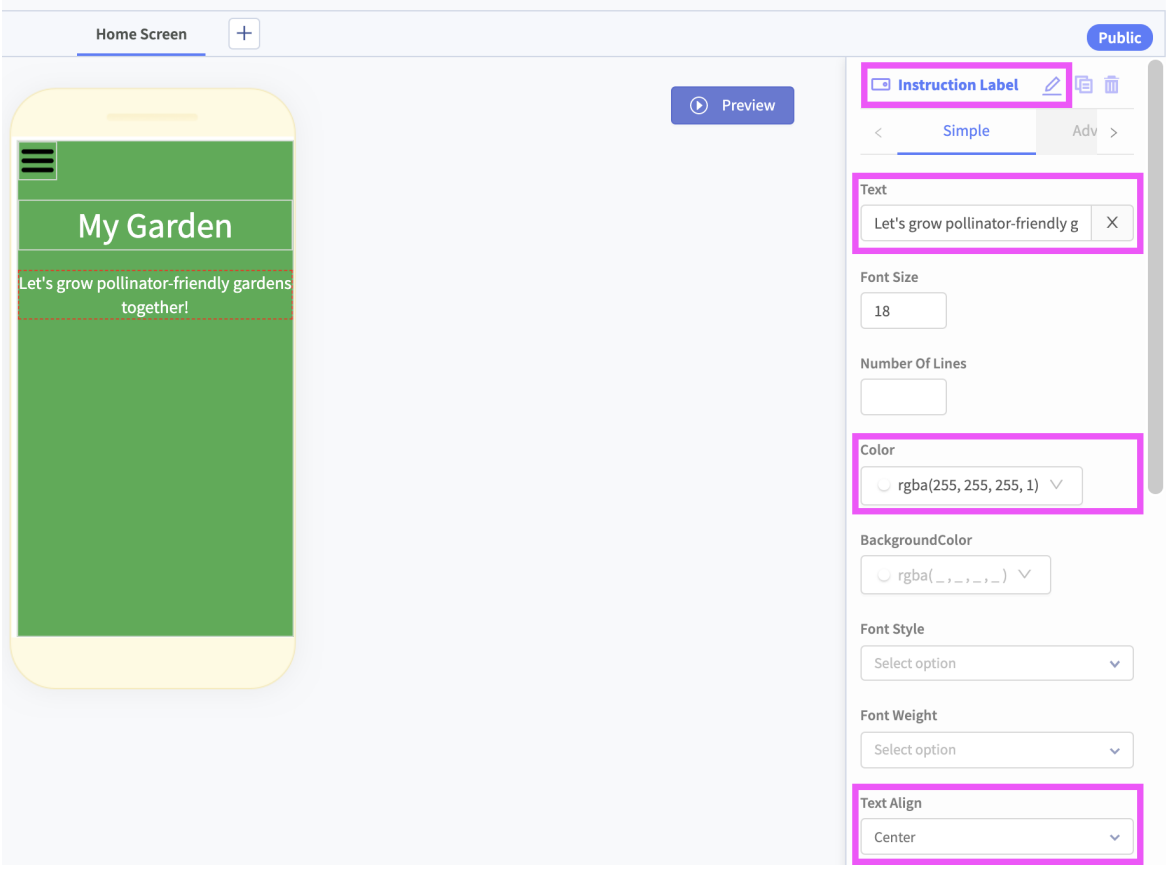
4. I also set text align to center, width to relative size = 100%, and top margin = 20 px.



Step: Add Instruction Label

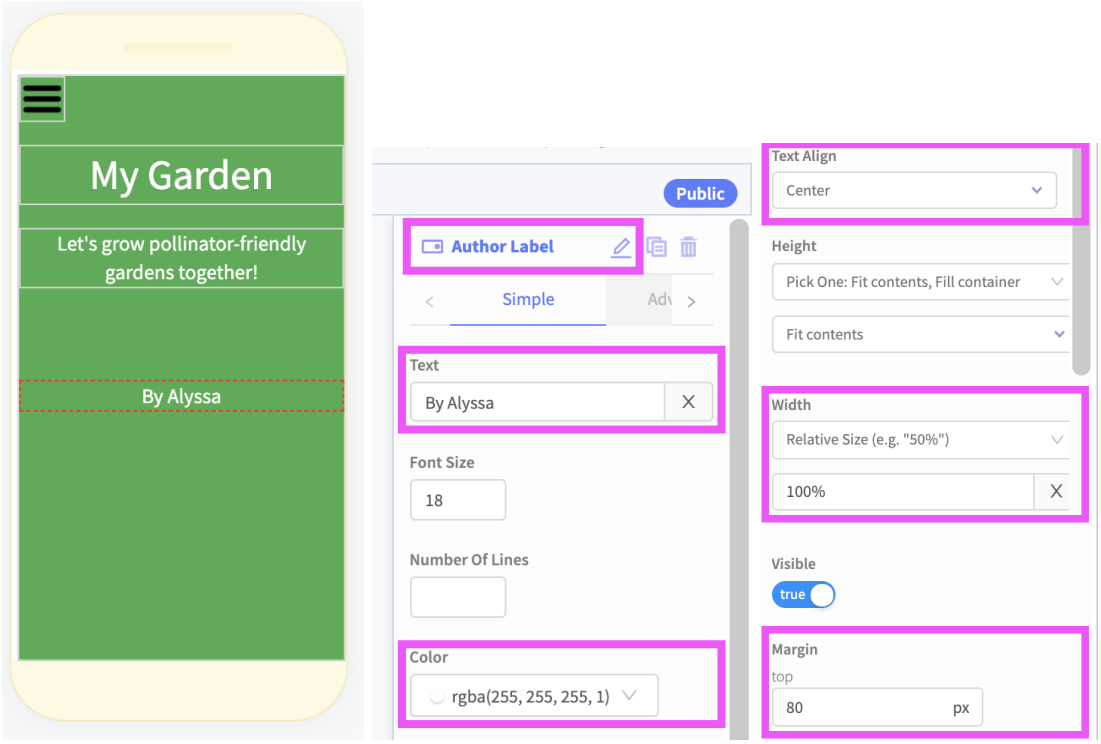
We want some text in the Home Screen to let users know what the app is for.

1. Add a label below the Title Label.
2. Rename it to Instruction Label.
3. Change the text to “Let's grow pollinator-friendly gardens together!”
4. Customize the label if you want. Here, I changed the text colour to white, added some space between the labels (top margin 20 px), and set text align to center.



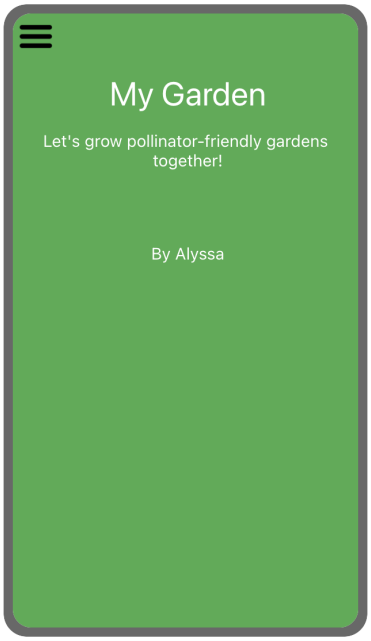
Step: Add Author Label

1. Add a label below the Instruction Label.
2. Rename it to Author Label.
3. Change the text to your name.
4. Customize the label if you want. Here I changed the text colour to white and added some space between the labels (top margin 80 px).



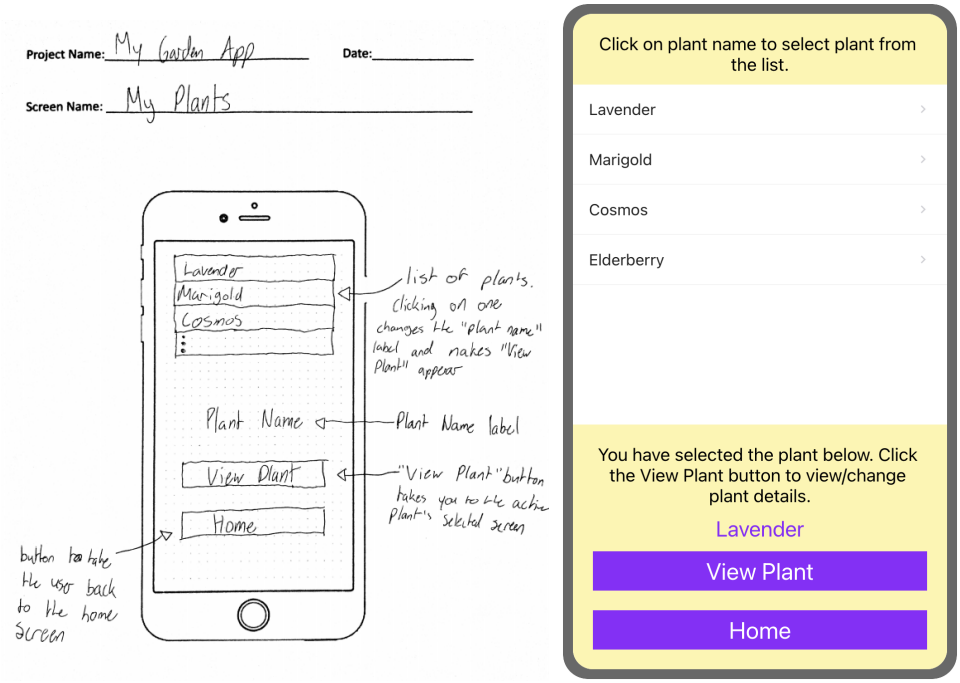
Live Test

Live test your app. You should see all your labels on the Home Screen. Now let's add more screens to our app.



My Plants Screen

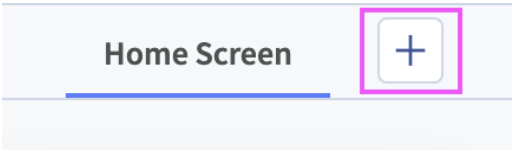
My Plants Screen Wireframe



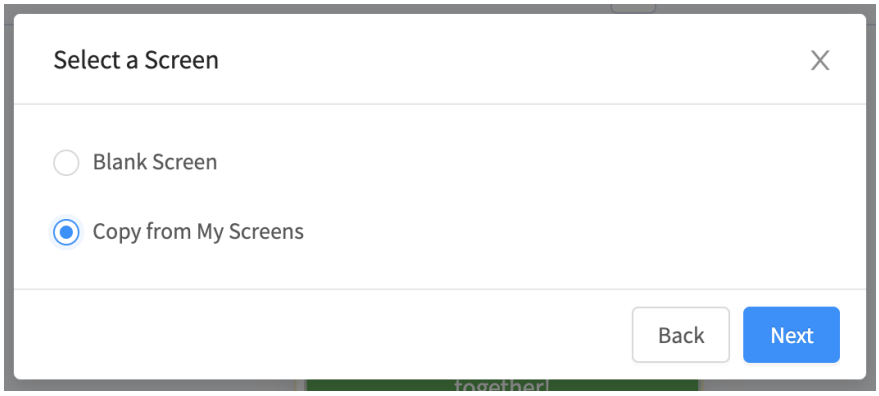
Step: Add My Plants Screen

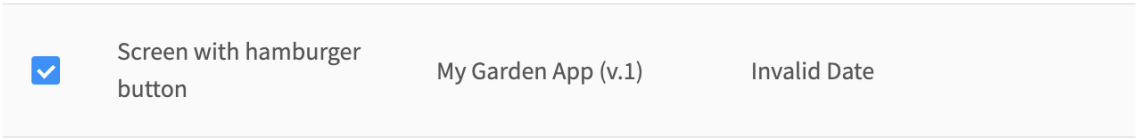
Now that you have a Home Screen and a working drawer navigator, let's work on the My Plants Screen.

1. To add a second screen, click on the plus button.

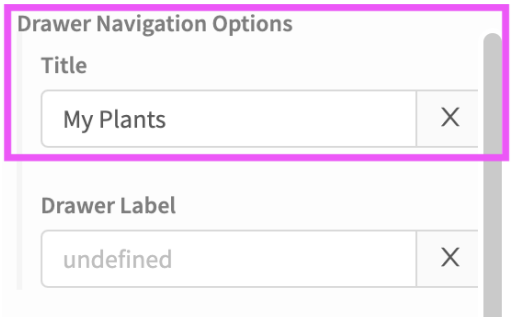


2. Because you saved the Home Screen, you should see the option to Copy from My Screens. Choose this option and then select the Screen with hamburger button.



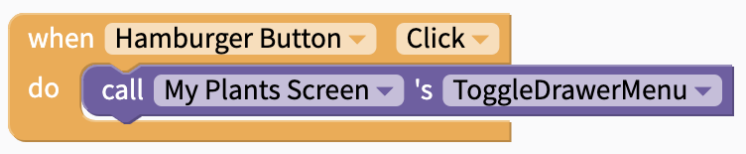


3. Rename the screen to My Plants Screen.
4. Drag the My Plants Screen under the Drawer Navigator.
5. In the properties section on your right, find the Drawer Navigation Options and change the title to My Plants.



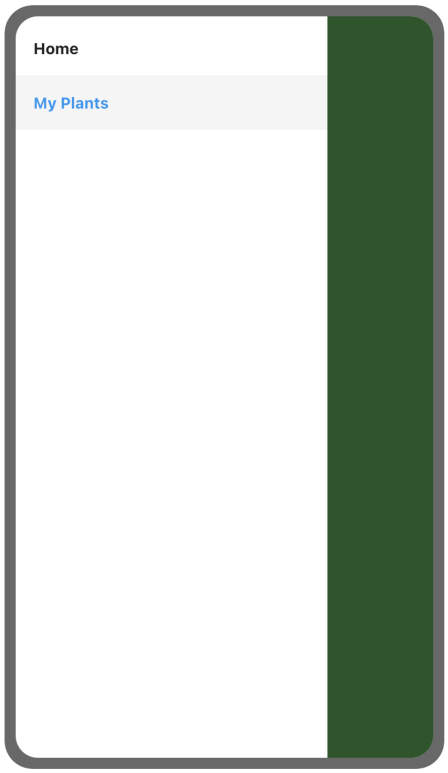
Step: Add My Plants Screen to the drawer navigator

1. Switch to the blocks tab.
2. Click on the Hamburger Button component and choose the When Hamburger Button Click block. Drag this block to your workspace.
3. Now find the call My Plants Screen's Toggle Drawer Menu block under the My Plants Screen component and snap that in the When Hamburger Button Click block.



Live Test

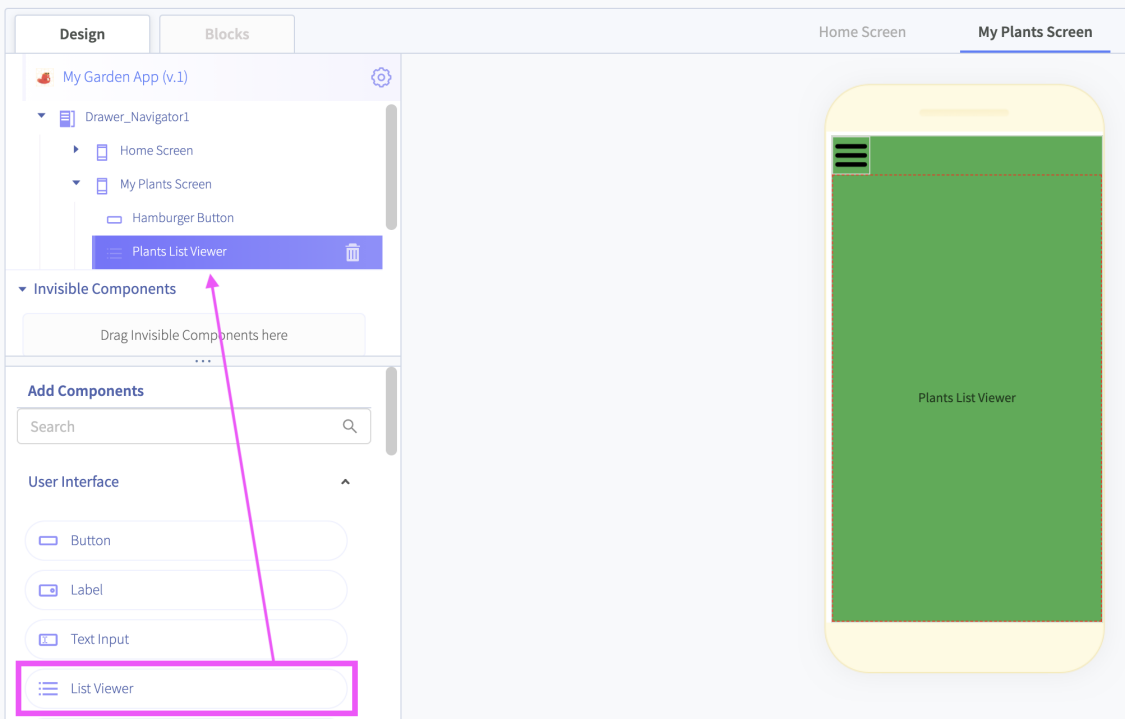
Click on the Hamburger Button. You should see the Home and My Plants options in the drawer navigator. Try navigating between both screens.



Add list of plants to My Plants Screen

Step: Add list component to My Plants Screen

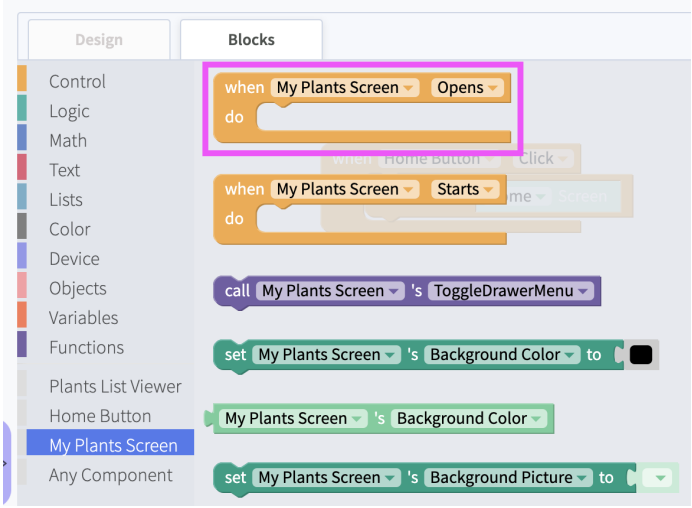
1. Add a list viewer component to the My Plants Screen.
2. Rename it to Plants List Viewer.
3. Customize the list viewer if you want. I changed the Text Items Background Color to white.



