

Kristy Stark

Let it Grow Garden Planner Project Documentation



Executive Summary

The Let it Grow Garden Planner is a digital space where data-driven gardeners can plan what to grow and analyze trends that support their gardening goals. There are many factors that impact how well a plant will grow, which makes advanced planning important to achieve high yields and other positive outcomes.

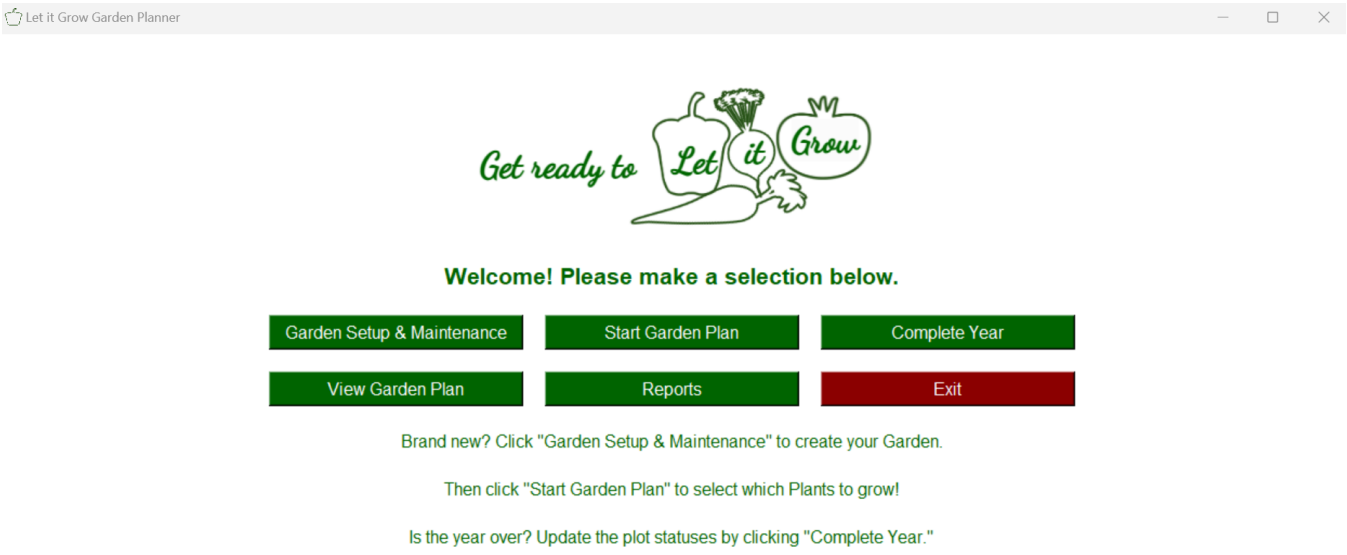
The Let it Grow Garden Planner is designed to streamline this planning process. The gardener will input details about which plants they wish to grow and the timeframe. The program can then use available data and default guidelines to suggest a planting layout. Alternatively, the gardener can manually create a custom plan for the program to approve based on specified parameters.

A secondary function is to provide a repository and retrieval system for tips and other useful information about the plants themselves. Examples are watering frequency recommendations or how deep seeds and bulbs need to be planted. This data would not be needed for the planning algorithm but is still useful to have at one's fingertips throughout the season.

Finally, the Let it Grow Garden Planner has the means to document outcomes, which along with a reporting functionality can help gardeners better identify trends over time. Gardeners will gain access to meaningful insights into successes and opportunities, allowing them to celebrate their achievements and make more informed future planning decisions.

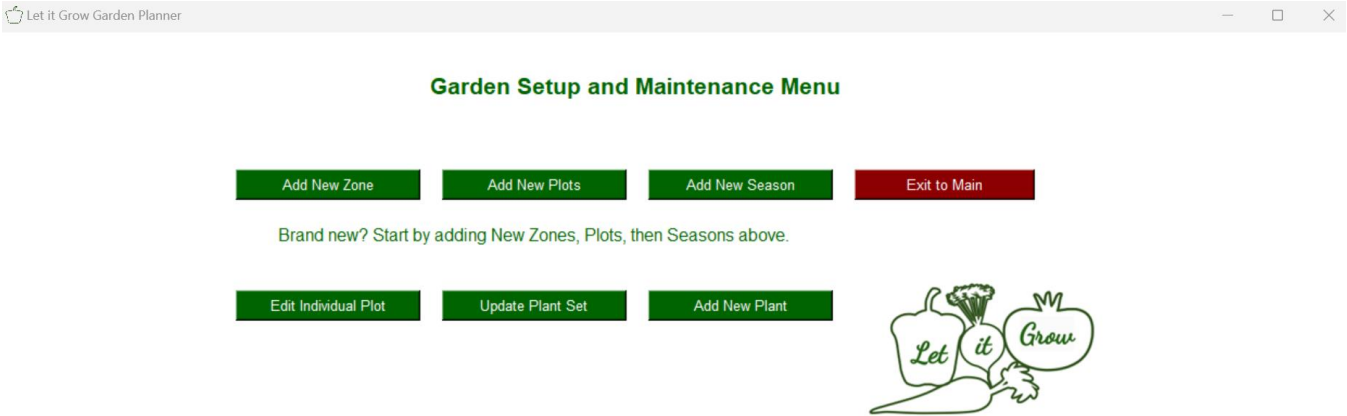
Scenario | Main Menu / Welcome Screen

When the user first opens the program, this window opens which contains buttons to launch all other selections.



Scenario | Garden Setup and Maintenance Menu

The user selects this window to reach a sub-menu for initial configuration and editing options.
A first time user visits the Add New Zone, Add New Plots and Add New Season windows to set up their “Garden” and range of seasons before starting their first Garden Plan.



Scenario | Add New Zone

The user visits this page to identify a Zone to add to the database. A “Zone” is a contiguous set of plots and used primarily for plot identification purposes. At least one Zone is required to create a Garden Plan.

Let it Grow Garden Planner

Add New Zone

Back to Setup MenuExit to Main

A "plot" is a space where a single plant or plant set can be planted.
A "Zone" is a group of contiguous plots, such as a designated garden area or raised bed

Please enter a unique name for your zone:

Ex. "Large Garden, Blue Flowerpot 1"

Number of rows of plots in this zone:

Number of columns of plots in this zone:

Zone Notes (Optional):

Add Zone

Scenario | Add New Plots

The user visits this page to add one or more new Plots to the database. A “Plot” is a square unit within a zone on which one or more plants can be planted. Plots that share the same qualities can be added in bulk. Row and column numbers are auto-assigned based on the number selected and those already existing.

Let it Grow Garden Planner

Add New Plots

Back to Setup MenuExit to Main

Please select a Zone:

Select Value

Number of rows of identical plots to add:

Number of columns of identical plots to add:

Size of each Plot: Unit of Measurement:

Select Value

☐ Is this plot a container (solid bottom)?

If container, container depth in inches:

Sun Level of Plot(s):

Select Value

Soil Moisture Level of Plot(s):

Select Value

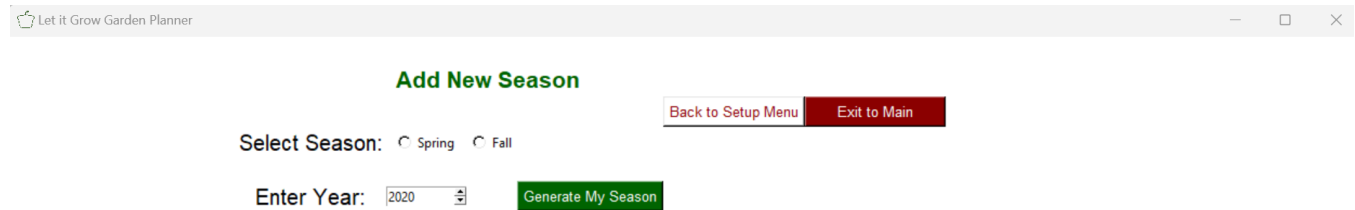
Click Refresh if new zones have been added:

Refresh

Add Plots

Scenario | Add New Season

A user visits this window to add a new planting Season to the database. The system “Season” name is auto-generated based on season and year selected. All active Seasons are included in the garden plan.

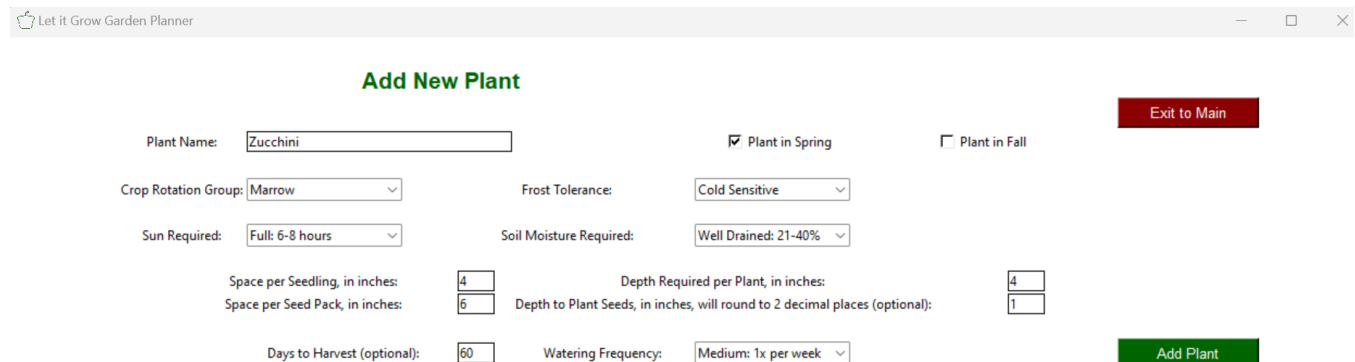


The screenshot shows a window titled "Let it Grow Garden Planner" with a title bar containing standard window controls. The main heading is "Add New Season" in green. Below it, there are two buttons: "Back to Setup Menu" (white with a red border) and "Exit to Main" (red). The "Select Season:" label is followed by two radio buttons: "Spring" (selected) and "Fall". Below this, the "Enter Year:" label is followed by a text input field containing "2020" and a green "Generate My Season" button.