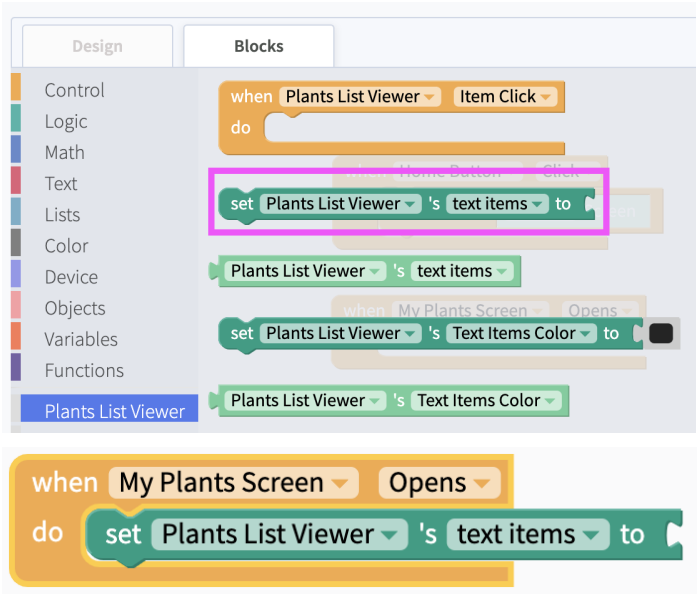
Step: Add plants to list component using code

We'll do this using code so let's switch to the blocks tab.

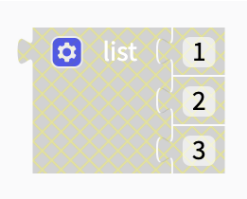
1. Click on the My Plants Screen component and choose the when My Plants Screen Opens block.
2. Drag this block to your workspace.



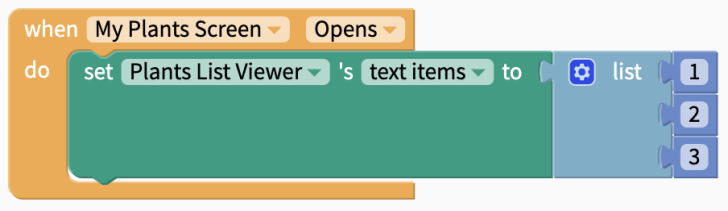
3. Click on the Plants List Viewer component and choose the set Plants List Viewer's text items block. Snap this into the when My Plants Screen Opens block.



4. Click on the Lists categories and select the list with numbers block.

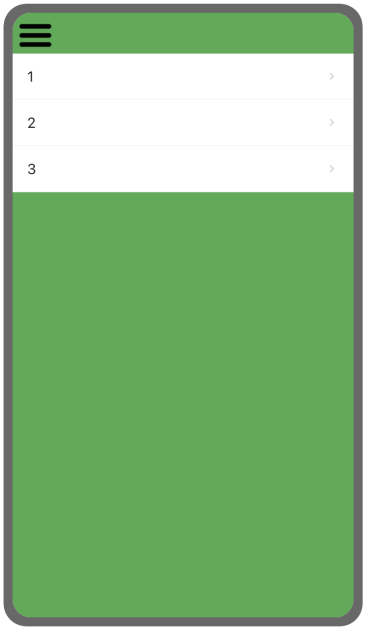


5. Snap that into the to section of the set Plants List Viewer's text items block.



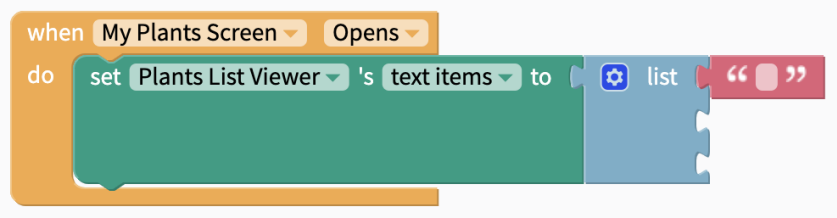
Live Test

When you live test your app, you should see 1, 2, 3 in your list viewer component. Let's change these into plant names.

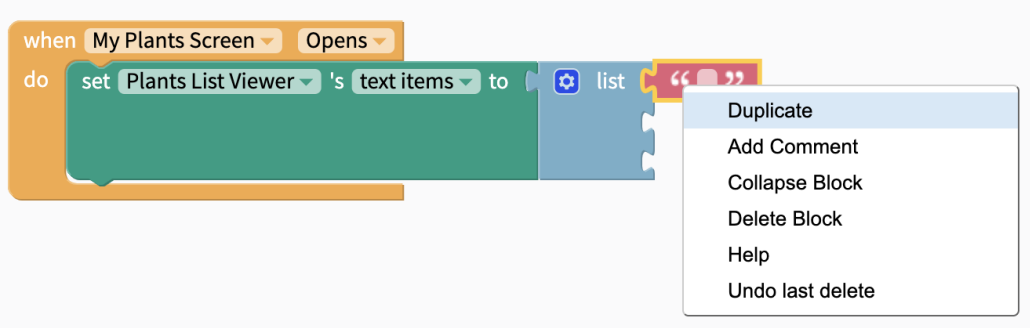


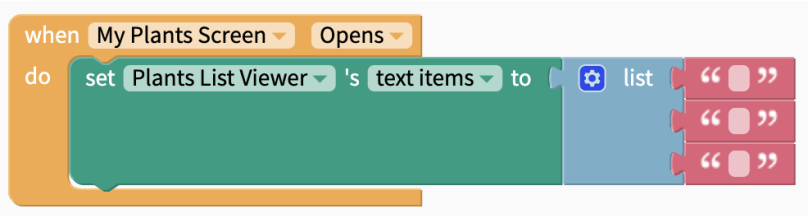
Step: Add plants to list component using code (continued)

6. Delete the number blocks from the list.
7. Go to the Text category and choose the empty text block. You want to snap this into your list.

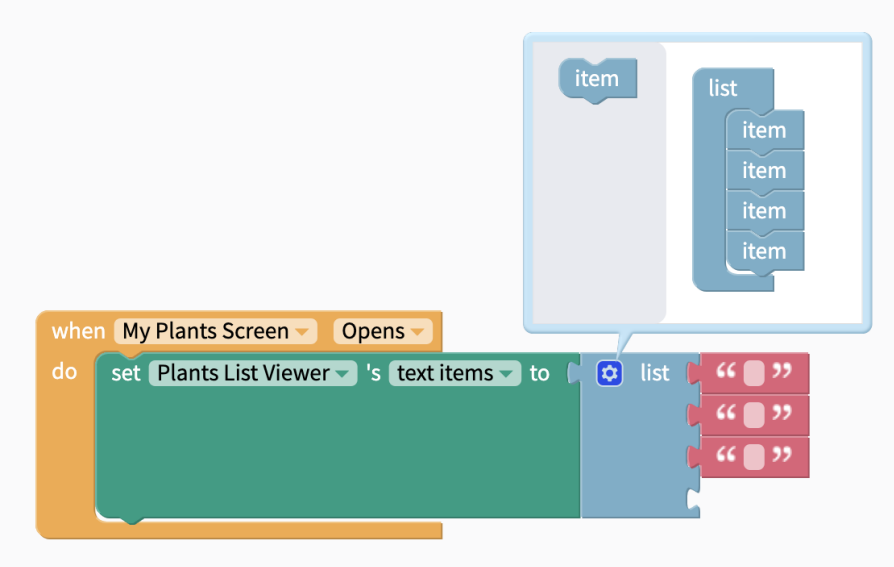


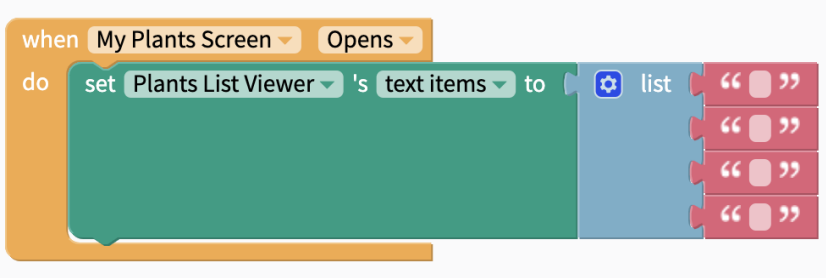
8. You can right click the empty text block and Duplicate this block to fill your list.



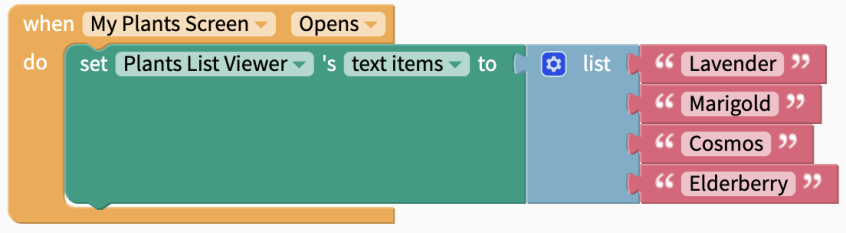


9. The default items in a list is 3. If you want to add an extra item, click on the cogwheel icon. Drag an extra item into the list and an extra space will appear. You need to click on the cogwheel again to close this option.



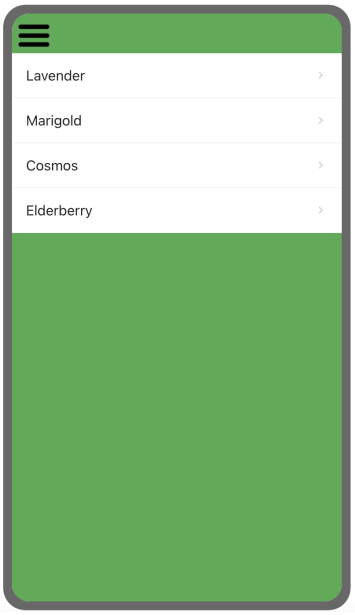


10. Enter plant names into the empty text blocks. You could enter plants you see around your area or at a community garden. I'm going to enter lavender, marigold, cosmos, and elderberry.



Live Test

You should see the plant names you entered in the list viewer component.



Show selected plant when item in the list viewer is clicked

We want users to click on a plant name in the list viewer and have this selected plant name appear at the bottom of the screen to show their selection.

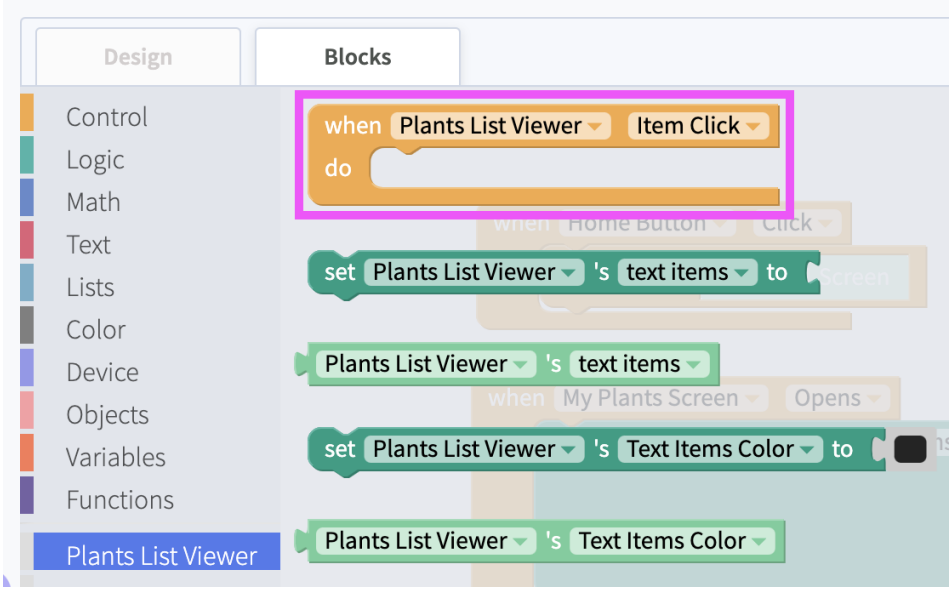
Step: Add Plant Name Label

1. Add a label to the project components and rename it Plant Name Label.
2. Customize how you want this label to look (e.g. text colour, margins etc.).

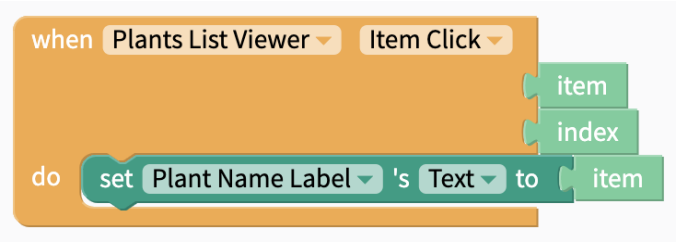
Step: Change label text to the selected plant name

Let's switch to the blocks tab for this.

1. Click on the Plants List Viewer component and choose the when Plants List Viewer Item Click block. Drag this into your workspace.

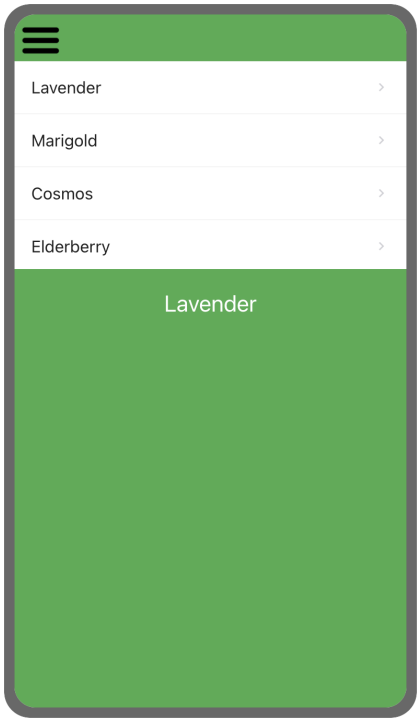


2. Find the set Plant Name Label's Text block under the Plant Name Label component. Snap this in the do section of the block. You'll have to duplicate the item and add that.



Live Test

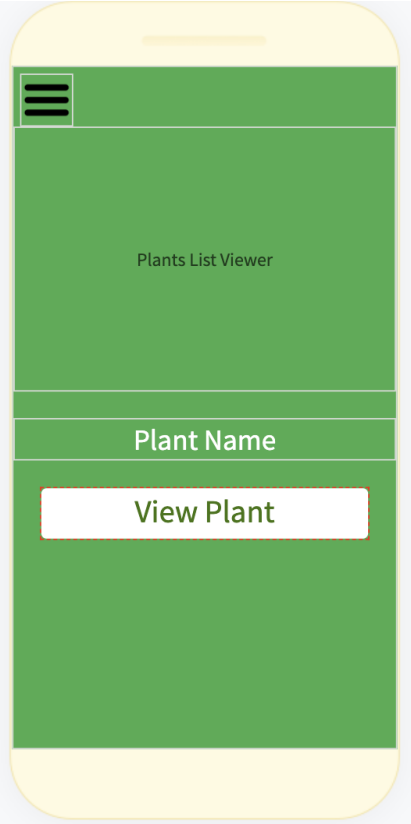
When you click on the plant names in the list viewer, your label should display your selection. In the screenshot below, Lavender is selected.



Step: Add View Plant Button

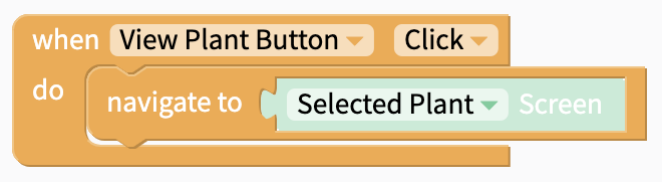
We want users to be able to rate the plant they selected based on how good the plant is for local pollinators. For this to happen, we'll add a View Plant Button which will navigate to the Selected Plant Screen. Users will be able to rate the selected plant on this screen, which we will work on later on.

1. Add a button below the Plant Name Label. Customize it however you want.

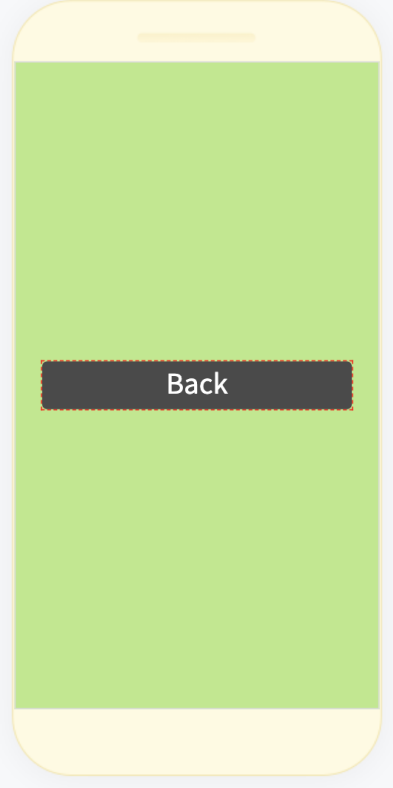


2. Since the View Plant Button navigates to the Selected Plant Screen, let's add the Selected Plant Screen. Rename the Selected Plant Screen and customize the screen if you want. *Do not copy this screen from the saved screens and do not drag this screen under the drawer navigator.*
3. Go back to the My Plants Screen and switch to the blocks tab.

4. Add code so that the View Plant Button will navigate to the Selected Plant Screen when it is clicked.



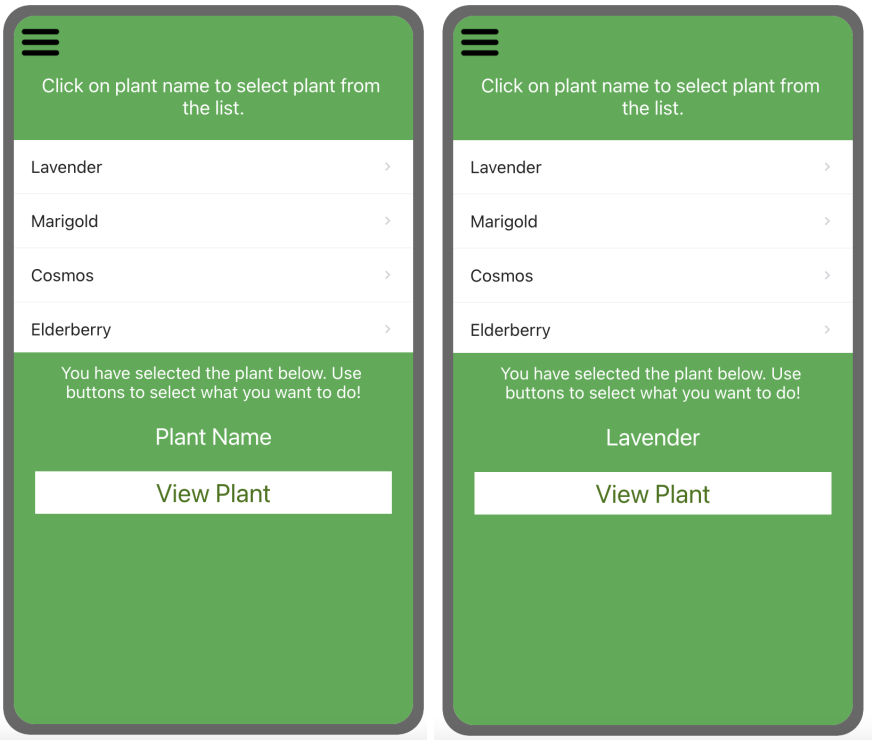
5. Go to the Selected Plant Screen and add a Back Button that will navigate back to the My Plants Screen when clicked.



Step: Add instructions

1. Let's add some instructions to our My Plants Screen.
2. I added a label that says "Click on plant name to select plant from the list." I renamed this the Select Instruction Label.
3. I also added a label that says "You have selected the plant below. Use buttons to select what you want to do!" I renamed this to the View Plant Instruction Label.

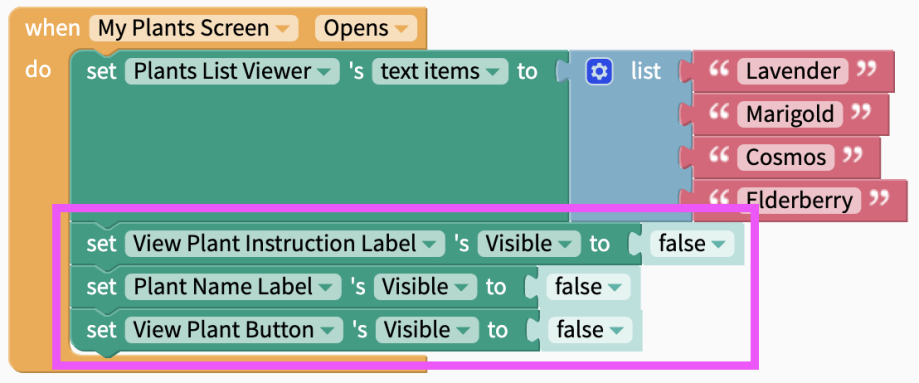
4. You can customize these labels however you want.



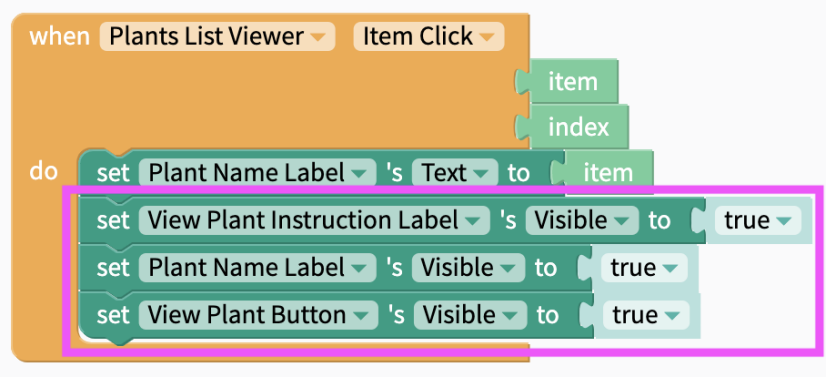
Step: Hide label and button when My Plants Screen open

To avoid confusing users, we want to hide the View Plant Instruction Label, Plant Name Label, and View Plant Button. We want this to show only after an item is clicked. Let's switch to the blocks tab for this.

1. Hide the View Plant Instruction Label, Plant Name Label, and View Plant Button. You can find these set Component's Visible to true/false blocks under the respective components.

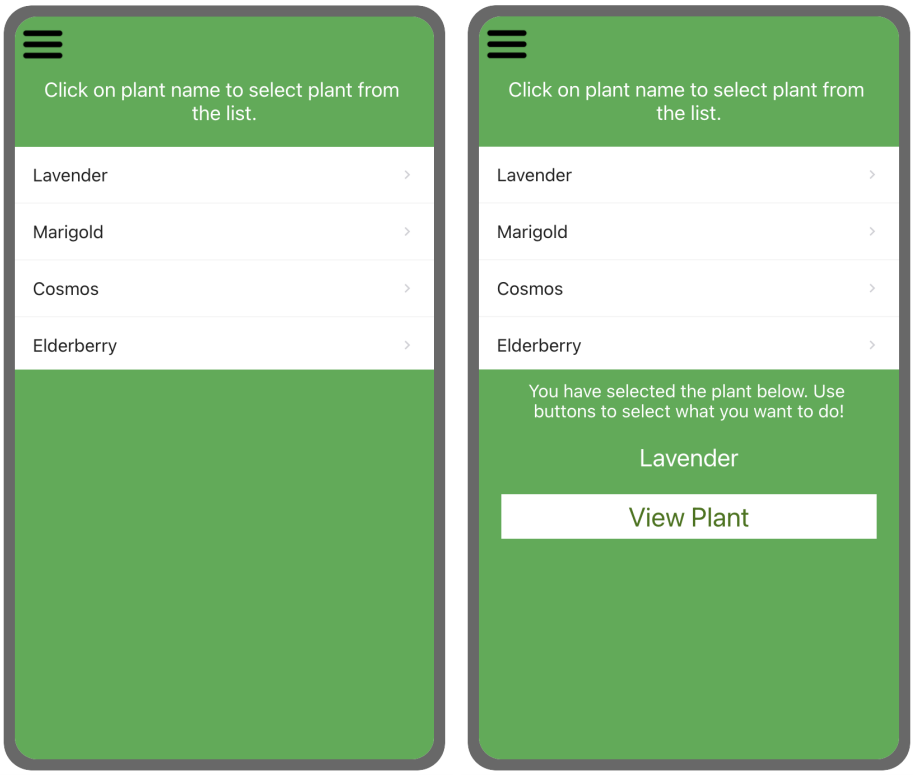


2. Now you want the hidden components to show when the item in the list viewer is clicked. Duplicate the set Component's Visible to true/false blocks above, set them to true, and snap them under the do section of the when Plants List Viewer Item Click block.



Live Test

When the My Plants Screen opens, you should not see your hidden components. These components will appear when you click on an item. In the screenshot below, I clicked on Lavender.

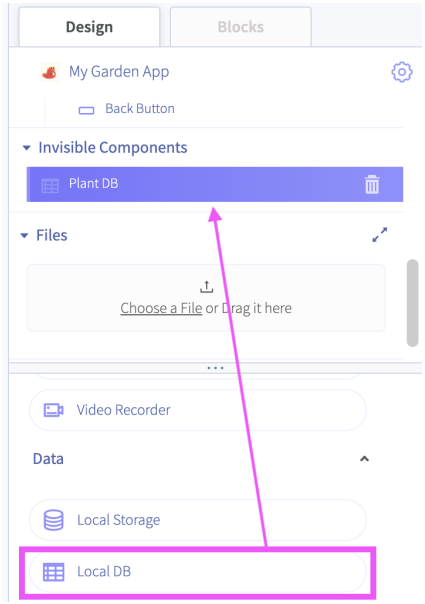


Use a local database and variables

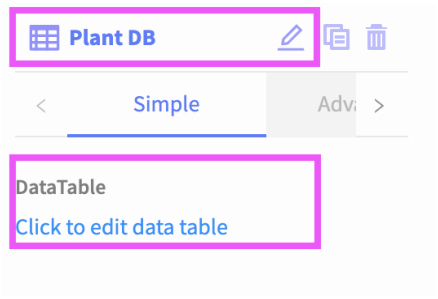
We used a list of strings in the My Plants Screen for our plant names. For new screens in our app, we also want to show these plant names. Because we don't want to duplicate this list, we'll use a local database (DB).

Step: Add local DB to project components

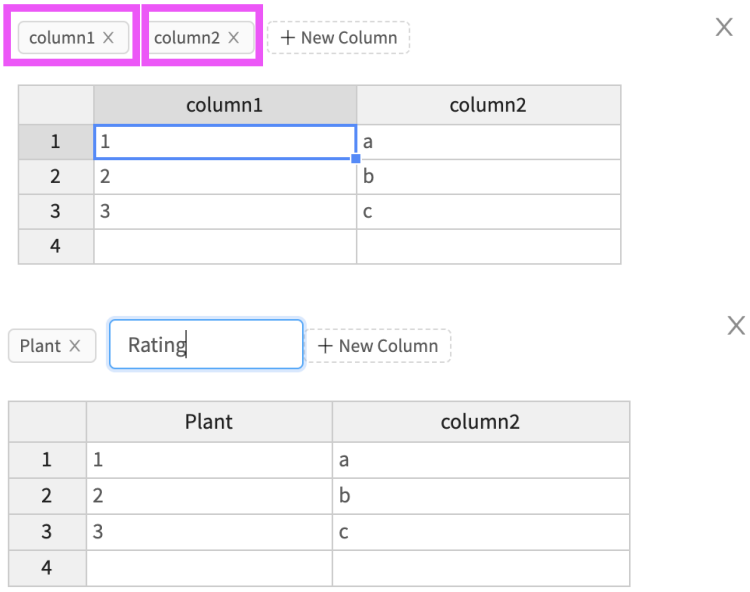
1. Add the Local DB to your project components.



2. Rename this to Plant DB. Click on the "Click to edit the data table" text to edit Plant DB.



3. Click on column1 to rename the column to Plant. Because we'll also be saving the ratings of these plants, click on column2 and rename the column to Rating.



	column1	column2
1	1	a
2	2	b
3	3	c
4		

	Plant	column2
1	1	a
2	2	b
3	3	c
4		

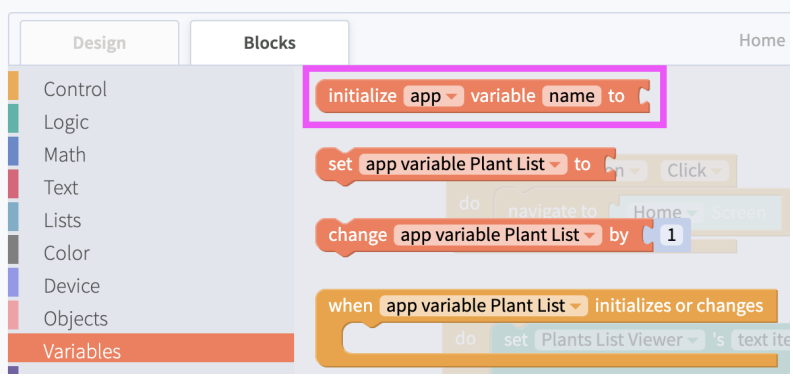
4. Fill in the rows of the Plant column with plant names and enter 0 for the rating. You will always get an extra empty row. Don't worry about this.

	Plant	Rating
1	Lavender	0
2	Marigold	0
3	Cosmos	0
4	Elderberry	0
5		

Step: Load Plant List variable from the Plant DB

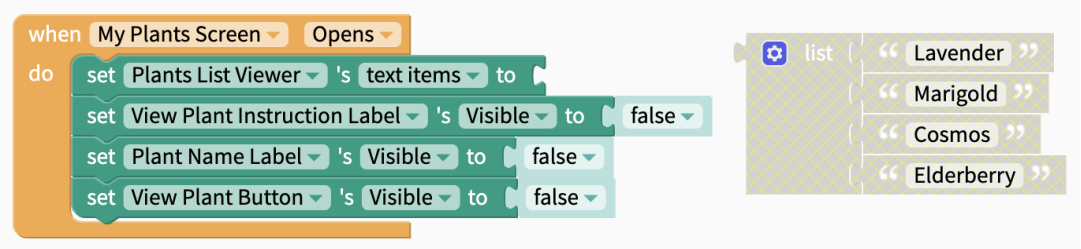
By loading the DB into a list, we can avoid duplicating lists which will reduce errors. In version 2 of the app, we'll add an Add Plant Screen which will allow users to add plants to the Plant DB.

1. First, we need to initialize the Plant List variable. Click on the Variables category and choose the initialize app variable name to block. Drag this into your workspace.

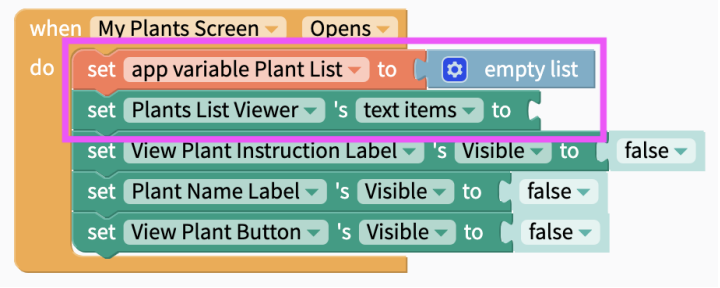


2. Change the name to Plant List and drag over the empty list block.

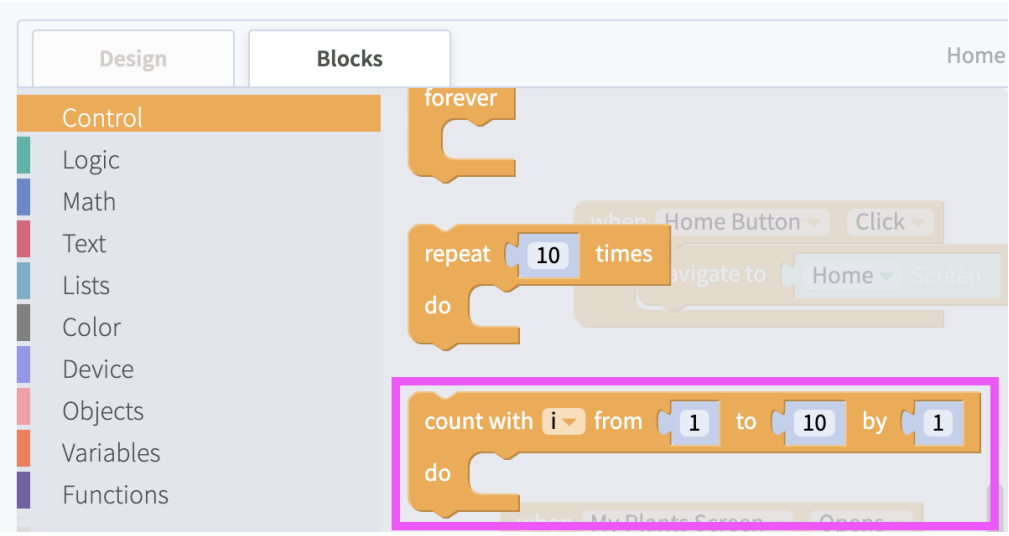
3. Delete the list of plant names from the set Plants List Viewer's text items block.



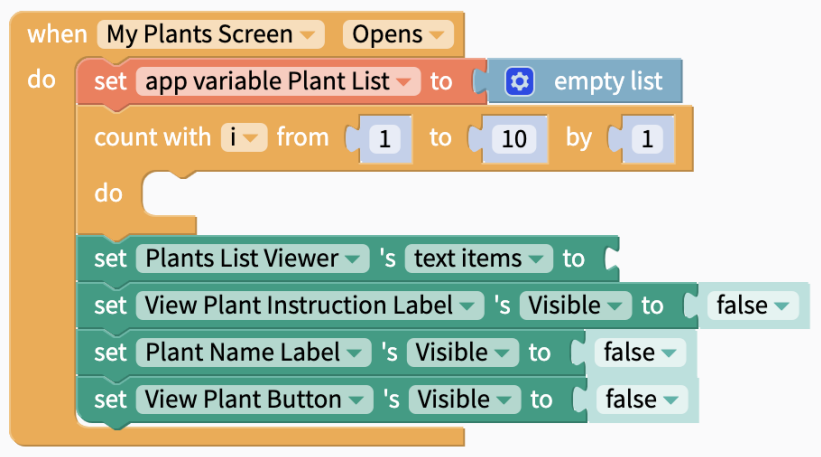
4. Under the Variables category, choose the set app variable Plant List block and snap it under the do section. Add an empty list block to it.



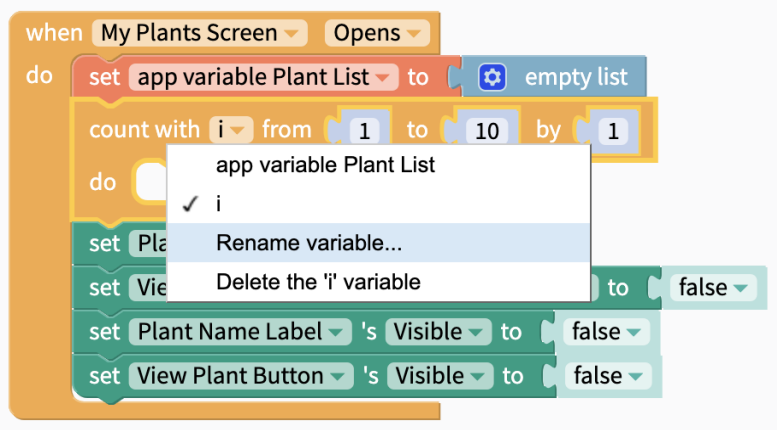
5. Click on the Control category and find the count-with-do block. This is a loop that will read through each row in the DB and load it into the Plant List variable.