Scenario | Add New Plant

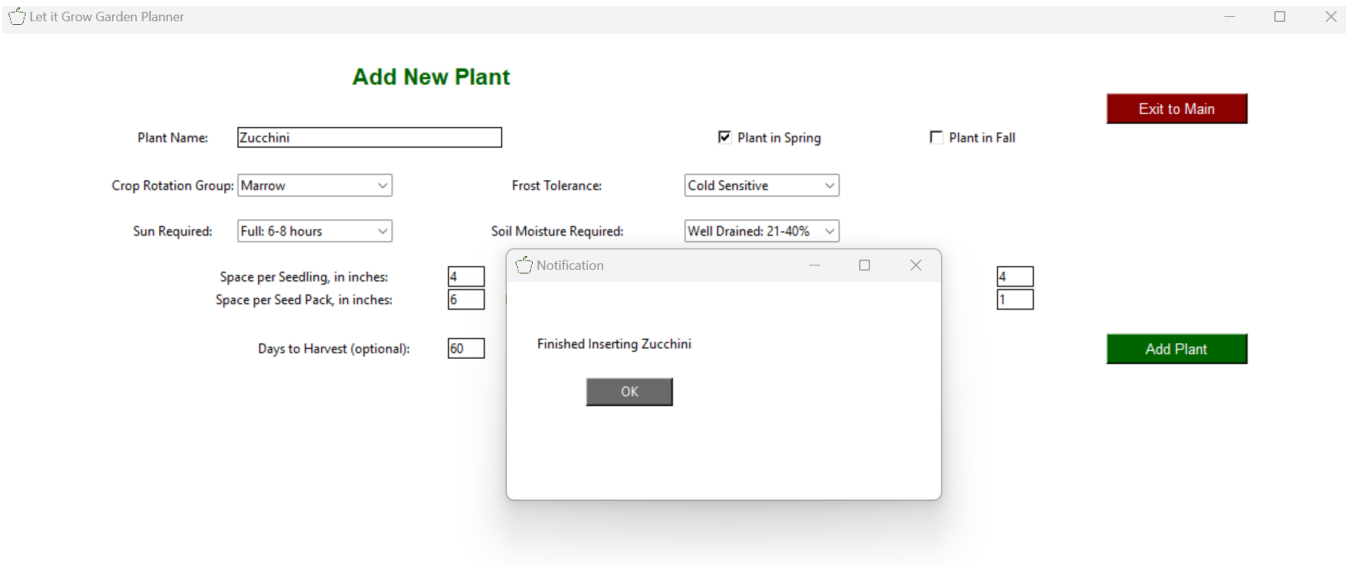
A user visits this window if they wish to add a new plant to the master list.

While it's technically optional because there are some Plants included by default, the majority of users will likely want to add their own before starting their first Garden Plan.



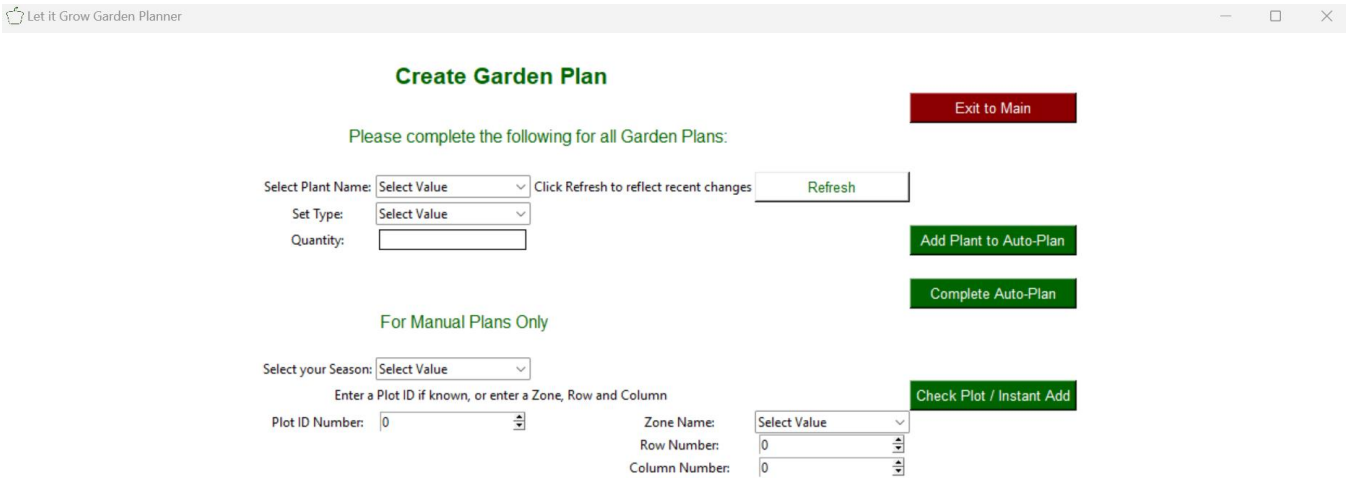
The screenshot shows a window titled "Let it Grow Garden Planner" with a title bar containing standard window controls. The main heading is "Add New Plant" in green. Below it, there are two buttons: "Exit to Main" (red) and "Add Plant" (green). The "Plant Name:" label is followed by a text input field containing "Zucchini". To the right, there are two checkboxes: "Plant in Spring" (checked) and "Plant in Fall" (unchecked). Below these, there are four dropdown menus: "Crop Rotation Group:" (Marrow), "Frost Tolerance:" (Cold Sensitive), "Sun Required:" (Full: 6-8 hours), and "Soil Moisture Required:" (Well Drained: 21-40%). Below the dropdowns, there are four text input fields: "Space per Seedling, in inches:" (4), "Depth Required per Plant, in inches:" (4), "Space per Seed Pack, in inches:" (6), and "Depth to Plant Seeds, in inches, will round to 2 decimal places (optional):" (1). Below these, there are two more text input fields: "Days to Harvest (optional):" (60) and "Watering Frequency:" (Medium: 1x per week).

The user will receive a confirmation message if the Plant is successfully added to the database, otherwise they will receive an error message indicating the first field found to be missing or invalid. This is true for all windows where data is added or edited.



Scenario | Create Garden Plan

A user visit this window each time they want to select plants to add to their automatic Garden Plan or create a custom Garden Plan manually, if they choose.



Scenario | Add to and Complete Auto-Plan

If the user wants to allow the system to create the Garden Plan automatically, they can add plants one at a time to a list, referred to as the tentative Garden Plan.

Let it Grow Garden Planner

Create Garden Plan

Exit to Main

Please complete the following for all Garden Plans:

Select Plant Name: Carrot

Click Refresh if new plants have been added

Refresh

Set Type: Seed Pack

Quantity: 1

Add Plant to Auto-Plan

Complete Auto-Plan

Check Plot / Instant Add

Select your Plot ID

Select Value

0

0

Notification

1 Seed Pack of Carrot added to tentative Garden Plan.

OK

Notification

2 Bulb(s) of Garlic added to tentative Garden Plan.

OK

Create Garden Plan

Exit to Main

Please complete the following for all Garden Plans:

Click Refresh if new plants have been added

Refresh

Add Plant to Auto-Plan

Complete Auto-Plan

Check Plot / Instant Add

For Manual Plans Only

Select your Season: Select Value

Enter a Plot ID if known, or enter a Zone, Row and Column

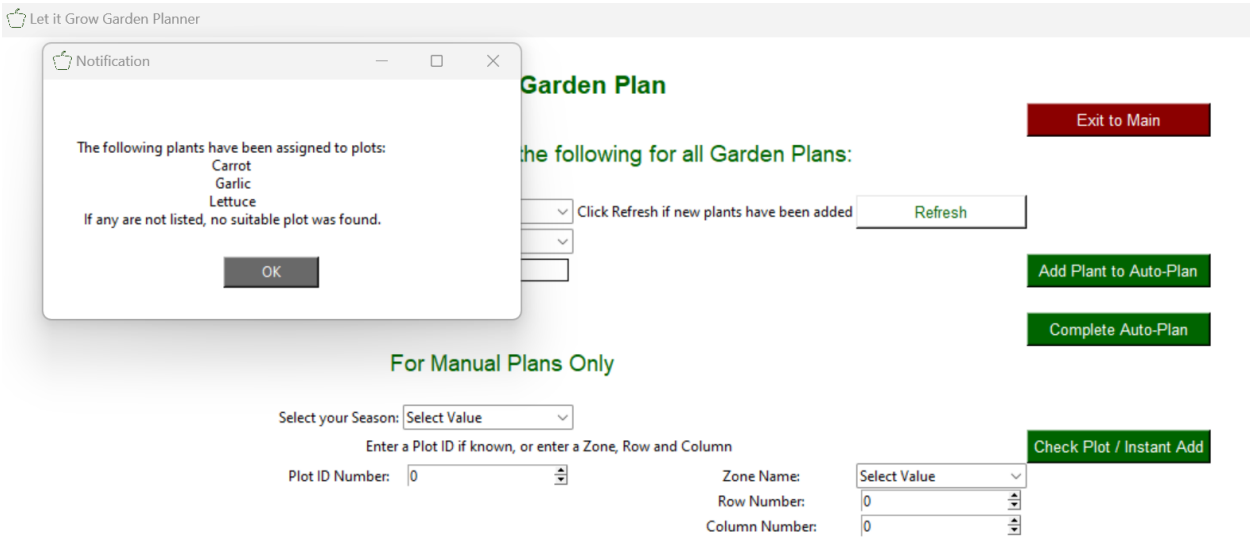
Plot ID Number: 0

Zone Name: Select Value

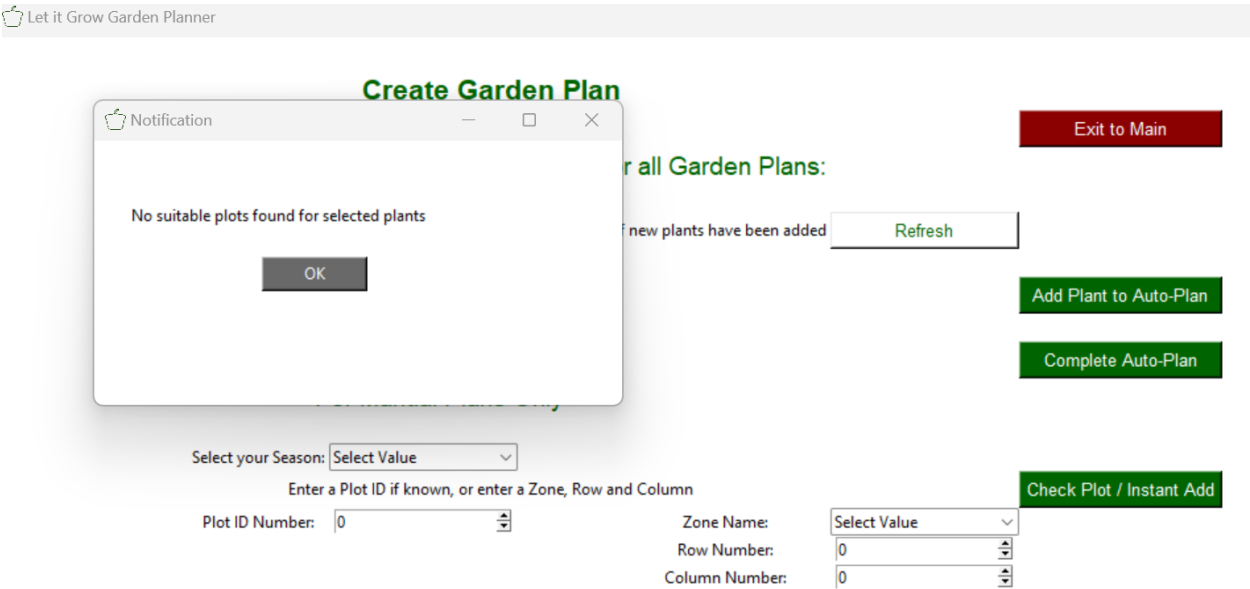
Row Number: 0

Column Number: 0

Once all desired plants have been added to the tentative Garden Plan list, the user clicks the Complete Auto-Plan button to run the algorithm to check the plants' requirements against the available plots' qualities. A popup message appears if any plants were successfully assigned to plots.