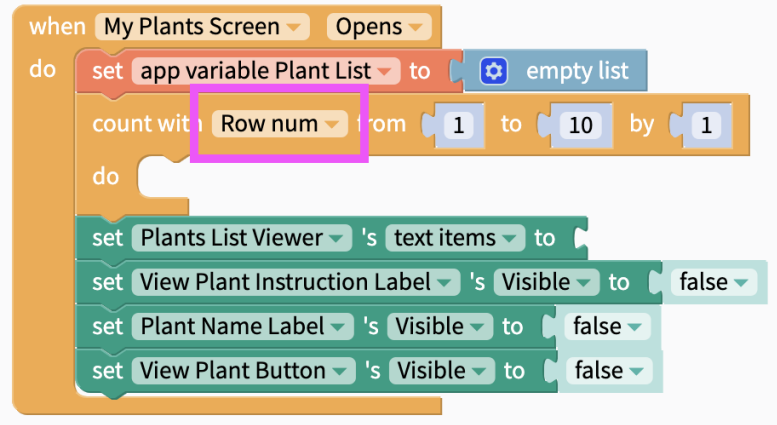


6. Snap this count-with-do block between the set app variable Plant List block and the set Plants List Viewer's text items block.

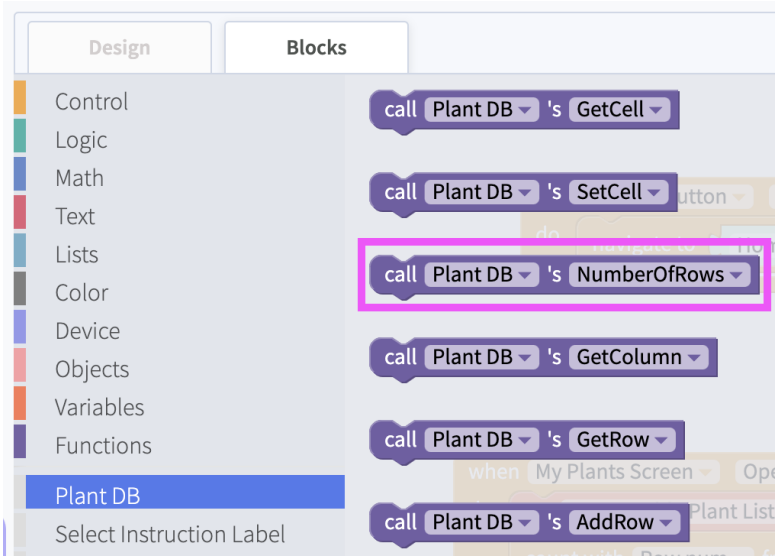


7. There are few things we need to change for this loop to do what we want it to do. Let's start with renaming "i" to something meaningful. Since we want to loop through the rows of the Plant DB, let's change this to Row num.

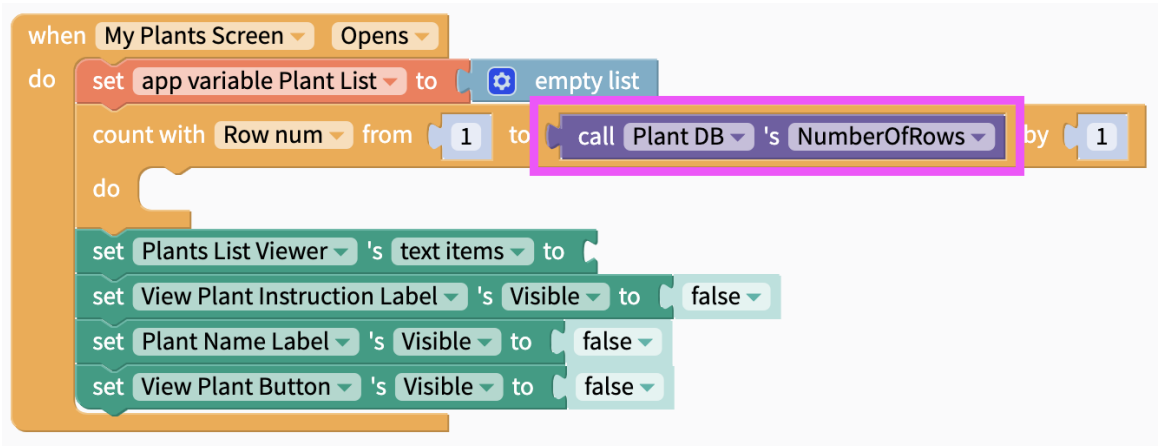




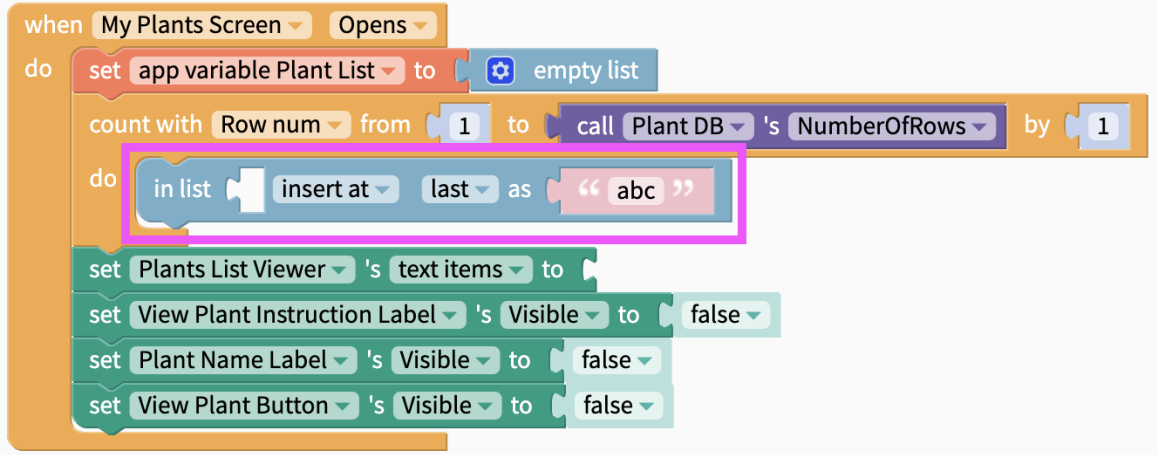
8. Next we want to loop through row 1 to the number of rows in our Plant DB instead of setting it to end after row 10. To do this, click on the Plant DB component and choose the call Plant DB's NumberOfRows block.



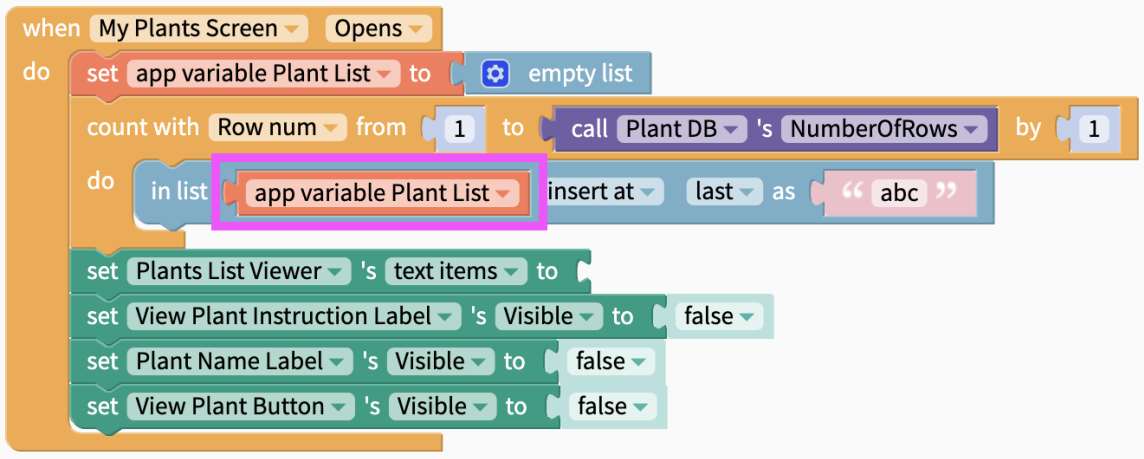
9. Snap this into the count-with-do block. You will note that this is a shape-shifting block.



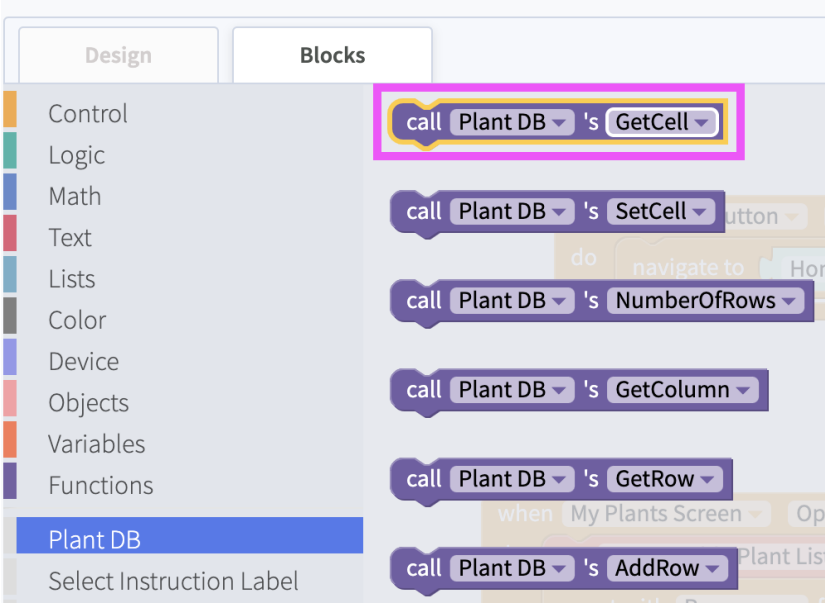
10. Now we want to find the block that will add each row to the end of the Plant List variable. To do this, we need to go to the Lists category and find the in list [variable name] insert at last block. Snap this under the do section of the count-with-do block.



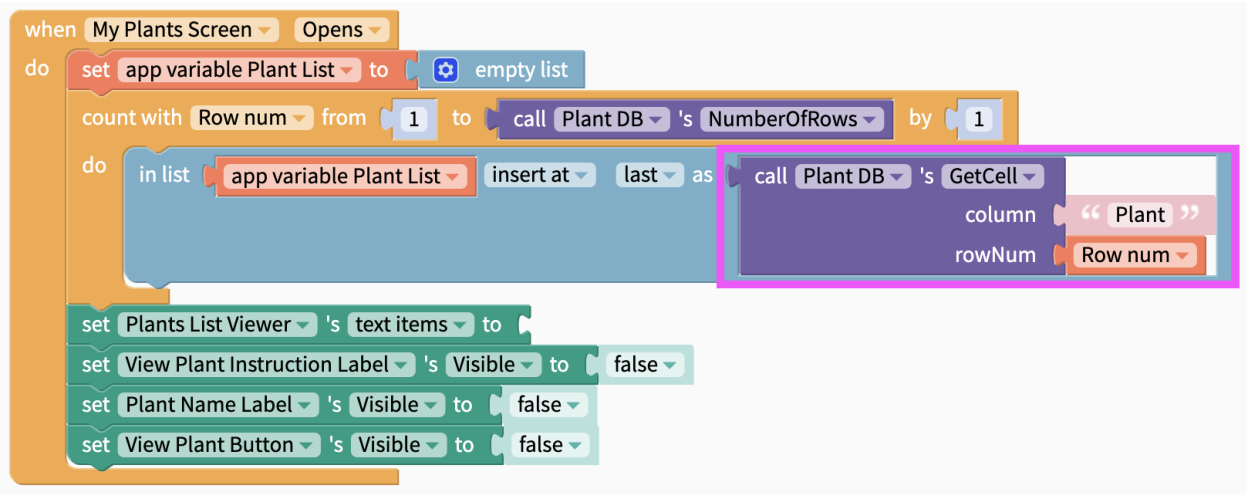
11. Next, add Plant List variable into the in list [variable name] insert at last block. You can find this under the Variables category.



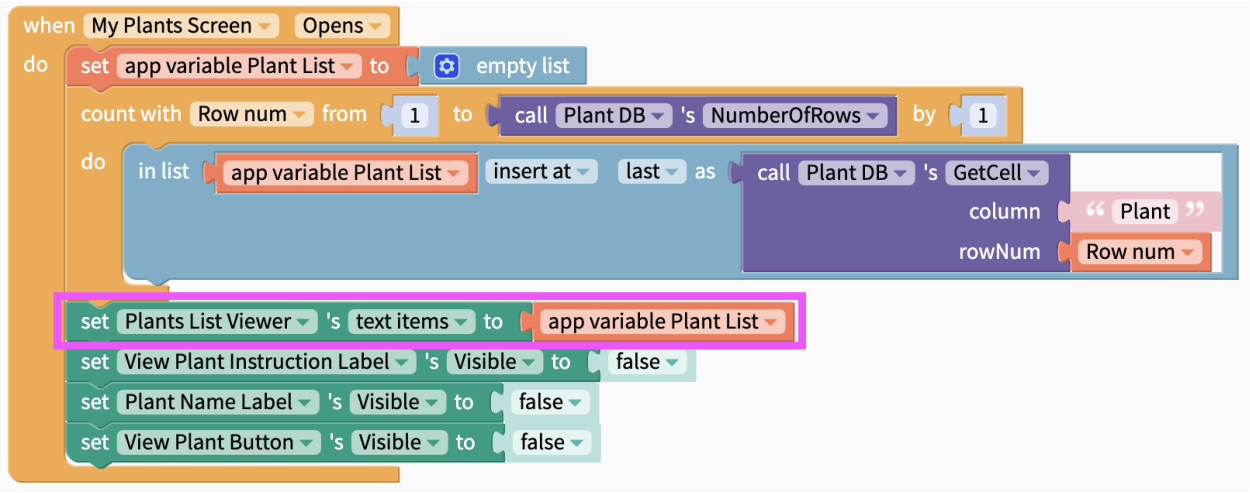
12. To load plant names into the Plant List variable, go to the Plant DB component and choose the call Plant DB's GetCell block.



13. Snap this into the in list [variable name] insert at last block. Note that this is a shape-shifting block. Enter Plant into the column text block and snap the Row num variable into the rowNum section.

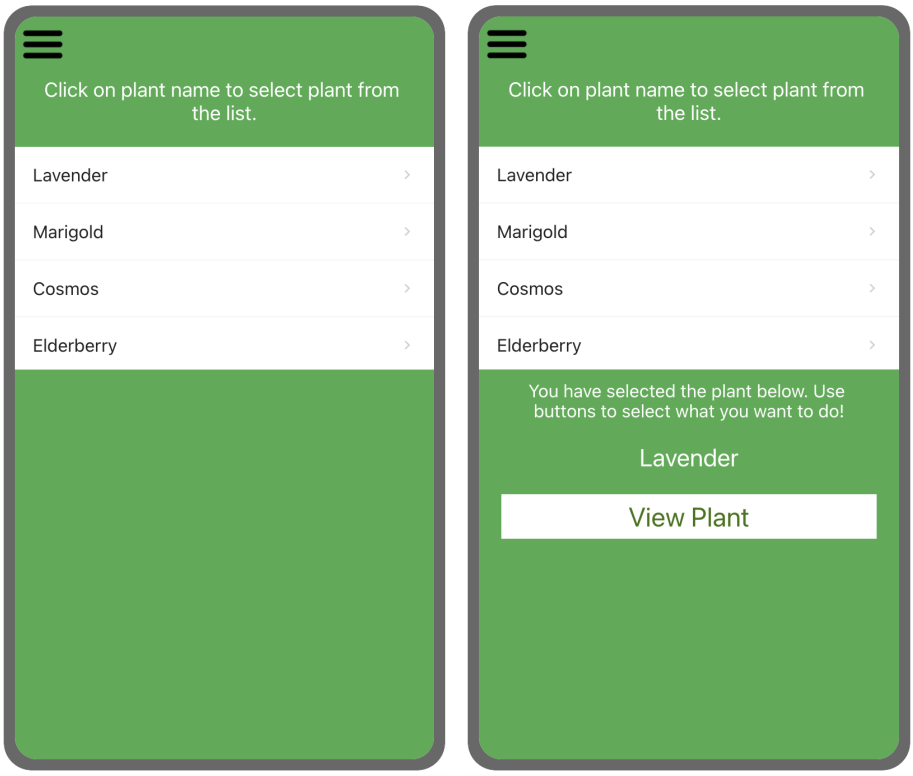


14. Lastly, snap the Plant List variable into the set Plants List Viewer's text items block.



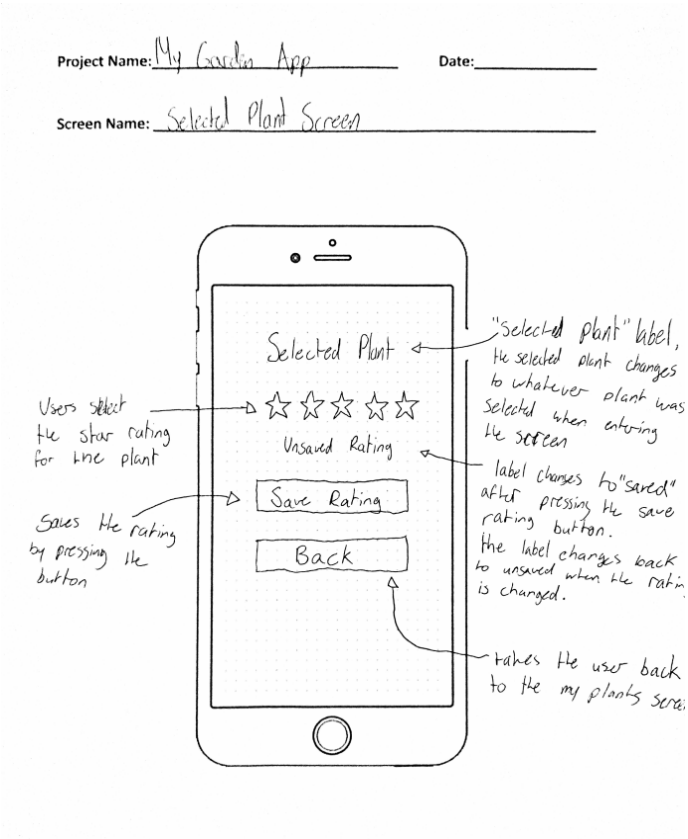
Live Test

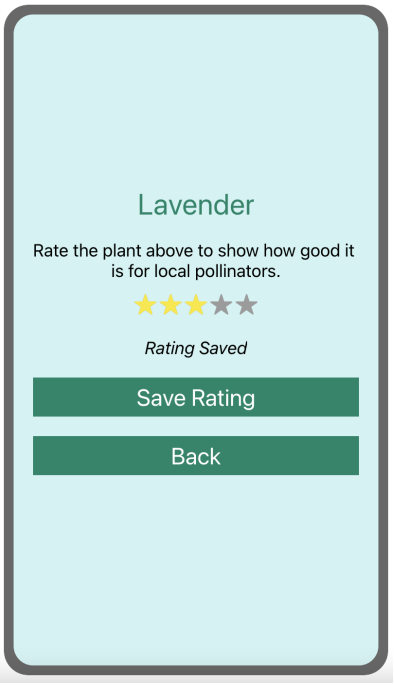
Nothing should change from the previous live test of your app. Except this time the plant names in the list viewer are from the Plant DB you created.