If no suitable plots are found, a failure message appears instead.



Scenario | Check and Complete Manual Plan

To manually attempt to assign a plant set to a specific plot and season, the user completes the applicable fields and then clicks “Check Plot / Instant Add.”

If a Plot ID number other than 0 is entered, the system will check the qualities of that Plot ID against the plant’s requirements. If the Zone, Row, and Column are entered instead, the system will use that data to look up the associated Plot ID to use to perform the checks.

Like with the auto-plan above, the user will get a confirmation message if the plant was successfully assigned to the selected plot, otherwise they will get an error message.

Create Garden Plan

Please complete the following for all Gardens

Select Plant Name: Pepper

Set Type: Seedling(s)

Quantity: 1

Click Refresh if new plants have been added

For Manual Plans Only

Select your Season: Spring24

Enter a Plot ID if known, or enter a Zone, Row and Column

Plot ID Number: 1

Zone Name: Select Value

Row Number: 0

Column Number: 0

Check Plot / Instant Add

Notification

Failed checks, Pepper not added to plan

OK

Scenario | Update Plant Set

Here a user can enter and edit details about a specific Plant Set which they are growing or have grown, including the dates they planted and harvested the set, the outcome, and optional notes.

Let it Grow Garden Planner

Update Plant Set

Select Plant Name: Garlic

Season: Fall24

Search

Next in Season

Exit to Main

Plant Set ID: 5

Plot ID Number: 3

Set Type: Bulb(s)

Quantity: 1

Date Planted: 9/27/24

First Harvest Date: 7/3/25

Last Harvest Date: 8/29/25

Plant Set Notes (Optional):

Outcome: ☐ Pending ☒ Success ☐ Failure

Save Without Checking

Check and Save Changes

Error handling logic includes checking that the date planted comes before the first date harvested, that the first harvest date comes before the last, and that if there is a first harvest date there is also a date planted, and so on.

Let it Grow Garden Planner

Edit Plant Set

Season: Fall24

Search

Next in Season

Exit to Main

Set Type: Bulb(s)

Quantity: 1

Date Planted: 8/2/24

First Harvest Date: 7/29/24

Last Harvest Date:

Outcome: ☐ Pending ☐ Success ☒ Failure

Save Without Checking

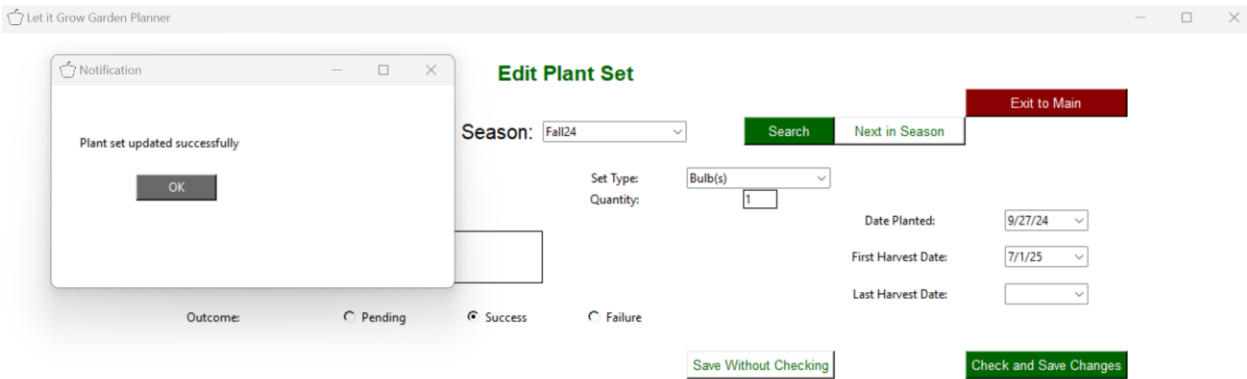
Check and Save Changes

Notification

Harvest date must be at least one (1) day after date planted.
Plant set not edited.

OK

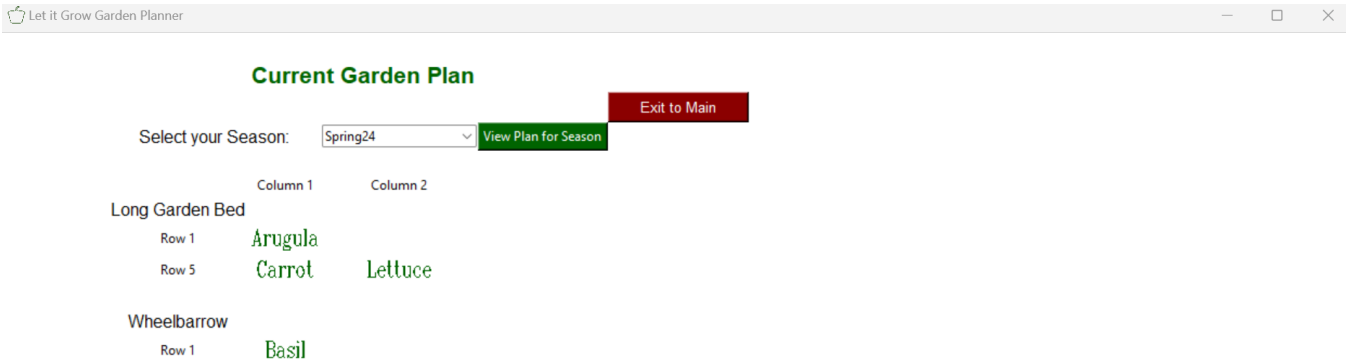
The user will receive a confirmation message if the Plant Set is saved successfully.



Scenario | View Current Garden Plan

The user clicks the View Garden Plan button from the Main Menu to see the current Garden Plan in grid view by season.

They select the season from the dropdown list then click View Plan for Season to generate the display.



Scenario | Complete Year

When a calendar year is over, a user clicks the Complete Year button from the Main Menu, which opens a window to close out the calendar year. The purpose of closing out a year is to update the plots' nitrogen levels in the database and mark the Seasons as inactive so that plant sets will be no longer be assigned to them in future Garden Plans.

🍷 Let it Grow Garden Planner

Complete Year

Select Year to Finalize:

2024

Go

Exit to Main

Complete

Season	Plant Set ID	Plant	Zone	Plot Number	Date Planted	Last Harvest	Outcome	Season Active
Spring24	2	Carrot	Long Garden Bed	13	2024-06-01	2024-09-13	Success	Yes
Spring24	4	Lettuce	Long Garden Bed	14	2024-05-10	2024-07-22	Success	Yes
Fall24	1	Garlic	Long Garden Bed	1	2024-09-27	None	Failure	Yes
Fall24	3	Garlic	Long Garden Bed	2	2024-09-27	2025-07-01	Failure	Yes
Fall24	5	Garlic	Long Garden Bed	3	2024-09-27	2025-08-29	Success	Yes
Fall24	6	Garlic	Long Garden Bed	4	2024-11-01	2025-07-28	Success	Yes

If the year is closed out successfully, the user sees a conformation message.

🍷 Let it Grow Garden Planner

Complete Year

Select Year to Finalize:

2024

Go

Exit to Main

Complete

Season	Plant Set ID	Plant	Zone	Plot Number	Date Planted	Last Harvest	Outcome	Season Active
Spring24	2	Carrot	Long Garden Bed	13	2024-06-01	2024-09-13	Success	Yes
Spring24	4	Lettuce	Long Garden Bed	14	2024-05-10	2024-07-22	Success	Yes
Fall24	3	Garlic	Long Garden Bed	2	2024-09-27	2025-07-01	Failure	Yes
Fall24	1	Garlic	Long Garden Bed	1	2024-09-27	None	Failure	Yes
Fall24	5	Garlic	Long Garden Bed	3	2024-09-27	2025-08-29	Success	Yes
Fall24	6	Garlic	Long Garden Bed	4	2024-11-01	2025-07-28	Success	Yes

🍷 Notification

Year completed successfully!

OK

The user can click “Go” again to refresh the onscreen values and confirm all applicable Seasons have been marked inactive.

Complete Year

Select Year to Finalize:

2024

Go

Exit to Main

Complete

Season	Plant Set ID	Plant	Zone	Plot Number	Date Planted	Last Harvest	Outcome	Season Active
Spring24	2	Carrot	Long Garden Bed	13	2024-06-01	2024-09-13	Success	No
Spring24	4	Lettuce	Long Garden Bed	14	2024-05-10	2024-07-22	Success	No
Fall24	3	Garlic	Long Garden Bed	2	2024-09-27	2025-07-01	Failure	No
Fall24	1	Garlic	Long Garden Bed	2	2024-09-27	2025-08-29	Success	No
Fall24	5	Garlic	Long Garden Bed	3	2024-09-27	2025-08-29	Success	No
Fall24	6	Garlic	Long Garden Bed	4	2024-11-01	2025-07-28	Success	No

Seasons not associated with the selected year remain active.

Complete Year

Select Year to Finalize:

2025

Go

Exit to Main

Complete

Season	Plant Set ID	Plant	Zone	Plot Number	Date Planted	Last Harvest	Outcome	Season Active
Spring25	7	Arugula	Long Garden Bed	1	None	None	TBD	Yes

Scenario | Edit Individual Plot

A user visits this optional page from the Settings Menu to edit certain details about a particular plot. Some examples are a change in the sun or soil moisture levels, or if extra nitrogen is added to the soil.

Let it Grow Garden Planner

—

□

×

Edit Individual Plot

Back to Setup Menu

Exit to Main

Enter Plot ID to Edit:

1

Search

Size of Plot:

12

Unit of Measurement:

Inches

Sun Level of Plot:

Full: 6-8 hours

Soil Moisture Level of Plot:

Well Drained: 21-40%

Current Nitrogen Level of Plot:

4

☐ Is this plot a container (solid bottom)?

If container, container depth in inches:

☒ Plot Active

Save Changes

Scenario | Reports Menu

The user visits the Reports Menu page to view and, if desired, export any of the four (4) canned reports that come with the program. These are “My Planting Plan,” “All Plants Detail,” “Outcome Summary,” and “Outcome Detail,” described in further detail in the Reports section below.

Let it Grow Garden Planner

—

□

×

Reports Menu

Exit to Main


Please Select a Report to View

My Planting Plan

All Plants Detail

Outcome Summary

Outcome Detail



Reports

My Planting Plan

A listing of all Planting Occurrences that are currently part of the Planting Plan.

Planting Plan report

My Planting Plan Report

Click "Refresh" to reflect changes:

Refresh

Export (save) to CSV

Close Report Window

Hint: Export to view all data if list exceeds screen size.

Plant	Season	Zone	Plot	Space / Seed Pack	Space / Seedling	Depth for Seeds	Watering Frequency	Days to Harvest
Arugula	Spring24	Long Garden Bed	1	10 inches	6 inches	0.50 inches	1x per week	45
Carrot	Spring24	Long Garden Bed	13	12 inches	3 inches	0.00 inches	2x per week	80
Lettuce	Spring24	Long Garden Bed	14	6 inches	6 inches	1.00 inches	2x per week	30
Basil	Spring24	Wheelbarrow	25	12 inches	18 inches	0.00 inches	1x per week	120
Garlic	Fall24	Long Garden Bed	2	6 inches	6 inches	4.00 inches	1x every other week	180
Garlic	Fall24	Long Garden Bed	2	6 inches	6 inches	4.00 inches	1x every other week	180
Garlic	Fall24	Long Garden Bed	3	6 inches	6 inches	4.00 inches	1x every other week	180
Garlic	Fall24	Long Garden Bed	4	6 inches	6 inches	4.00 inches	1x every other week	180
Lettuce	Fall24	Long Garden Bed	14	6 inches	6 inches	1.00 inches	2x per week	30
Arugula	Spring25	Long Garden Bed	1	10 inches	6 inches	0.50 inches	1x per week	45
Green Bean	Spring25	Long Garden Bed	2	18 inches	8 inches	1.00 inches	2x per week	55
Arugula	Spring25	Long Garden Bed	3	10 inches	6 inches	0.50 inches	1x per week	45
Lettuce	Spring25	Long Garden Bed	14	6 inches	6 inches	1.00 inches	2x per week	30
Garlic	Fall25	Long Garden Bed	1	6 inches	6 inches	4.00 inches	1x every other week	180

All Plants Detail

- A listing of all available Plants in the database which can be selected for consideration in a Planting Plan.

All Plant Details report

All Plants Detail Report

Click "Refresh" to reflect changes:

Refresh

Export (save) to CSV

Close Report Window

Hint: Export to view all data if list exceeds screen size.

Plant	In Plan?	Crop Group	Sun Required	Soil Moisture	Space / Seed Pack	Space / Seedling	Depth Requirement	Watering Frequency	Frost Tolerance	Days to Harvest	Plant in Spring?	Plant in Fall?
Arugula	Yes	Cabbage	6-8 hours	Well Drained	10 inches	6 inches	6 inches	1x per week	Cold Sensitive	45 days	Yes	Yes
Carrot	Yes	Carrot	4-6 hours	Well Drained	12 inches	3 inches	18 inches	2x per week	Cold Sensitive	80 days	Yes	Yes
Basil	Yes	Non-rotation	2-4 hours	Well Drained	12 inches	18 inches	6 inches	1x per week	Cold Sensitive	120 days	Yes	No
Lettuce	Yes	Non-rotation	4-6 hours	Well Drained	6 inches	6 inches	6 inches	2x per week	Frost Tolerant	30 days	Yes	Yes
Garlic	Yes	Onion	6-8 hours	Well Drained	6 inches	6 inches	12 inches	1x every other week	Cold Hardy	180 days	No	Yes
Green Bean	Yes	Pea & Bean	6-8 hours	Well Drained	18 inches	8 inches	8 inches	2x per week	Cold Sensitive	55 days	Yes	No
Beet	No	Beetroot	6-8 hours	Moist	12 inches	2 inches	12 inches	1x per week	Frost Tolerant	60 days	Yes	Yes
Broccoli	No	Cabbage	6-8 hours	Well Drained	20 inches	12 inches	6 inches	2x per week	Frost Tolerant	90 days	Yes	Yes
Cucumber	No	Marrow	6-8 hours	Moist	36 inches	36 inches	12 inches	2x per week	Cold Sensitive	120 days	Yes	No
TestPlant1	No	Marrow	6-8 hours	Extremely Dry	4 inches	3 inches	6 inches	1x every other week	Warm Loving	50 days	Yes	Yes
Zucchini	No	Marrow	6-8 hours	Well Drained	6 inches	4 inches	4 inches	1x per week	Cold Sensitive	60 days	Yes	No
Pepper	No	Nightshade	6-8 hours	Well Drained	18 inches	18 inches	12 inches	1x per week	Warm Loving	120 days	Yes	No
Asparagus	No	Non-rotation	6-8 hours	Well Drained	72 inches	72 inches	36 inches	1x per week	Warm Loving	425 days	Yes	No
TestPlant	No	Non-rotation	4-6 hours	Well Drained	2 inches	1 inches	12 inches	1x every other week	Cold Hardy	None days	Yes	Yes

Outcome Summary Report

- A summary of each Plant that has been included in a Planting Plan, including how many times it's been planted, the success ratio, and information regarding the most recent outcome.

Outcome Summary report

Outcome Summary Report

Close Report Window

Click "Refresh" to reflect changes:

Refresh

Export (save) to CSV

 Hint: Export to view all data if list exceeds screen size.

Plant Name	Times Planted	Success Ratio	Most Recent Season	Most Recent Outcome
Lettuce	31	32.26%	Fall2024	Failure
Garlic	30	0.0%	Fall2024	Failure
Pepper	9	0.0%	Spring2024	Failure
Arugula	11	0.0%	Spring2024	Failure

Outcome Detail Report

- A listing of all documented Outcomes by Planting Occurrence.

Outcome Detail report

Outcome Detail Report

Close Report Window

Click "Refresh" to reflect changes:

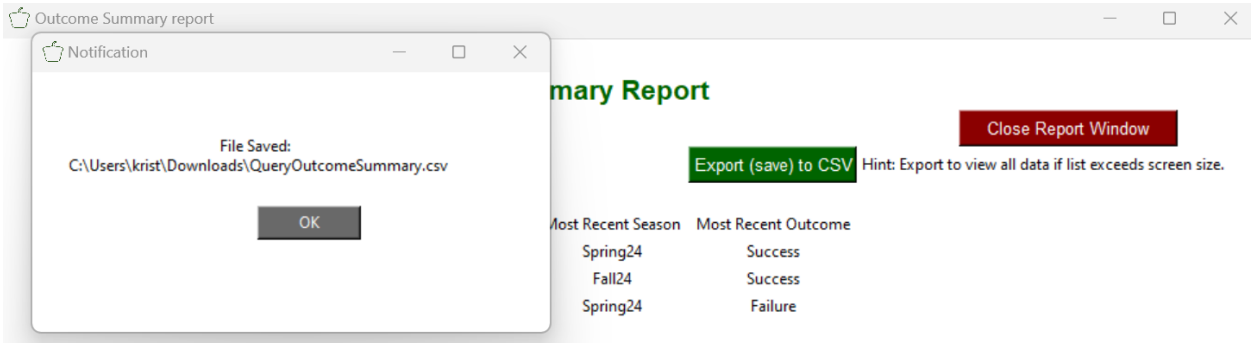
Refresh

Export (save) to CSV

 Hint: Export to view all data if list exceeds screen size.

Plant Set ID	Season	Plant	Zone	Plot	Set Quantity	Set Type	Date Planted	First Harvest	Last Harvest	Outcome
19	Spring2024	Garlic	Long raised bed	13	1	Bulb(s)	None	None	None	None
20	Spring2024	Garlic	Long raised bed	16	1	Seedling(s)	None	None	None	None
21	Spring2024	Garlic	Long raised bed	17	1	Seedling(s)	None	None	None	None
22	Spring2025	Green Bean	Long raised bed	12	1	Seedling(s)	None	None	None	None
24	Spring2025	Green Bean	Long raised bed	13	1	Seedling(s)	None	None	None	None
25	Spring2024	Arugula	Long raised bed	19	1	Seedling(s)	None	None	None	None
26	Fall2024	Garlic	Long raised bed	20	1	Seedling(s)	None	None	None	None
27	Fall2024	Garlic	Long raised bed	24	1	Seedling(s)	None	None	None	None
28	Fall25	Garlic	Long raised bed	16	1	Seedling(s)	None	None	None	None
29	Fall25	Garlic	Long raised bed	17	1	Bulb(s)	None	None	None	None
30	Spring26	Green Bean	Long raised bed	12	1	Seedling(s)	None	None	None	None
31	Fall25	Garlic	Long raised bed	19	1	Bulb(s)	None	None	None	None
32	Spring26	Green Bean	Long raised bed	13	1	Seedling(s)	None	None	None	None
33	Fall2024	Garlic	Long raised bed	19	1	Bulb(s)	None	None	None	None
34	Spring2024	Lettuce	Long raised bed	34	1	Seedling(s)	None	None	None	None
17	Spring2024	Lettuce	White Flowerpot 2	26	1	Seedling(s)	None	None	None	Failure
18	Spring2024	Garlic	Long raised bed	12	1	Bulb(s)	2024-04-01	2024-07-29	None	Failure
7	Spring2024	Arugula	Long raised bed	18	4	Seedling(s)	2024-04-15	None	None	Failure
4	Fall2024	Lettuce	Long raised bed	22	1	Seedling(s)	2024-05-01	2024-07-01	2024-08-01	Success
6	Spring2024	Pepper	Long raised bed	22	2	Seedling(s)	2024-05-01	None	None	Failure
5	Fall2024	Lettuce	Long raised bed	23	2	Seed Pack	2024-08-01	2024-09-01	2024-10-15	Failure
3	Fall2024	Garlic	Long raised bed	21	1	Bulb(s)	2024-11-08	None	None	Failure

If the user chooses to export any of these Reports to a CSV file, they will see a popup message indicating the location of the file. This is set to be the user's Downloads folder.



	A	B	C	D	E	
1	Plant Name	Times Planted	Success Ratio	Most Recent Season	Most Recent Outcome	
2	Lettuce	31	32.25806451612903	Spring24	Success	
3	Garlic	34	0	Fall24	Success	
4	Carrot	2		Spring24	Failure	
5						

System Architecture

Source Code Structure



The following is a summary of the source code directories and their contents:

Code Directory	
Directory	Usage
LetItGrowAllFiles.zip	Contains the CreateNewGardenDatabase.sql script to install the database and the LetItGrowCodeFiles folder described below. This is the zip file that the Developer downloads from GitHub.
LetItGrowCodeFiles	Contains all the Python code for the project and the Images folder described below. This is the folder from which the Windows executable file is created.
Images	Contains the .png files used for the logos, icons, and other images displayed in the interface.













Highlighted rows indicate directories containing source code.