Selected Plant Screen

Selected Plant Wireframe

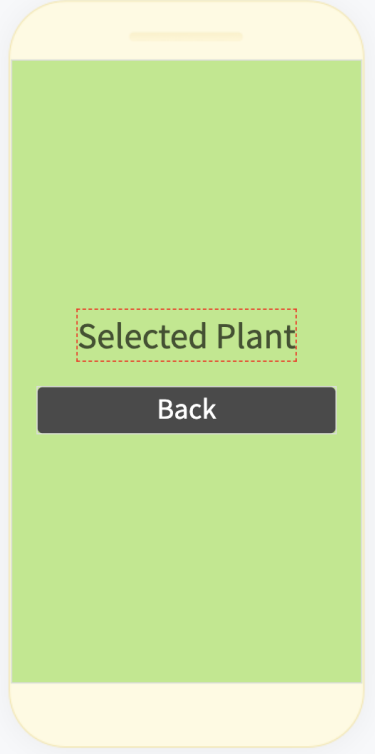




Step: Add Selected Plant Label

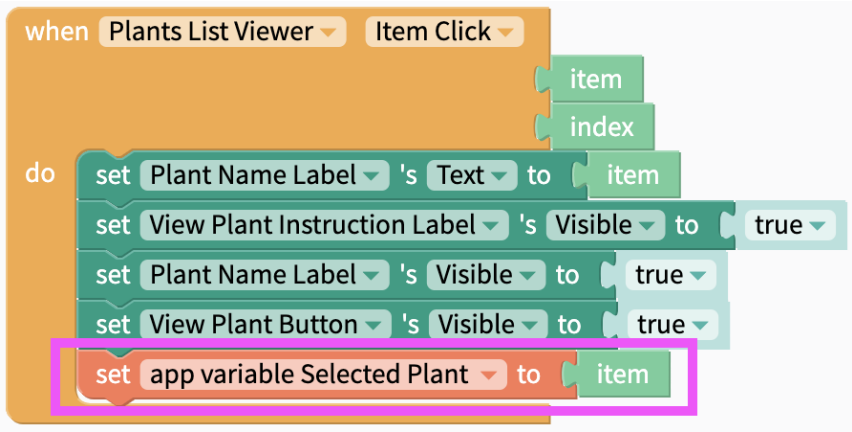
We want the name of the selected plant to display on the Selected Plant Screen, which we will work on in this section.

1. Add a label to the Selected Plant Screen and rename it to Selected Plant Label. Customize the label based on your preference.

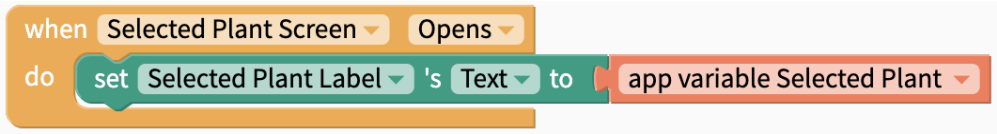


2. Switch to the blocks tab of the My Plants Screen and initialize the Selected Plant variable.

3. Under the when Plants List Viewer Item Click, add the set app variable Selected Plant to item block.



4. Switch back to the Selected Plant Screen. Find the when Selected Plant Screen Opens block under the Selected Plant Screen component and drag this into your workspace. Add the set Selected Plant Label's Text block to the do section and snap the Selected Plant variable to this block.



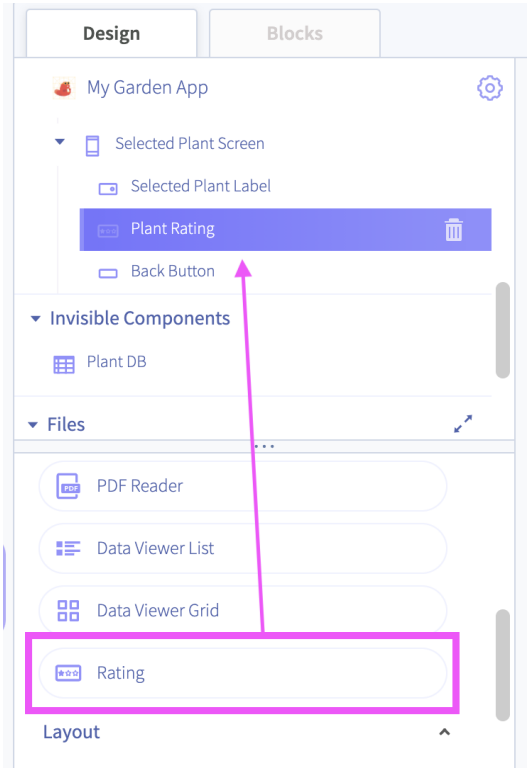
Live Test

Start from the My Plants Screen, click on a plant name on the list viewer and then click on the View Plant button. When you navigate to the Selected Plant Screen, you should see the plant you selected in the My Plants Screen. In this case, I selected Lavender.

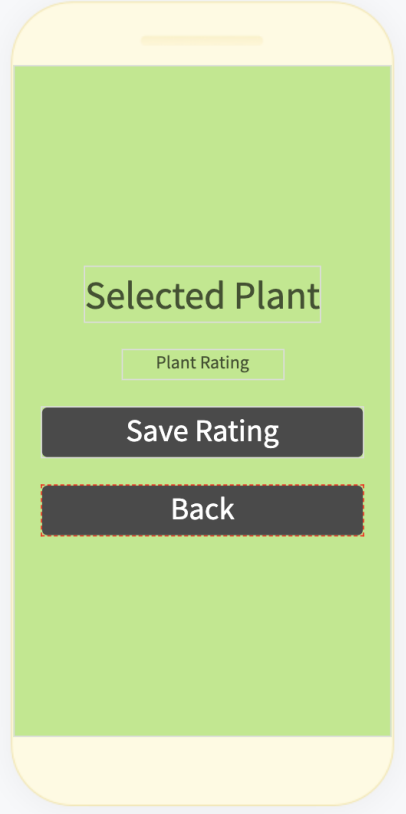
Step: Save plant ratings

We want users to be able to rate how good the plants are for local pollinators. To do this, we need to add the Rating component to our app.

1. Add a rating component between the Selected Plant Label and Back Button. Rename this to Plant Rating.

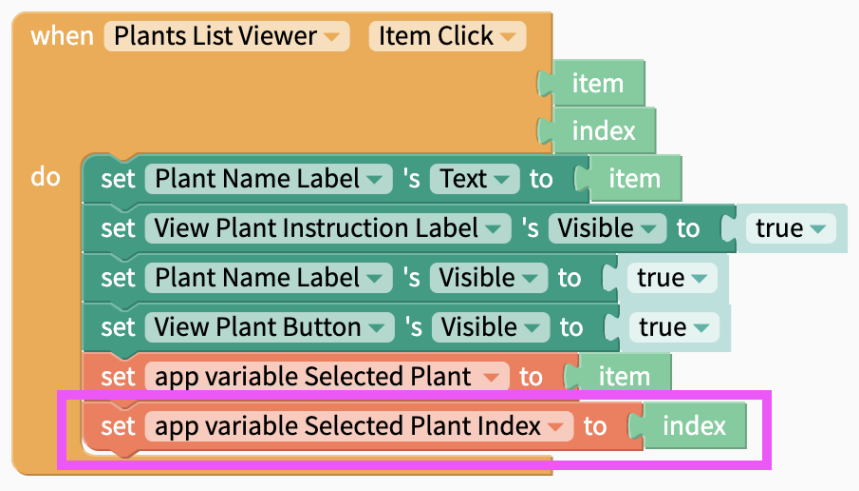


2. Add a button between the Plant Rating and the Back Button. Rename it to Save Rating Button and customize it to your liking.

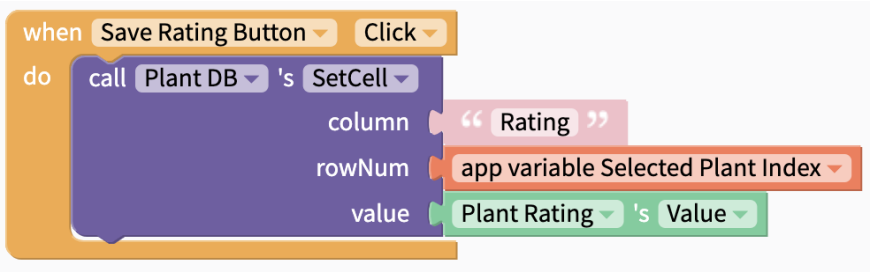


3. Now let's add code so that Plant Rating is saved to the Plant DB when the Save Rating Button is clicked. To do that, we first need to switch back to the blocks tab of the My Plants Screen.
4. At the My Plants Screen, initialize the Selected Plant Index variable.

5. Also at the My Plants Screen, set the Selected Plant Index to index under the when Plants List Viewer Item Click block.

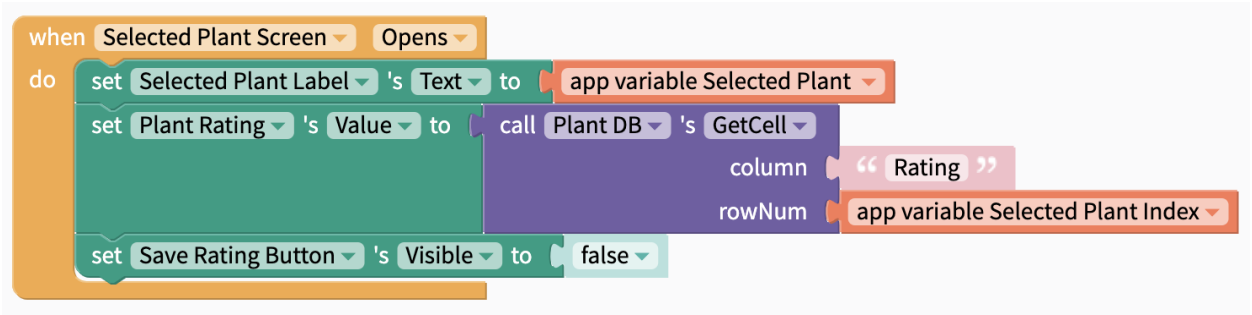


6. Now go back to the blocks tab of the Selected Plant Screen. Add the when Save Rating Button Click block to your workspace. Under the Plant DB component, choose the call Plant DB's SetCell block. Change the column to Rating, the rowNum to the Selected Plant Index variable, and value to Plant Rating's Value. This block of code will save the user's plant rating into the Plant DB.

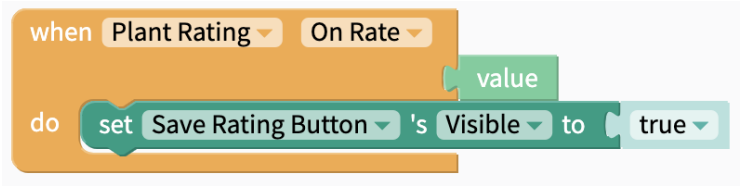


7. We're not finished! We need to make sure that the saved rating is shown after the user navigates away and back to the Selected Plant Screen.

8. To do this, add the set Plant Rating's Value to block to the workspace. Snap in the call Plant DB's GetCell block. Change the column name to Rating and rowNum to Selected Plant Index variable. Let's also hide the Save Rating Button when the screen opens.



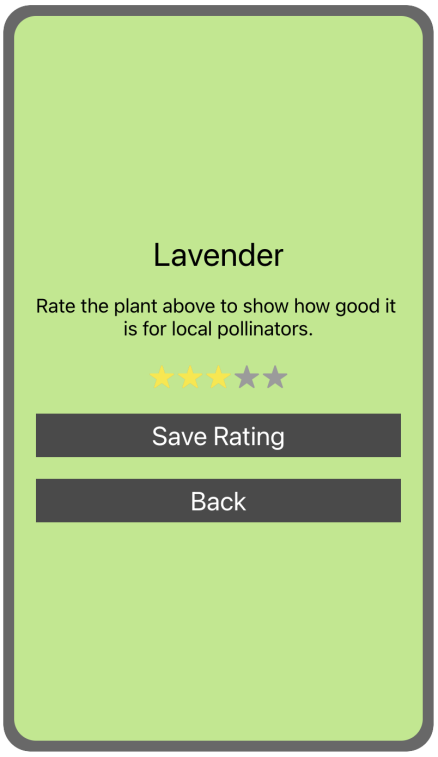
9. Next, click on the Plant Rating component and choose the when Plant Rating On Rate block. Snap in the set Save Rating Button's Visible to true block. This code will show the Save Rating Button when the user is rating the selected plant.



10. Lastly, let's add a label to let users know what the ratings are for. I rename this label to the Rating Instruction Label and change the text to "Rate the plant above to show how good it is for local pollinators."

Live Test

Start from the My Plants Screen, click on a plant name on the list viewer and then click on the View Plant button. When you navigate to the Selected Plant Screen, rate the plant you selected. Click on the Back Button and reselect the plant you just rated. You should see your previously saved rating of the plant.

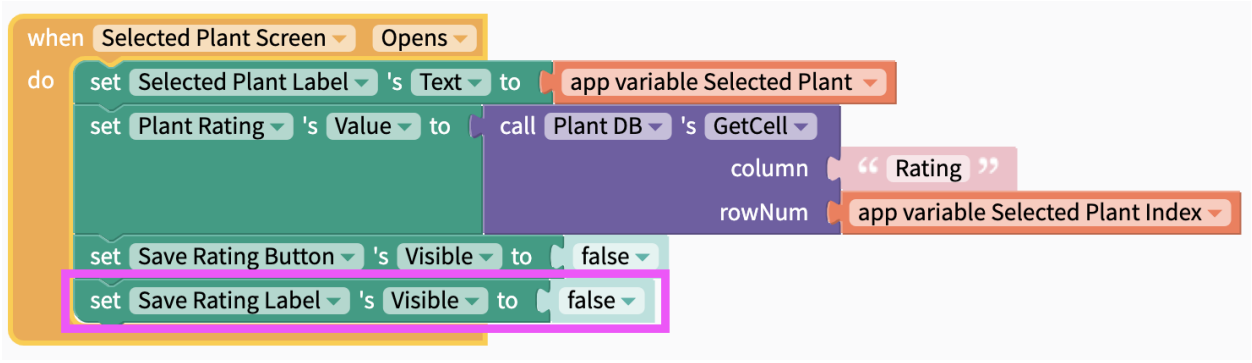


Step: Add Saved/Unsaved Rating Label

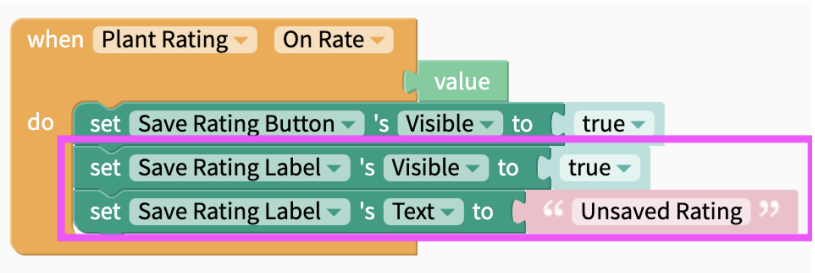
Next, we want to add a label that tells us if the rating has been saved or not.

1. Add a label between the Plant Rating component and the Save Rating button. I rename this to the Save Rating Label and changed the text to Unsaved Rating.

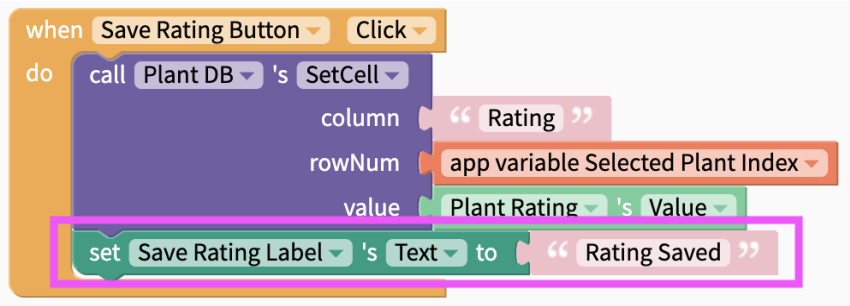
2. Add code to hide the Save Rating Label when the screen opens.



3. When the user is rating the plant, add code to show the Save Rating Label with the text "Unsaved Rating."

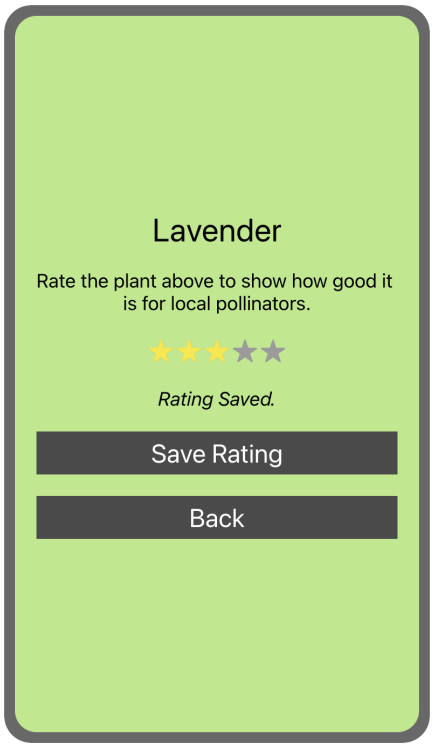


4. When the Save Rating Button is clicked, add code to change the Save Rating Label to "Rating Saved."



Live Test

- Start from the My Plants Screen, click on a plant name on the list viewer and then click on the View Plant button. When you navigate to the Selected Plant Screen, you should only see the Selected Plant Label, the Plant Rating component, and the Back Button.
- Rate the plant you selected and this will reveal the Save Rating Label and Button.
- When you click on the Save Rating button, the Save Rating Label will change from Unsaved Rating to Saved Rating.
- Click on the Back Button and reselect the plant you just rated. You should see your previously saved rating of the plant.



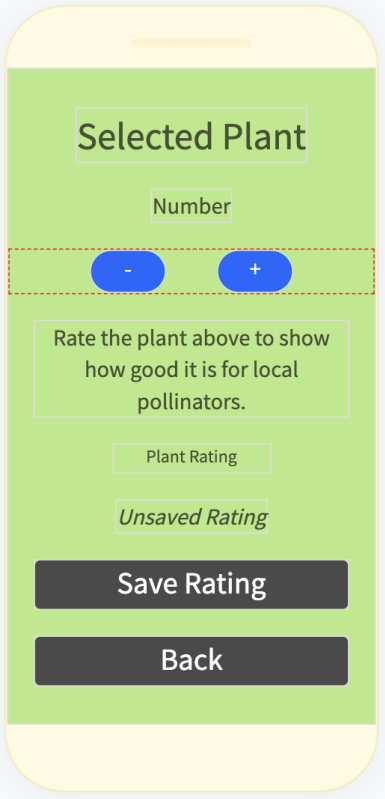
Step: Add plus and minus buttons to enter plant quantity

We want to store the number of plants in the Plant DB. We'll add buttons to help us do that.

1. Add the Number column to the Plant DB.

	Plant	Rating	Number
1	Lavender	0	0
2	Marigold	0	0
3	Cosmos	0	0
4	Elderberry	0	0
5			

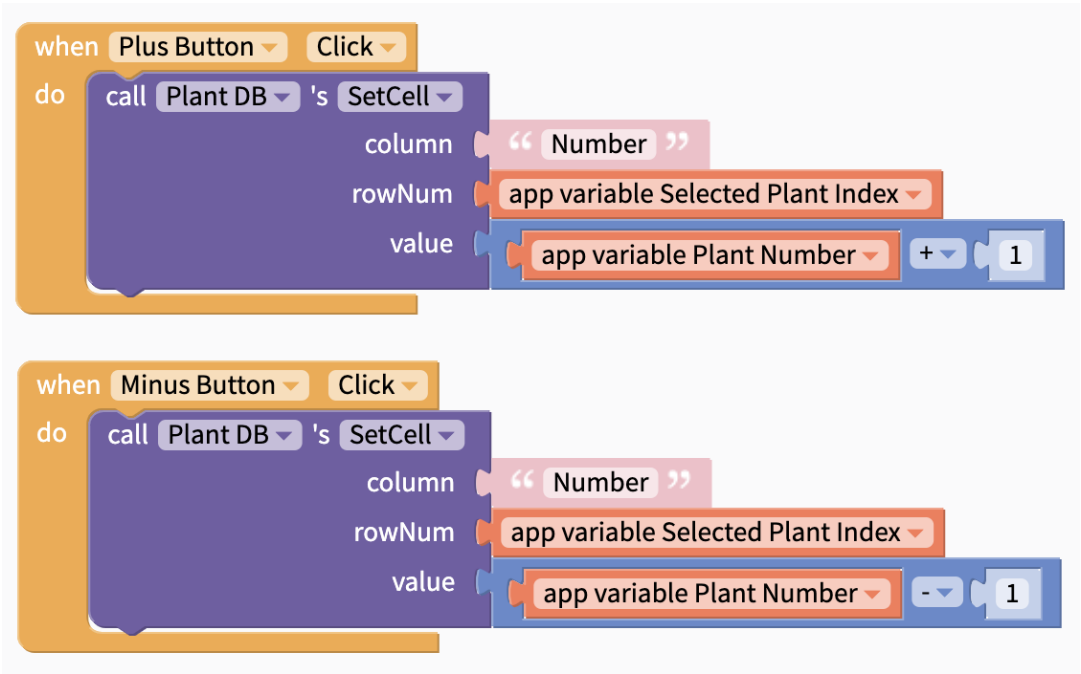
2. Add the Number Label under the Selected Plant Label.
3. Add a row below the Number Label. This will help you organize the Plus and Minus Buttons next to each other.
4. Customize these components to your liking.



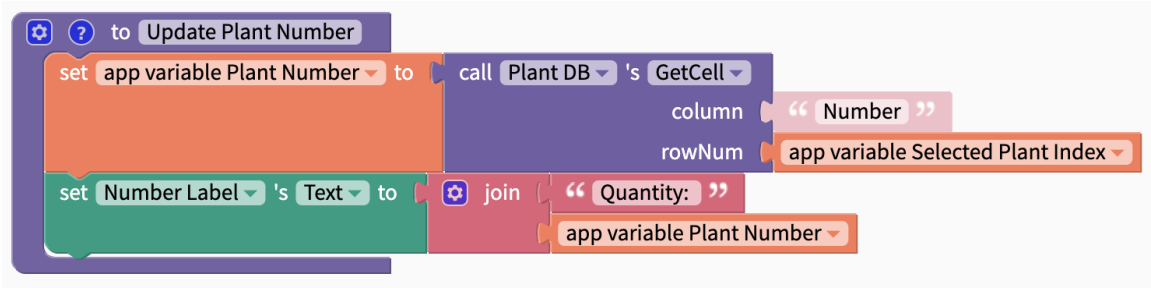
Step: Add and subtract plant quantity when Plus and Minus Buttons are clicked

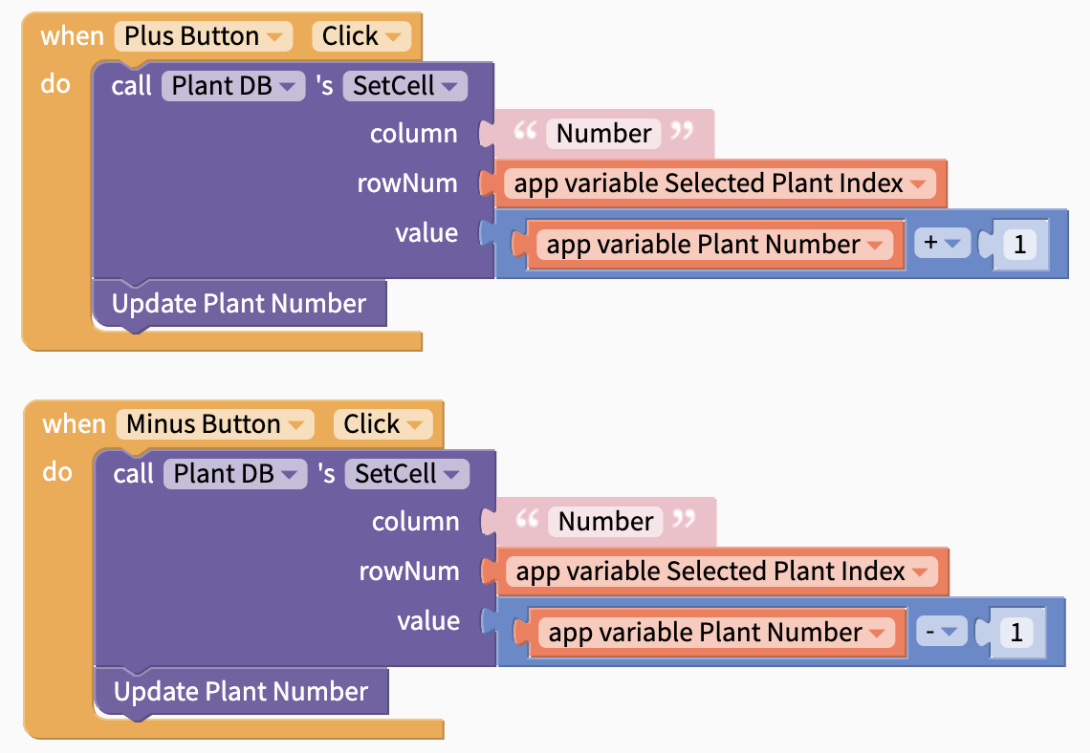
1. Initialize the Plant Number variable.

2. Add code to increase plant number by 1 each time the Plus Button is clicked.
3. Add code to decrease plant number by 1 each time the Minus Button is clicked.

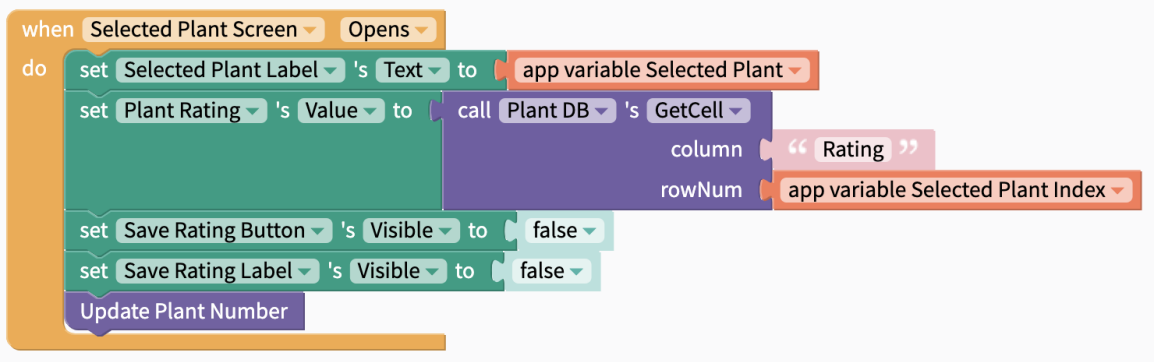


4. Add code to update the Plant Number Label when the Plus and Minus Buttons are clicked.

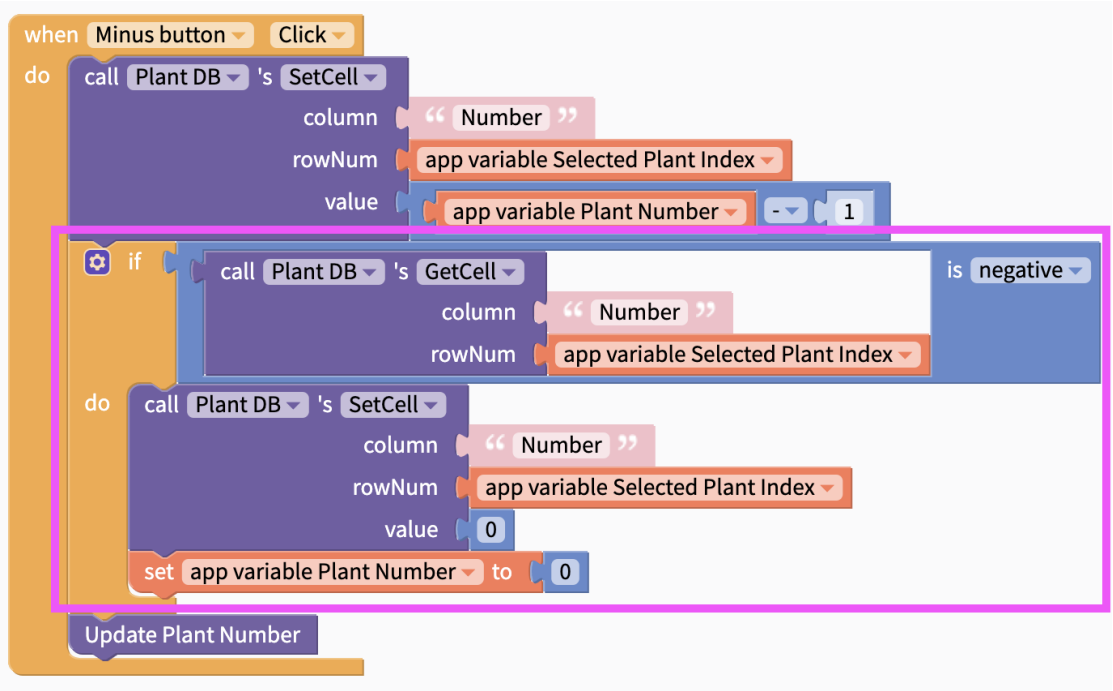




5. Make sure that plant quantity is updated when the Selected Plant Screen opens.



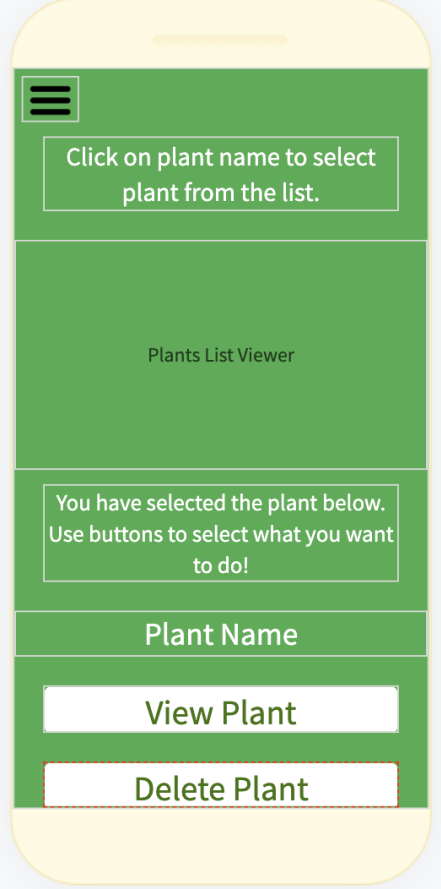
6. Add a condition to make sure that plant quantity doesn't become a negative number. This code sets the plant quantity to 0 should the number become a negative number.



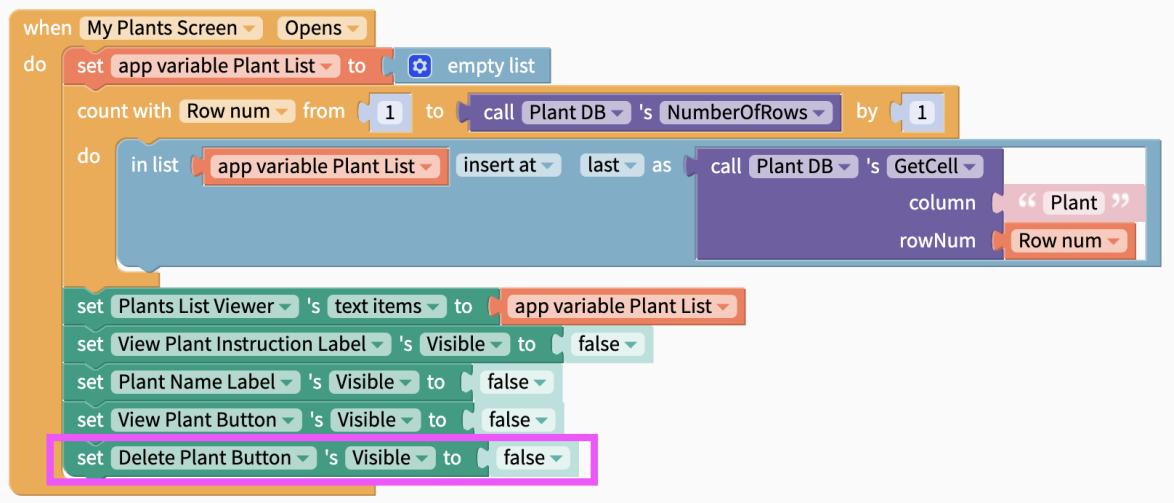
Step: Add Delete Button

Let's add a Delete Plant Button so we can delete our plant entries if we want to. This is actually a bit tricky to work with using the local DB. The reason being that we can't delete a row from the local DB.

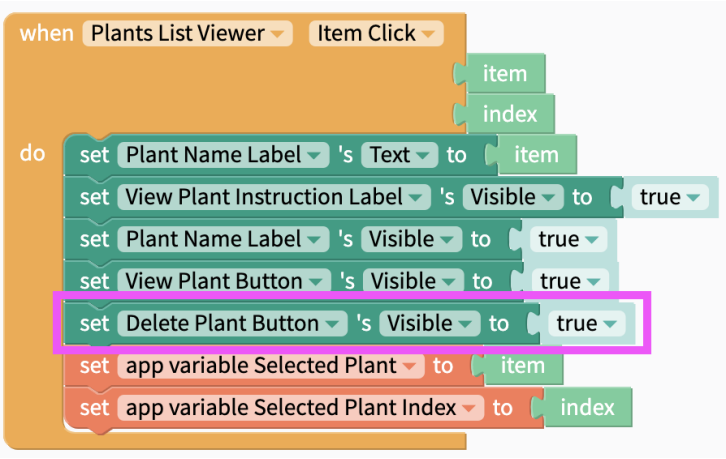
1. Add the Delete Plant Button and customize it however you want.



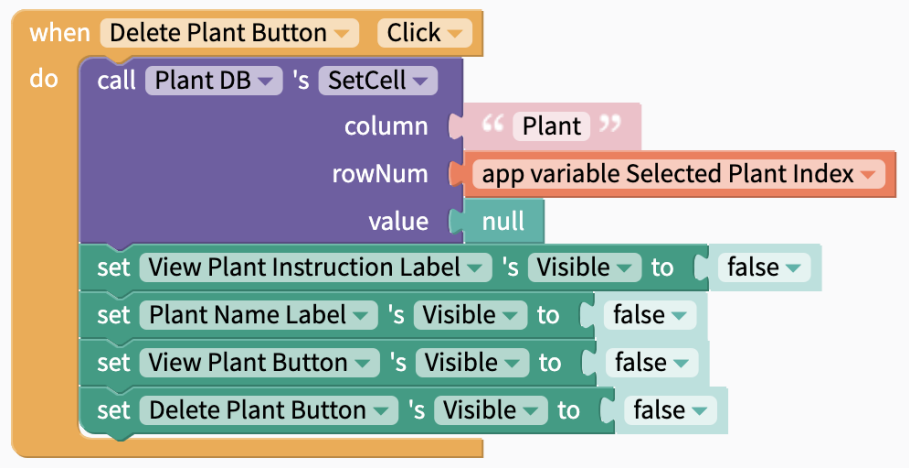
2. Hide the Delete Plant Button when the My Plants Screen opens.