Screenshots of the Live System




LetItGrowAllFiles.zip

<input type="checkbox"/> Name ^	Type
 LetItGrowCodeFiles	File folder
 CreateNewGardenDatabase.sql	Microsoft SQL Server Query File

LetItGrowCodeFiles

<input type="checkbox"/> Name ^	Type
 Images	File folder
 data_connection.py	PY File
 export_query.py	PY File
 main.py	PY File
 my_season.py	PY File
 plan.py	PY File
 plant.py	PY File
 plant_set.py	PY File
 planting_year.py	PY File
 plot.py	PY File
 validate.py	PY File
 window.py	PY File

Images

<input type="checkbox"/> Name ^	Type
 Icon.png	PNG File
 Logo.png	PNG File
 WelcomeLogo.png	PNG File

Executables

EXE LetItGrow (LetItGrow.exe)

The developer creates the executable by using the following commands in the Command Prompt terminal from within the LetItGrowCodeFiles folder:

```
pip install pyinstaller
```

```
pyinstaller --onefile --noconsole --hidden-import babel.numbers --add-data Images\*.png;.\\Images" main.py
```

The developer then renames the Main.exe file to LetItGrow.exe for end user clarity.

EXE CreateNewGardenDatabase (CreateNewGardenDatabase.sql)

The developer creates the SQL back end “Garden” database by executing this SQL script in SQL Server Manager.

This script will set up the complete database, including the loading of seed data for dropdowns and a default list of Plants to get the user started.

Code Architecture

The Let it Grow Garden Planner utilizes a set of Python files, some of which interact with a SQL database installed under the same user profile as the main program.

The main file starts up the program and calls the window file, which manages the interface and keeps running until the user exits the program.

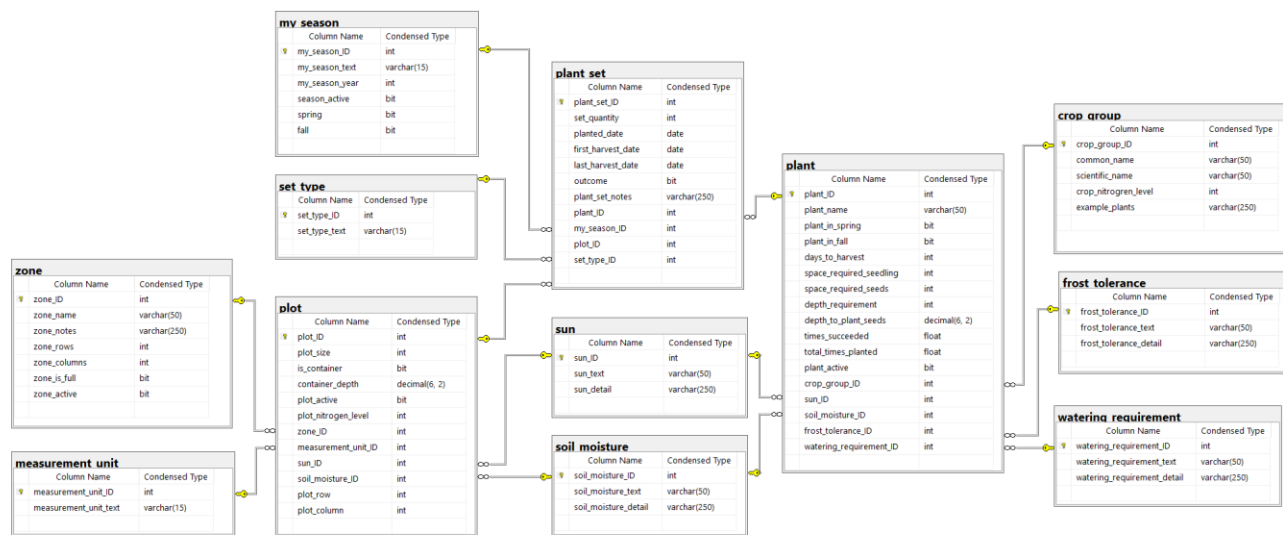
The plant, plant_set, plot, my_season, and planting_year files are all used to generate objects used to set up the Garden and manage the Garden Plan.

The plan file runs the algorithms used to generate and update the Garden Plan by performing a series of plot suitability checks.

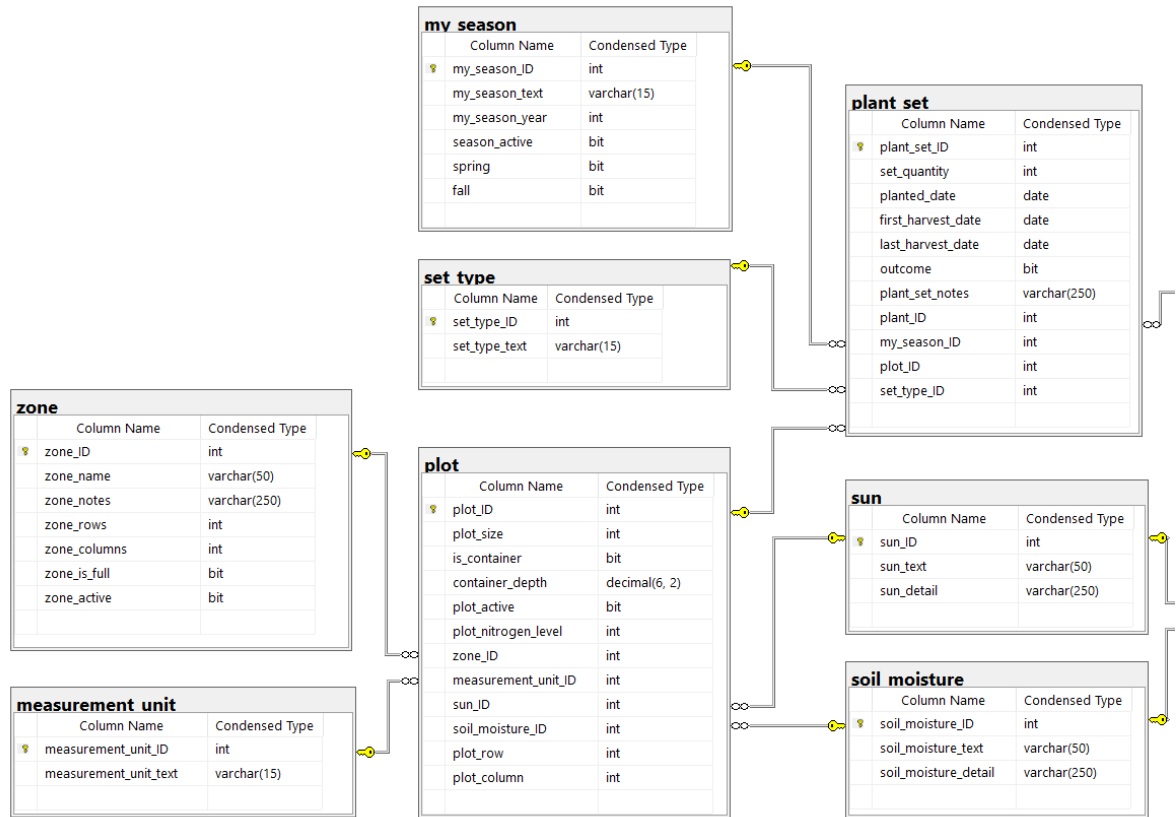
The data_connection, export_query, and validate files all manage objects used to perform generic actions that are repeated throughout the program.

Database or Data Store

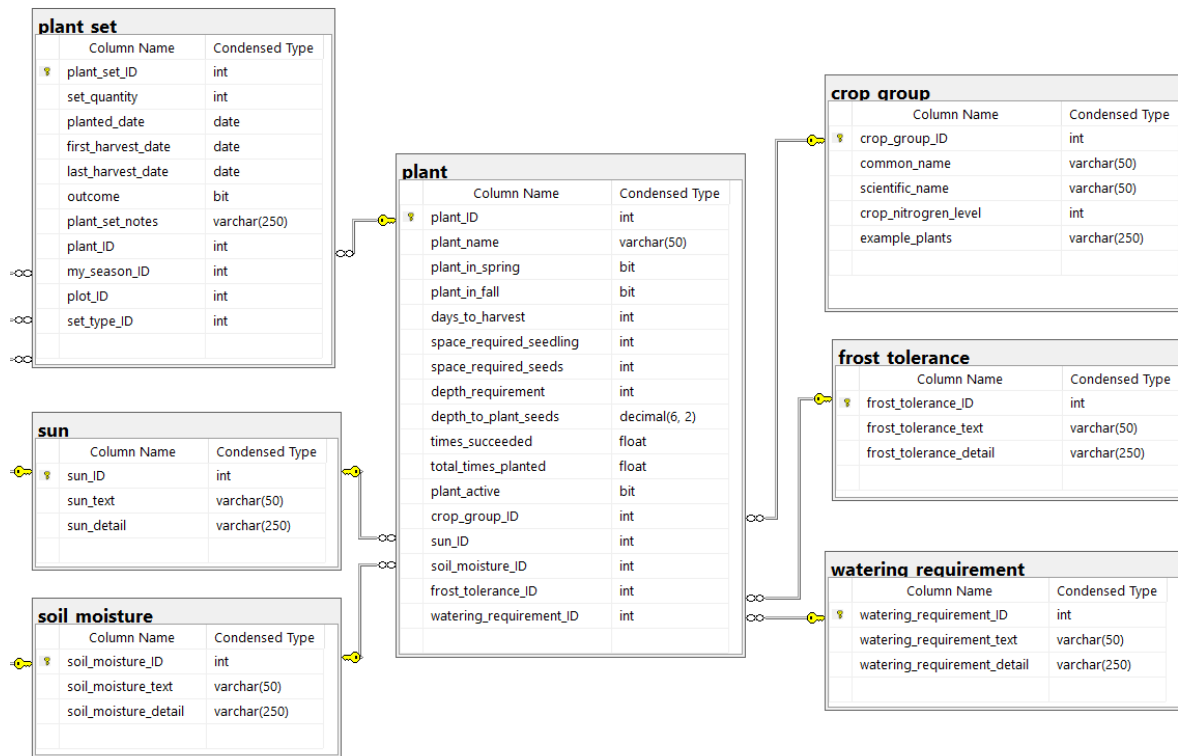
Below is the Full UML Diagram of the SQL Express database that is utilized for this project.



Zoomed in to highlight relationships between Zones, Plots, Plant Sets, Seasons, and Plot qualities:



Zoomed in to highlight relationships between Plant Sets, Plants, Crop Groups, and Plant Requirements:



The my_season table represents a planting season (i.e., Spring2024, Fall2024, Spring2025).

A plant_set is a single occurrence of planting a species of plant, which can refer to one or more individual plants or seeds. This table tracks what was planted and the outcome of that occurrence. The set_type refers to whether the plant_set contains seedling(s), seeds, or bulbs.

The plant table contains information about each species of plant that could be planted. It will be linked to a crop_group table that represents details that apply to a set of similar plants. The crop_group and other linked tables for fields that don't change are set up to be used with dropdown menus.