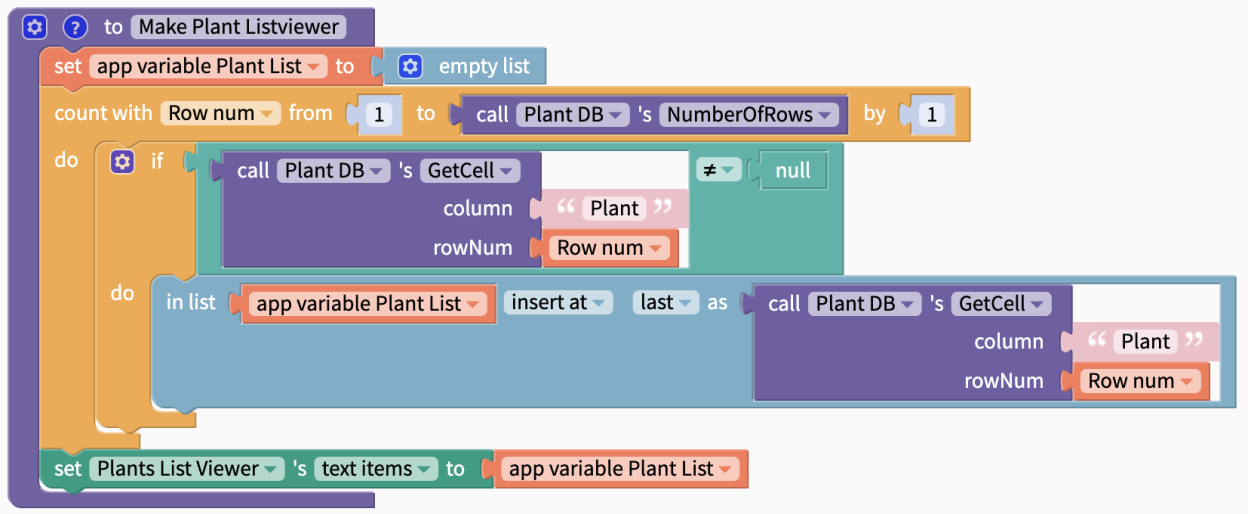
3. Show the Delete Plant Button when an item in the list viewer is clicked.



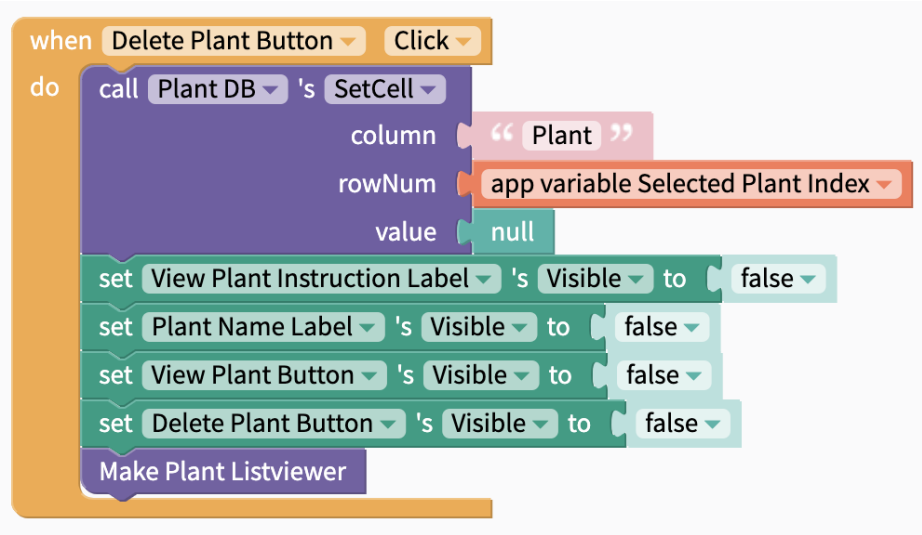
4. When the Delete Plant Button is clicked, set the Plant name to null. Hide the View Plant Instruction Label, Plant Name Label, View Plant Button, and Delete Button.



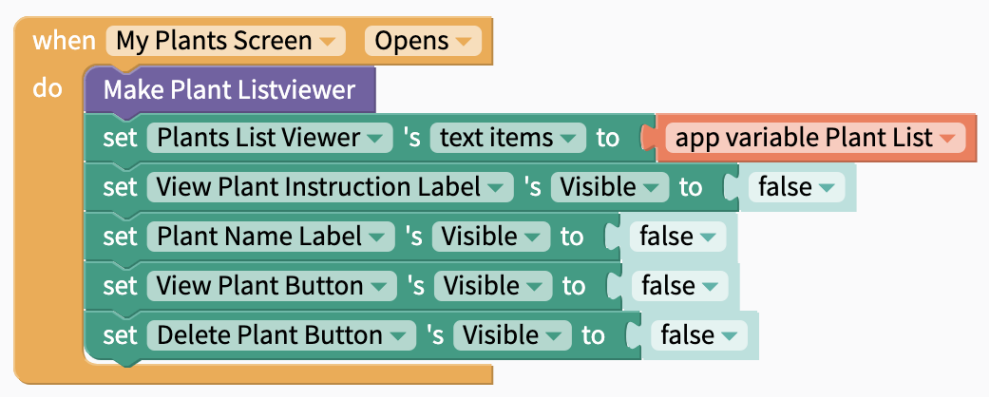
5. Write a function to Make Plant Listviewer. This function ensures that null values in the Plant column are not included in the listviewer.



6. Snap this under the When Delete Plant Button Click block to update the Plant List Viewer in real time.



7. Snap this under the When My Plants Screen Opens block to update the Plant List Viewer each time the screen opens.

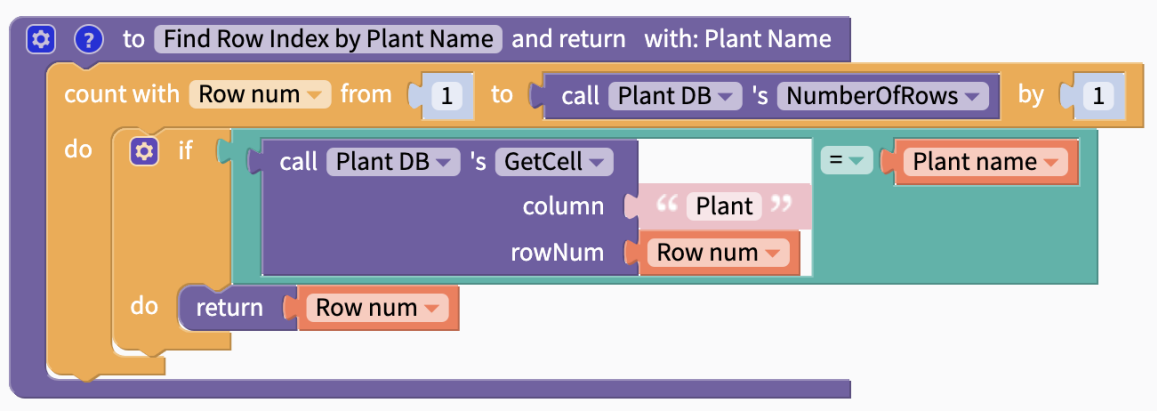


Live Test

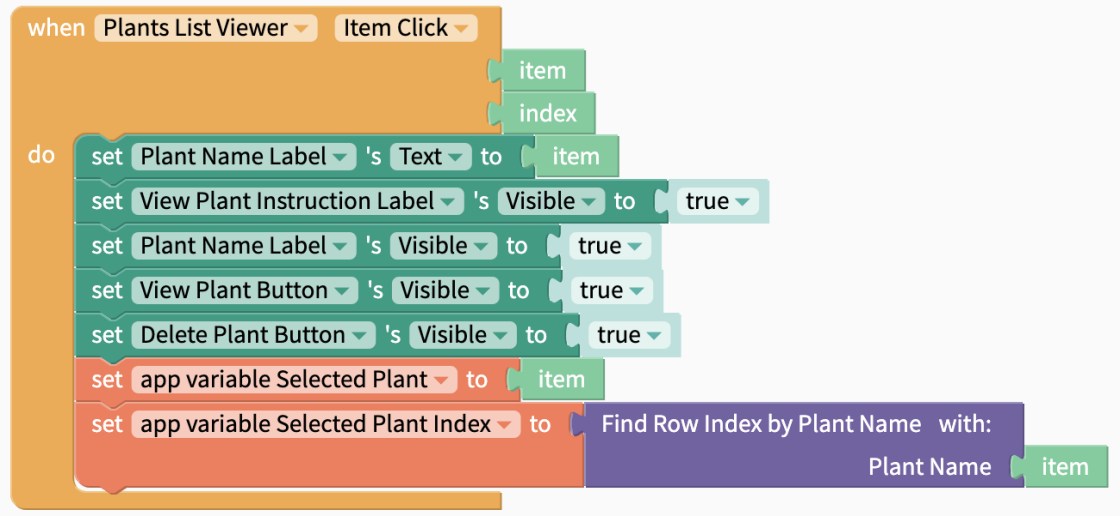
- Start from the My Plants Screen, click on the first plant name on the list viewer and then click on the Delete Plant Button.
- When your first selection is deleted, click on the first plant name on the list viewer again. You'll realize that it's not deleting.
- While the Selected Plant Index has changed, the row containing the deleted plant still exists in the Plant DB. Let's fix this.

Step: Add Delete Button (continued)

8. Write a function to find the row index by plant name.



9. Use this function to determine the Selected Plant Index.

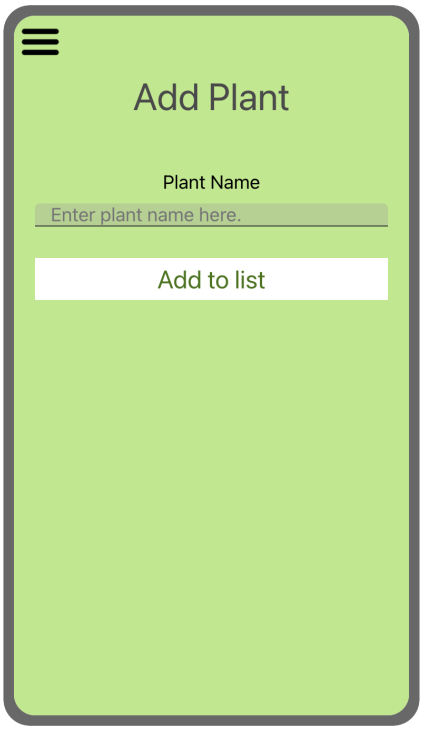


Live Test

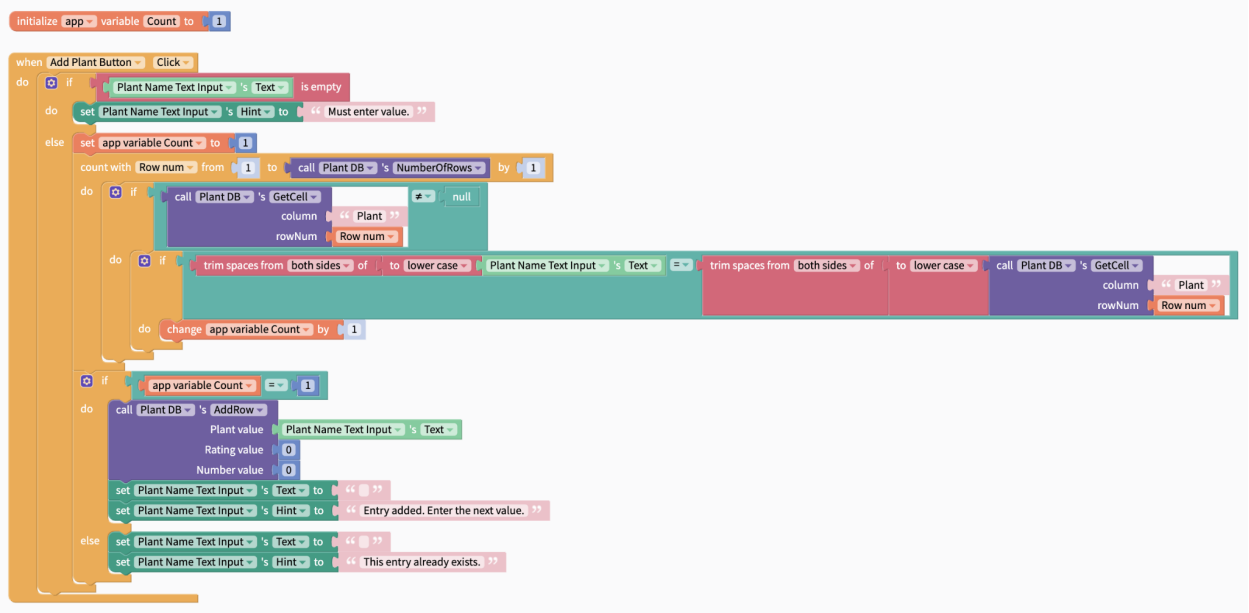
You should be able to delete plants from the list viewer without issues now.

Extension: Add Plant Screen

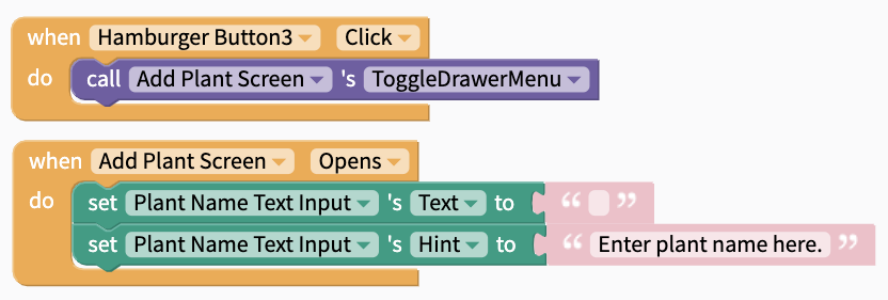
Now that you have the Plant DB set up, you can add the Add Plant Screen for users to enter new plants into the app. See notes below.



- The text input component is used on this screen for users to input new plant names.
- When the Add New Plant Button is clicked, the user input gets added into the Plant DB with a rating value of 0 if the input is not empty or a duplicate in the Plant DB.
    - The user input will be cleared and an “Entry added. Enter the next value.” hint will prompt users to enter a new plant if they want to.
    - If the entry already exists, show the hint “This entry already exists.”
    - This code will also filter for null values which occur when a plant name is deleted.



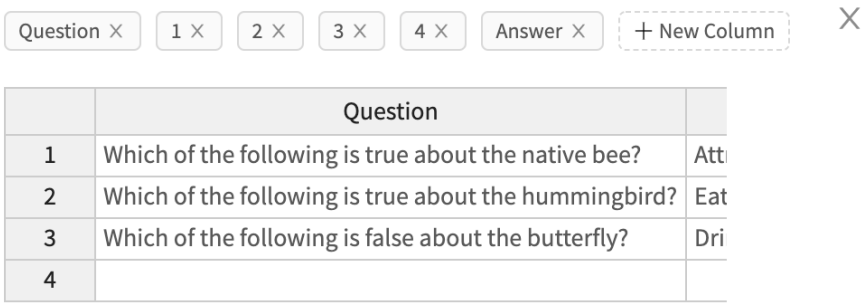
- The following code adds the Add Plant Screen to the drawer navigator and sets the hint to “Enter plant name here.” when the Add Plant Screen opens.



Extension: Add Pollinator Quiz Screen

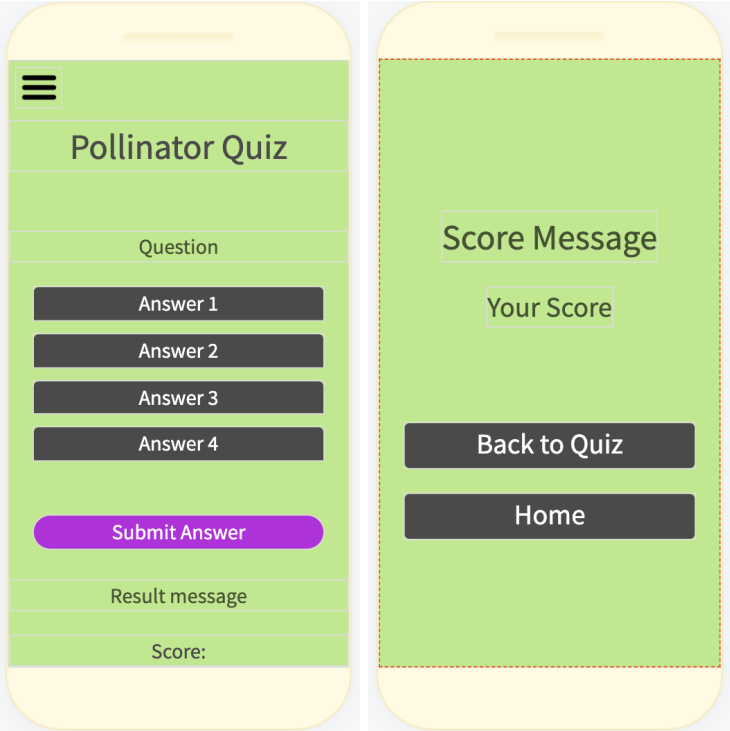
Let's add a Pollinator Quiz to help users learn why it's important to conserve pollinators. See notes below.

1. Set up Quiz DB.

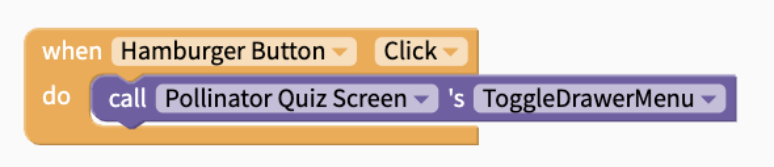


Question	1	2	3	4	Answer
Which of the following is true about the native bee?	Attracted to bright colours.	Drinks nectar.	Collects pollen on bristles on its legs and body.	All of the above.	4
Which of the following is true about the hummingbird?	Eats with a long beak.	Needs an area where it can land on a flower.	Collects pollen on bristles on its legs and body.	Colour of a flower is not a factor.	1
Which of the following is false about the butterfly?	Drinks nectar.	Can "feel" nectar with feet.	Eats pollen.	Attracted to bright colours.	3

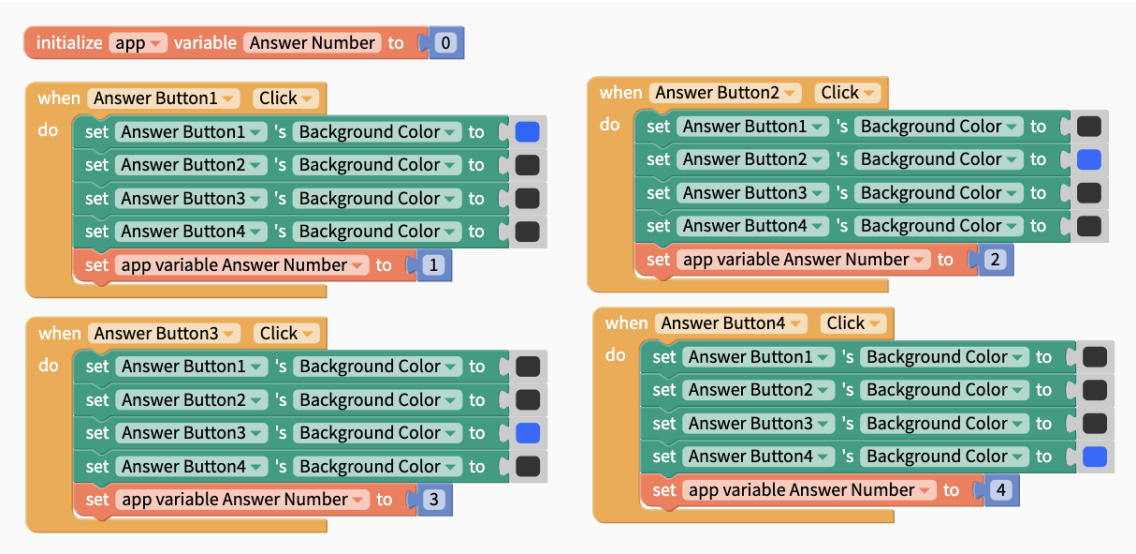
2. Create Pollinator Quiz Screen and Score Screen.



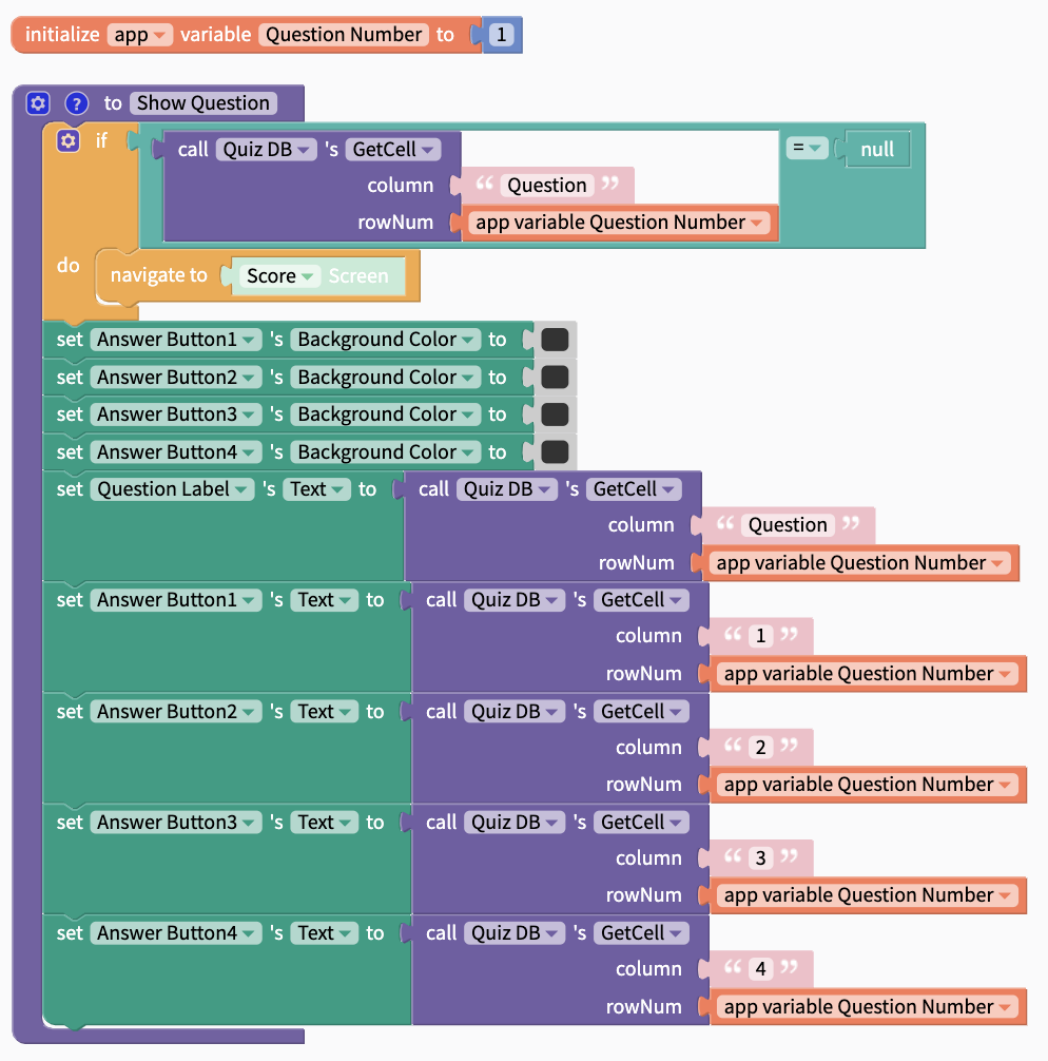
3. Add code to add Pollinator Quiz Screen to drawer navigator.



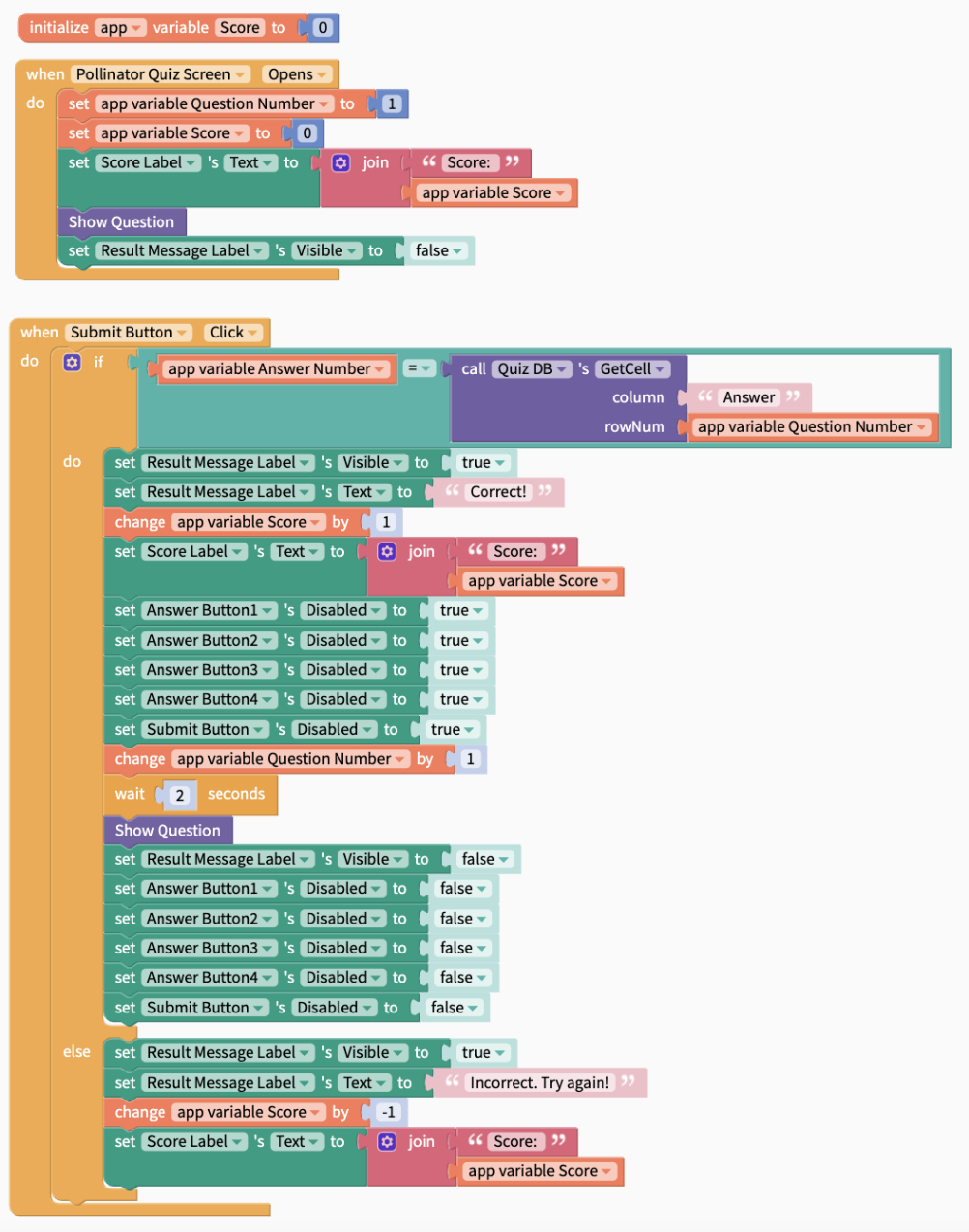
4. Save the selected answer with the code below.



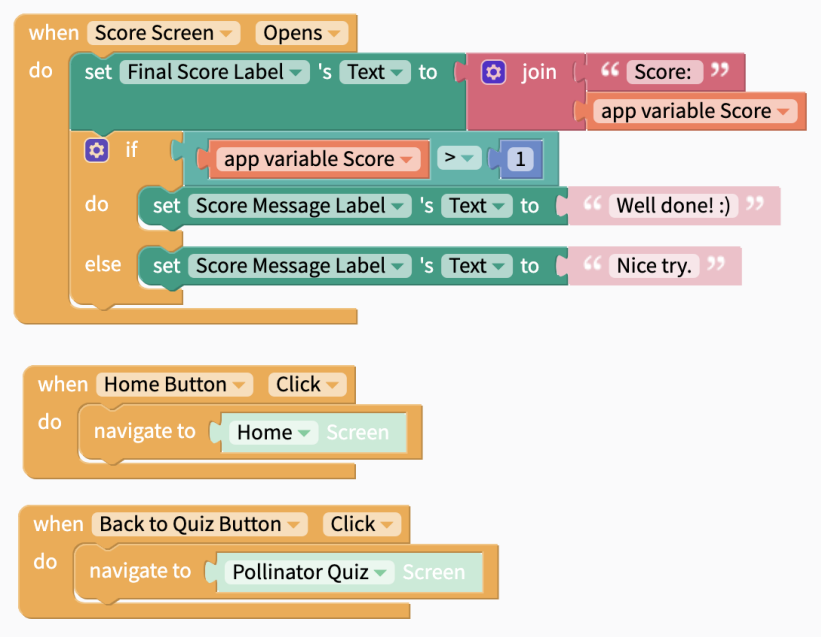
5. Write a function to show questions.



6. The code below shows Question 1 from Quiz DB when the Pollinator Quiz Screen opens.
    - When the Submit Button is clicked, it'll check if the answer is correct.
    - If the answer is correct, the Answer Buttons are disabled and the score increases by 1 and the next question will show up.
    - If the answer is incorrect, the user gets to try again but the score decreases by 1 each time the user gets the answer wrong.
    - When all questions from Quiz DB have been attempted, the user will navigate to the Score Screen (see the function above).



7. At the Score Screen, a score of > 1 will show the score message “Well done! :)” and anything < 1 will show the score message “Nice try.” Users can click on the buttons to return to the Home Screen or the Pollinator Quiz Screen to try the quiz again.

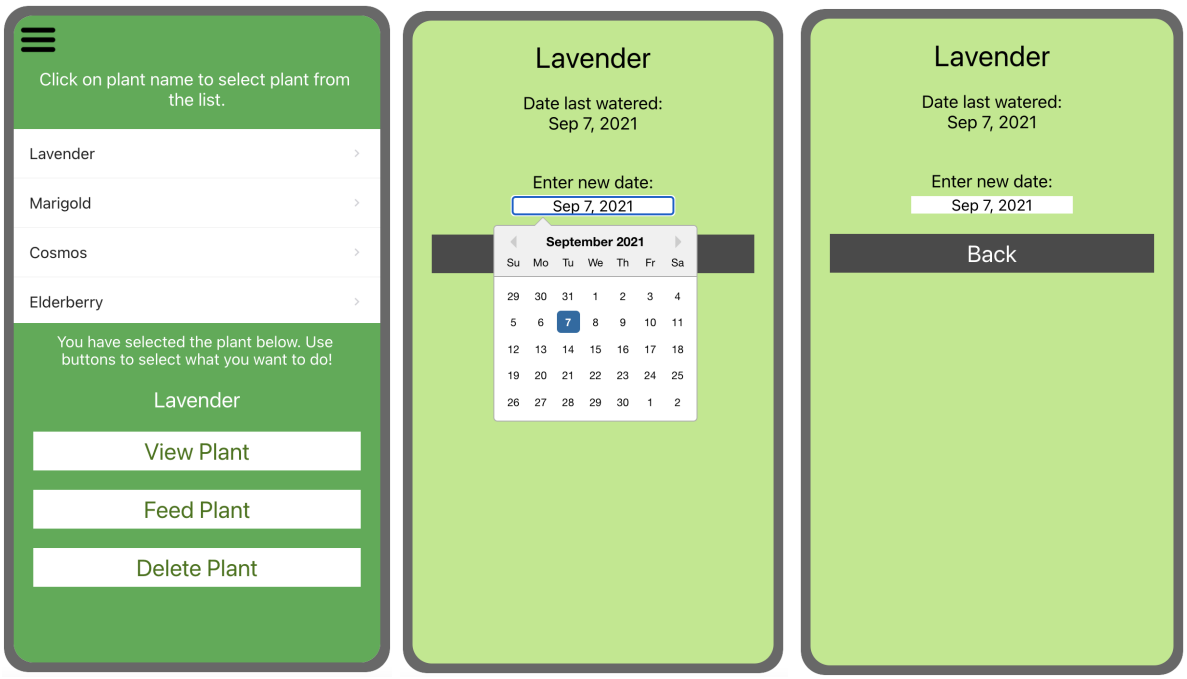


Challenge

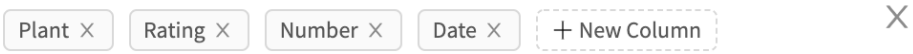
- Make questions appear at random and not repeat themselves.
- Add a Timer component.

Extension: Add Feed Plant Screen

This feature allows users to store the date they last watered their plants.



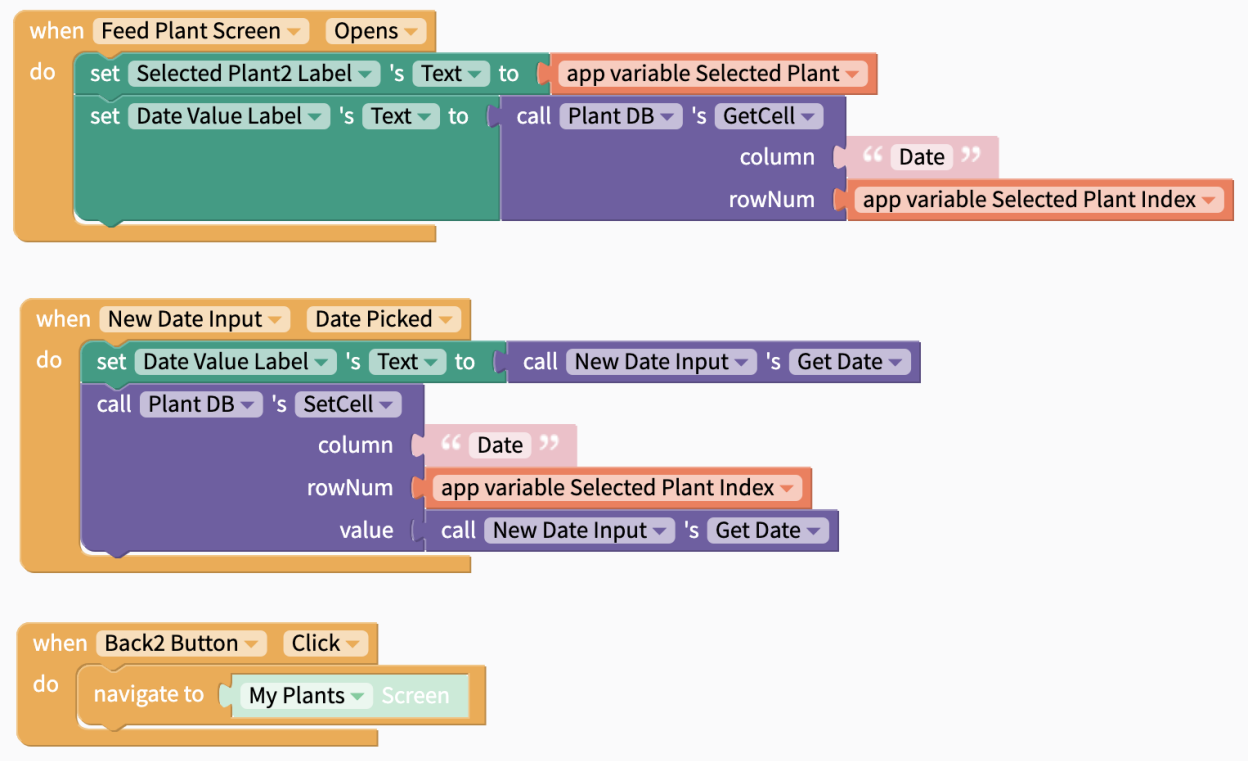
1. Add Date column in Plant DB.



	Plant	Rating	Number	Date
1	Lavender	0	0	0
2	Marigold	0	0	0
3	Cosmos	0	0	0
4	Elderberry	0	0	0
5				

2. Go back to the My Plants Screen and add the Feed Plant Button. Make sure you make changes to how the Feed Plant Button appears/disappears.

3. Add code below to save the date input and also a Back Button to return to the My Plants Screen.



Challenge

- Add conditions so users cannot enter a future date in the Date Input component.
- Add a label to show the number of days since the user last watered the plant.

Extension: Add Plant Stores Screen

This feature shows a map with markers of a few plant stores in the Lower Mainland of BC. You need a mobile device or a table to test this out.

1. Add Plant Stores DB.

Store × | Latitude × | Longitude × | + New Column

	Store	Latitude	Long
1	Its About Thyme Nursery	49.20163794893492	-122.9743(
2	GARDENWORKS	49.265680904646395	-122.9743(
3			

2. The code below will set the default location to the SFU Burnaby campus. It will then go through the Plant Stores DB to add markers. When you click on the location marker, the store name will show up.