Times_succeeded and total_times_planted will both be calculated fields that update automatically. Possible outcomes will be Success and Failure.

The garden is broken out into Plots and Zones. A plot is the smallest unit of space in which a plant_set can be planted. A zone is made up of contiguous plots and is strictly used for ease of identifying planting locations. Options for "Container" will be yes or no. If yes, then container_depth is required.

Stored Procedures and User Defined Functions

Stored Procedures

- AddPlant: Used to add a new plant to the master list.
- AddPlantSet: Used to add a new planting occurrence.
- AddPlot: Used to add a new plot to the Garden.
- AddSeason: Used to add a new planting season.
- AddZone: Used to add a new zone, which is a contiguous set of plots.
- CompleteYearPlantset: Updates plot nitrogen levels based on the crop that was planted there.
- IncrementPlotNitrogen: Increments the nitrogen level of all plots by 1 after a season is completed.
- QueryAllPlantsSetupDetail: Used for All Plants Detail report.
- QueryLastSeasonPlot: Used in Garden Plan to check what was planted in and the nitrogen level of a specific plot during the prior season.
- QueryMySeasonData: Used in Garden Plan to check if the current season is spring or fall, to then compare against seasons in which a specific plant can be planted.
- QueryOutcomeDetail: Used for Outcome Detail report.
- QueryOutcomeSummary: Used for Outcome Summary report.
- QueryPlantingPlan: Used in Planting Plan report and other areas where plan details are needed.
- QueryPlantSetData: Used to retrieve data about specific planting occurrences, such as to populate fields in Edit Plant Set page.
- QueryPlotGrid: Used to create the Garden grid on View Garden Plan page.
- QuerySinglePlotData: Used to retrieve current data about a specific Plot ID to display onscreen in in Edit Plot window.
- QueryYearData: Used to retrieve a list of all Plants planted during a calendar year along with the applicable details to help confirm the closeout of a planting year.
- RetrieveCropGroupData: Populates dropdown menu.
- RetrieveFrostToleranceData: Populates dropdown menu.
- RetrieveMeasurementUnitData: Populates dropdown menu.
- RetrieveMySeasonData: Populates dropdown menu.
- RetrievePlantNames: Populates dropdown menu.
- RetrievePlantRequirements: Used to get plant requirements to compare against plot qualities during location suitability check when executing the Garden Plan.
- RetrievePlotData: Used to retrieve a listing of all active plots and their qualities, which is then converted into a list of Plot objects for use in multiple functions throughout the program. Two primary usages are comparing against plant requirements during location suitability check when executing the Garden Plan and retrieving Plot details to display on the Edit Plot window.
- RetrieveSetTypes: Populates dropdown menu.
- RetrieveSoilMoistureData: Populates dropdown menu.
- RetrieveSunData: Populates dropdown menu.
- RetrieveWateringRequirementData: Populates dropdown menu.
- RetrieveZoneData: Populates dropdown menu.
- UpdateMySeason: Created to edit Seasons; not currently being used but saving for future development.
- UpdatePlantSet: Updates planting occurrence details after edited.
- UpdatePlot: Updates planting occurrence details after edited.
- UpdateZone: Created to edit Zones; not currently being used but saving for future development.

Used Defined Functions

- `get_last_season_id`: Calculates and returns the season ID that occurred prior to the one input, used in `QueryLastSeasonPlot` stored procedure.

External Files & Data

There are three external image files that are used in this program, which are all included in the Images folder within the zip file:

- Icon.png is the icon that appears in the top left corner of the window and the taskbar.
- Logo.png is the full Let it Grow logo, which is displayed on the Reports and Garden Setup & Maintenance windows.
- WelcomeLogo.png is the welcome page logo, which is the full Let it Grow logo with the “Get ready to...” intro text included.

The following entries should also be added to the developer and user’s PATH Environment Variables when the additional Python libraries are installed:

C:\Users\[user folder name]\AppData\Local\Programs\Python\Python312\Lib\site-packages\babel

C:\Users\[user folder name]\AppData\Local\Programs\Python\Python312\Lib\site-packages\Babel-2.15.0.dist-info

C:\Users\[user folder name]\AppData\Local\Programs\Python\Python312\Lib\site-packages\tkcalendar

C:\Users\[user folder name]\AppData\Local\Programs\Python\Python312\Lib\site-packages\tkcalendar-1.6.1.dist-info

To do this, go to System Properties > Advanced > Environment Variables.

Select the “path” entry, then click “Edit.”

Click “New” to add each entry.

Programming Language | Python

This project is written in the Python programming language for the back end and utilizes a SQL Express database for persistent data storage.

Imported Python libraries are pyodbc for the SQL data connection, tkinter for the interface, tkcalendar for date entry handling, and Babel to support tkcalendar.

Project Classes

Main | main.py

Driver class that launches the program.

Connection | data_connection.py

Class manages the connection to a SQL database.

ExportQuery | export_query.py

Class exports data retrieved from SQL queries in the form of a stored procedure to a CSV file.

Validate | validate.py

Class performs all generic data validation for text, integer, float, and date inputs.

Window | Window.py

Class to manage all interface windows, which are the class names ending in “page” below.

All primary windows are generated when this class is called upon startup, and then raised as frames when selected from the Main Menu or Garden Setup & Maintenance Menus.

Frames for report and notification windows are both generated and raised at the time the report selection is made or when the program state prompts a notification.

DropDown | included with Window.py

Class to manage all dropdown lists.

Connects to a SQL database and executes a stored procedure to retrieve values list values.

These values are stored in a Python list, which can then be used to retrieve the value selected or the ID associated with the value as needed.

StartPage | included with Window.py

Class manages the interface and options associated with the startup window / main menu.

AddPlantPage | included with Window.py

Class manages the interface, options, and functions associated with adding a new Plant to the master list.

GardenPlanPage | included with Window.py

Class manages the interface, options, and interactive functions associated with setting up a Garden Plan.

EditSetPage | included with Window.py

Class manages the interface, options, and interactive functions associated with editing or updating a planting occurrence.

SetupPage | included with Window.py

Class manages the interface and options associated with the Garden Setup & Maintenance Menu.

AddZonesPage | included with Window.py

Class manages the interface, options, and interactive functions associated with adding a new garden Zone.

AddPlotsPage | included with Window.py

Class manages the interface, options, and interactive functions associated with adding new Plots.

EditPlotPage | included with Window.py

Class manages the interface, options, and interactive functions associated with editing individual Plots.

AddSeasonPage | included with Window.py

Class manages the interface, options, and interactive functions associated with adding a new planting Season.

DisplayPlan | included with Window.py

Class manages the interface, options, and interactive functions associated with displaying the current Garden Plan for a selected Season in a grid format.

ReportsMenu | included with Window.py

Class manages the interface and options associated with the Reports Menu.

PlantingPlanReport | included with Window.py

Class manages the interface, options, and interactive functions associated with displaying the Planting Plan Report in a new window.

PlantsDetailReport | included with Window.py

Class manages the interface, options, and interactive functions associated with displaying the Plant Detail Report in a new window.

OutcomeDetailReport | included with Window.py

Class manages the interface, options, and interactive functions associated with displaying the Outcome Detail Report in a new window.

OutcomeSummaryReport | included with Window.py

Class manages the interface, options, and interactive functions associated with displaying the Outcome Summary Report in a new window.

CompleteYearPage | included with Window.py

Class manages the interface, options, and interactive functions associated with completing a planting calendar year.

Plot | plot.py

Class handles the back end management of Plot objects, including transmittal of data to and from SQL.

MySeason | my_season.py

Class handles the back end management of MySeason objects, including generating the Season Text value and transmittal of data to and from SQL.

Plant | plant.py

Class handles the back end management of Plant objects, including transmittal of data to and from SQL as well as setting the requirements for each Plant for use with the Garden Plan.

PlantSet | `plant_set.py`

Class handles the back end management of Plant Set objects, including transmittal of data to and from SQL.

Plan | `plan.py`

Class runs the algorithms associated with creating a Garden Plan using a Plan object.

This includes functions to assign a list of Plant Sets to Plots automatically or assign Plant Sets to specifically selected Plots manually.

Either way, the algorithm will compare Plot qualities to the master Plant requirements for suitability before finalizing the Plot assignment.

Qualities include sun, soil moisture, and current nitrogen levels, as well as what was planted previously, and anticipated nitrogen levels based on the current plan.

Each Plant either adds or removes nitrogen based on an integer value assigned to its crop group, plus the nitrogen level of each Plot is automatically incremented by 1 after each season.

PlantingYear | `planting_year.py`

Class contains functions that run the algorithms associated with updating the Garden database at the end of a planting calendar year.

This includes updating the nitrogen levels of Plots in the database based on the Plant that was planted each Season that year, plus the automatic increment of 1 each season. It also includes changing the status of those Seasons to inactive.

Project Modules

os | Built-In Python

This built-in module is used to locate user's file path to create the directory location of images.

sys | Built-In Python

This built-in module is used to close the program when the Exit button is selected from the Main Menu.

datetime | Built-In Python

This built-in module is used for date formatting and validation.

re | Built-In Python

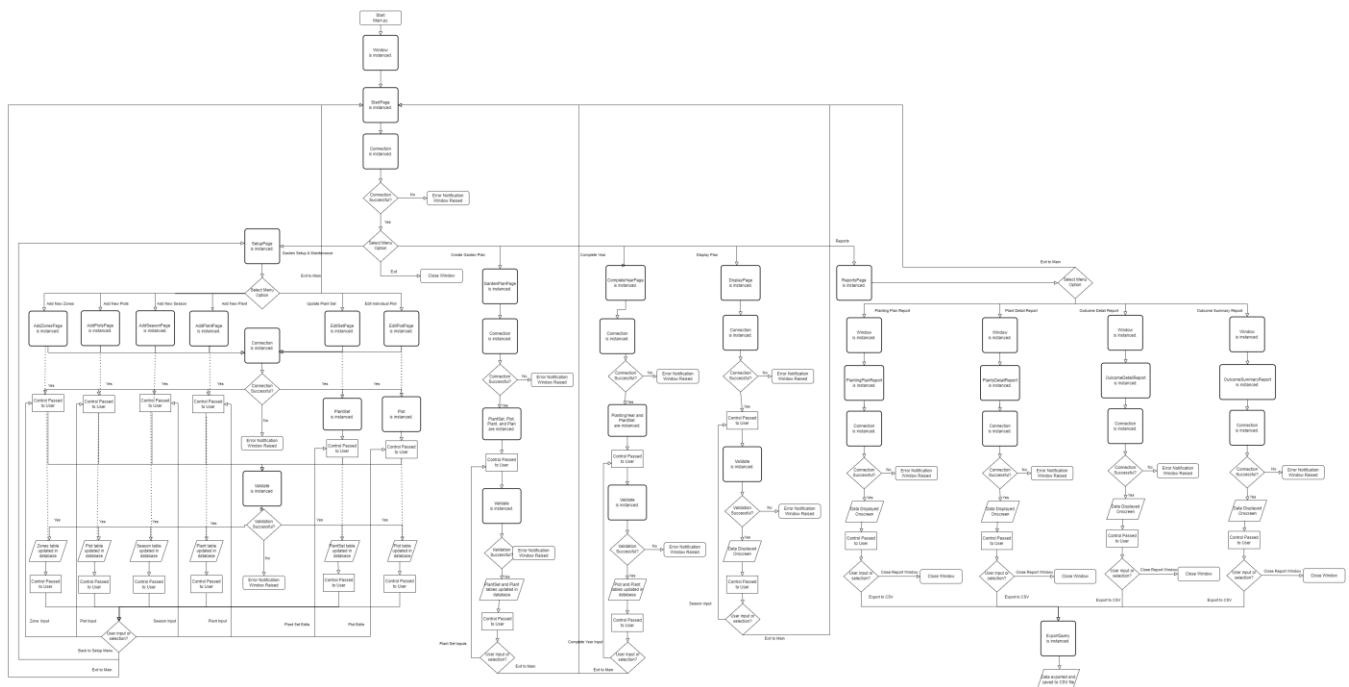
This built-in module is used for text validation via regular expressions (regex).

csv | Built-In Python

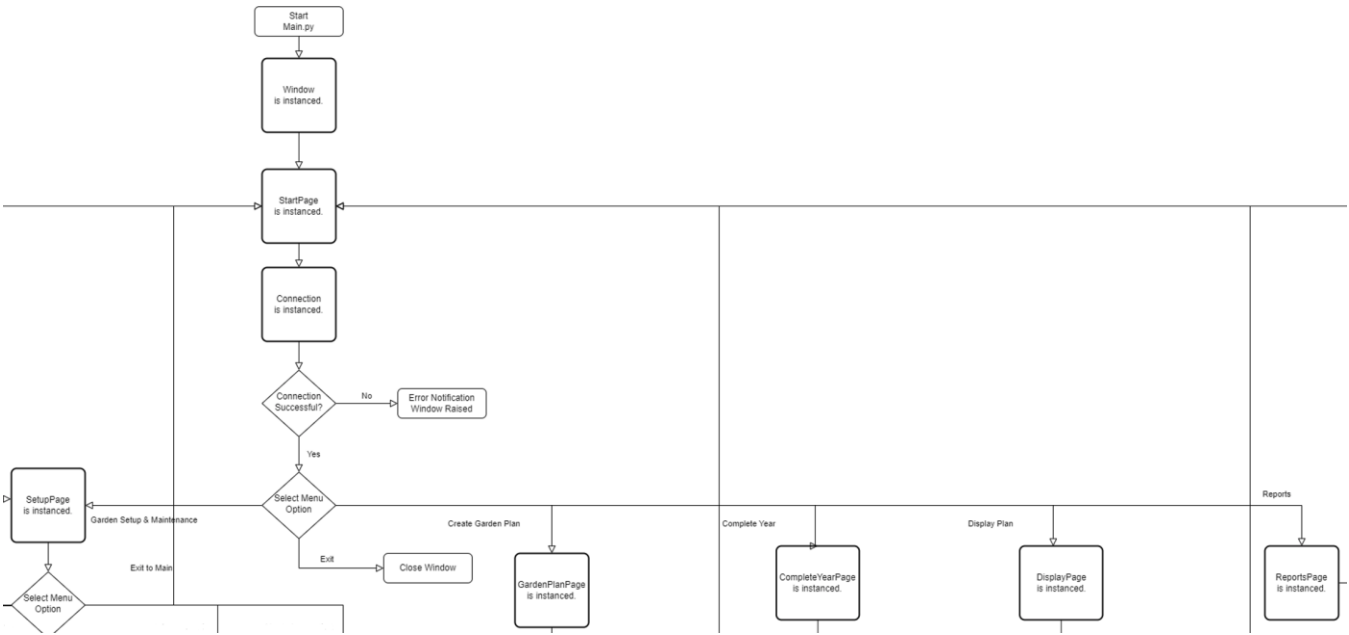
This built-in module is used for exporting data to a .csv file.

Most options will open in the same window, aside from notifications and reports which open new Window instances. The new windows can be closed by clicking “ok” (for notification) or “Close Report Window” (for reports), which only close that Window instance but not the main Window.

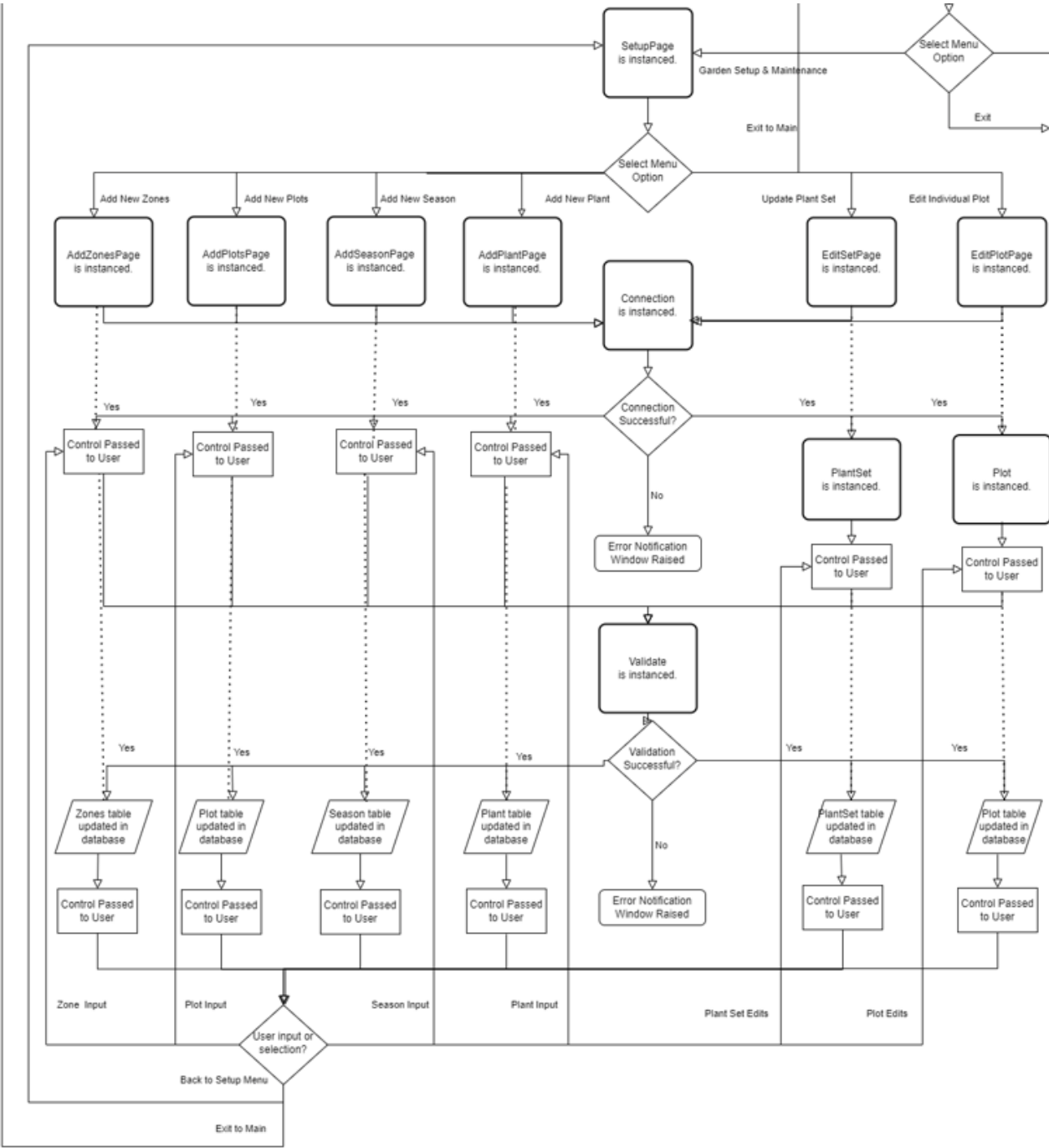
Full Program Diagram



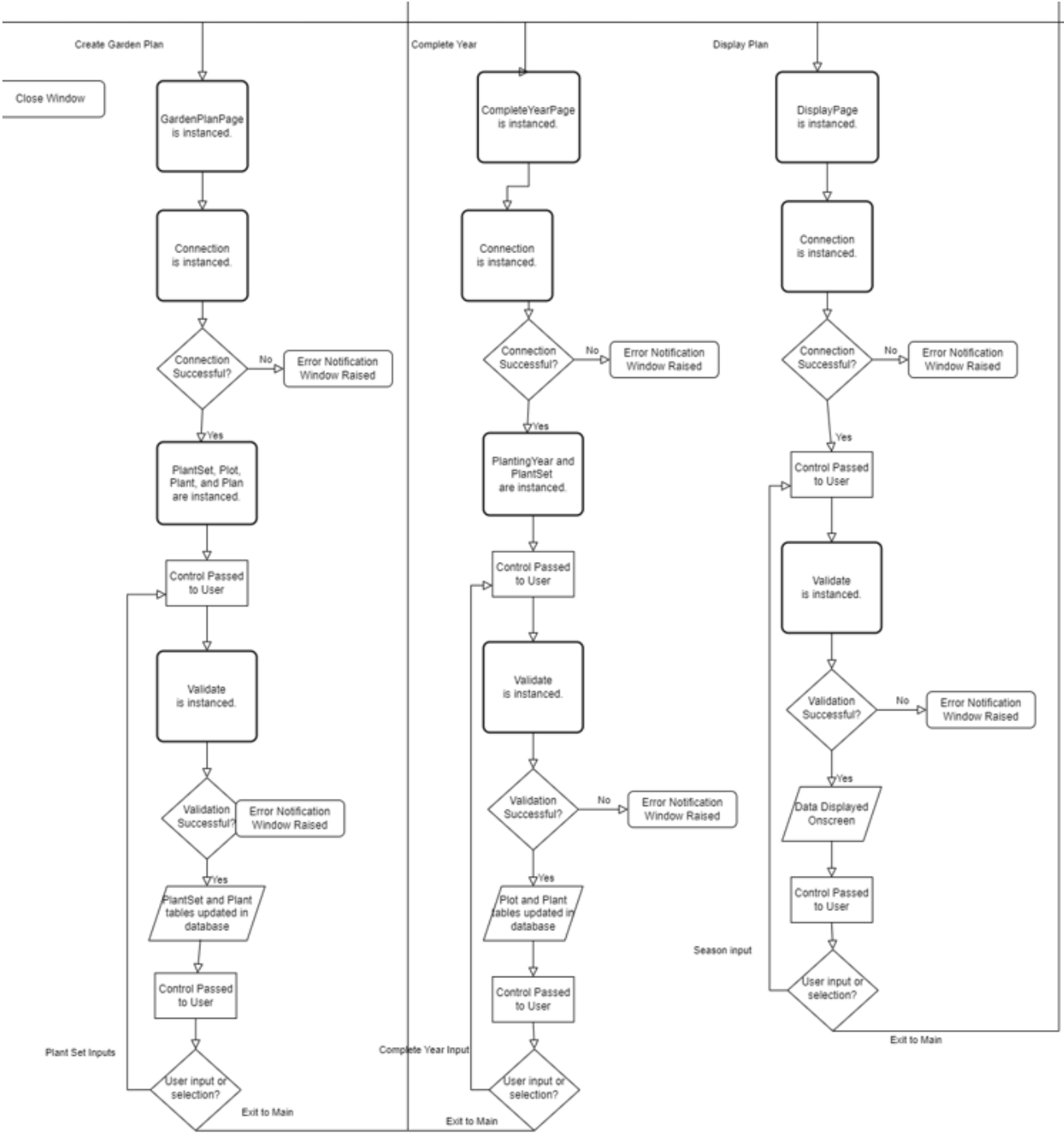
Zoom in on Main Menu



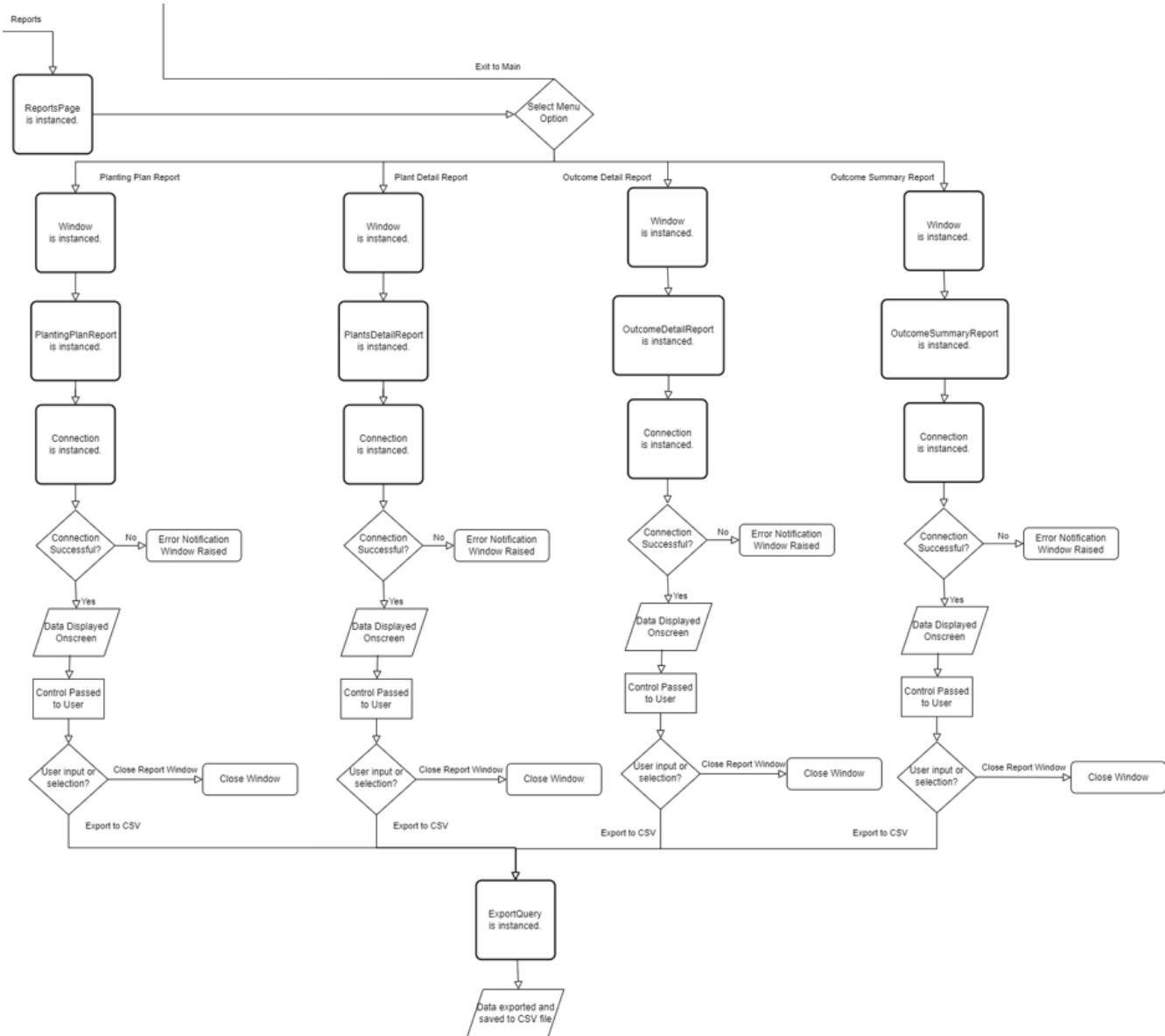
Zoom in on Garden Setup & Maintenance Menu Options



Zoom in on Primary Functions



Zoom in on Reports Menu Options



Summary

Let it Grow Garden Planner is a local desktop application that utilizes a Python codebase and is connected to a SQL database for persistent data storage. The connection to SQL is managed by the pyodbc Python module and the user interface is created using the tkinter Python package. All interactions between SQL and Python are handled by executing SQL stored procedures. Python classes include those to manage the interface, SQL connection, data validation and exports. Also included are classes to manage specific objects related to the Garden Plan, which are Plants, Plant Sets, Plots, Planting Years, and Seasons.

The project code is all kept in one folder, “LetItGrowCodeFiles” with an Images subfolder. This is the folder from which the Windows executable is created. The LetItGrowCodeFiles folder is packaged along with the SQL script to create a new instance of the Garden database in a zip file. The zip file can be downloaded from GitHub here: <https://github.com/kristyyt/LetItGrowGardenPlanner/blob/main/LetItGrowAllFiles.zip>

This concludes the outline of the code and system configuration used to develop the Let it Grow Garden Planner application. Additional information regarding the testing, deployment, maintenance of the project, along with setup instructions is available in the Appendices below.

APPENDIX A (TESTING PLAN)

The Let it Grow Garden Planner application was unit tested at the end of each iteration throughout development. SQL stored procedures were first run as queries in SQL Server Manager before being integrated with the Python classes. The Python classes were tested two ways. The first and earliest method was inserting test code which printed specific variable values to the terminal to confirm they were as expected. Then, once the user interface was built, it was utilized to enter test data and verify the interface behaved as expected.

Once the project was completely built, two full days were devoted to testing the full product usage cycle. This included installing the database, setting up the Garden, creating a Garden Plan, and completing a Planting Year. Major bugs that would impact the other scheduled tests were fixed immediately, while others were added to a list to go back to following the testing session. Third-party feedback was also requested on specific areas regarding the design, some of which was incorporated into the final program.

When all the fixes that time allowed had been implanted, a final test was completed by walking through each user scenario again.

APPENDIX B (BUILD AND RELEASE PROCESS)

Let it Grow Garden Planner

Build and Release Process

Deployment Plan

The initial release of the Let it Grow Garden Planner application will be deployed as a beta version to a group of users who have expressed interest in the project. Subsequent releases based on the feedback from the pilot group will be released using an A / B Testing deployment method. Once a version is deemed satisfactory for general release, any additional updates will first be released to a pilot group.

Installation on the client system will be performed by the developer for the foreseeable future, with input from the client as to where the executable file is to be stored and other system specifics as needed. The client will only be responsible for configuring the Garden within the application after it's installed.

Maintenance Plan

Regular maintenance of the Let it Grow Garden Planner application is recommended to improve program functionality and efficiency.

Updates will initially be focused on adding functionality that was omitted from the initial release to adhere to time constraints, which includes the ability to edit Plants, Zones and Seasons.

The next phase will be to improve the planning algorithm to take multiple plots into consideration, so that larger plants could take up more than one plot.

Updates based on user feedback may occur before or after those listed above, depending on priority.

Once all planned enhancements have been implemented, the code should still be reviewed periodically to see if it can be refactored to increase efficiency and reduce redundancy.

Any new release will also include bug fixes and small interface changes to improve the end user experience.

APPENDIX C (CLIENT SETUP INSTRUCTIONS)

Let it Grow Garden Planner

Client Setup Instructions

Complete the following steps to start using the Let it Grow Garden Planner software for the first time.

1. Double click on the LetItGrow.exe file to open the application.
2. Click the “**Garden Setup & Maintenance**” button.

This is where you will identify and configure your Garden, as well as generate your initial list of seasons to plan.

3. Click “**Add New Zone**” to identify at least one Zone, which will consist of one or more contiguous plots.
 - A Zone could be a garden bed, flowerpot, etc.
 - Each Zone should have a unique name for easier identification purposes.

Enter the information about your Zone, then click “Add Zone.”

To add another Zone, type over the existing data and click “Add Zone” again.

When done, click “Back to Setup Menu.”

The screenshot shows a window titled "Add New Zone". At the top right are two buttons: "Back to Setup Menu" and "Exit to Main". Below these, there is explanatory text: "A 'plot' is a space where a single plant or plant set can be planted." and "A 'Zone' is a group of contiguous plots, such as a designated garden area or raised bed". A text input field is labeled "Please enter a unique name for your zone:" with an example "Ex. 'Large Garden, Blue Flowerpot 1'". Below this are two small input fields for "Number of rows of plots in this zone:" and "Number of columns of plots in this zone:". At the bottom left is a text area labeled "Zone Notes (Optional):". At the bottom right is a green button labeled "Add Zone".

4. Click **“Add New Plots”** to add one or more new Plots that share the same qualities to the database.

- A “Plot” is a square unit within a zone on which one or more plants can be planted.
- The information entered here will be used to determine location suitability for the planting plan.
- Row and column numbers are auto-assigned based on the number selected and those already existing.

Enter the information, then click “Add Plots.”

To add another set of Plots, type over the existing data and click “Add Plots” again.

When done, click “Back to Setup Menu.”

Add New Plots

Please select a Zone:

Number of rows of identical plots to add:

Number of columns of identical plots to add:

Size of each Plot: Unit of Measurement:

☐ Is this plot a container (solid bottom)?

If container, container depth in inches:

Sun Level of Plot(s): Soil Moisture Level of Plot(s):

Click Refresh if new zones have been added:

5. Click **“Add New Season”** to add a new planting Season to the database.

- The system “Season” name is auto-generated based on season and year selected.
- All active Seasons are included in the garden plan.

Select whether the season you’re adding starts in spring or fall, enter the year, then click “Generate My Season.”

To add another Season, type over the existing data and click “Generate My Season” again.

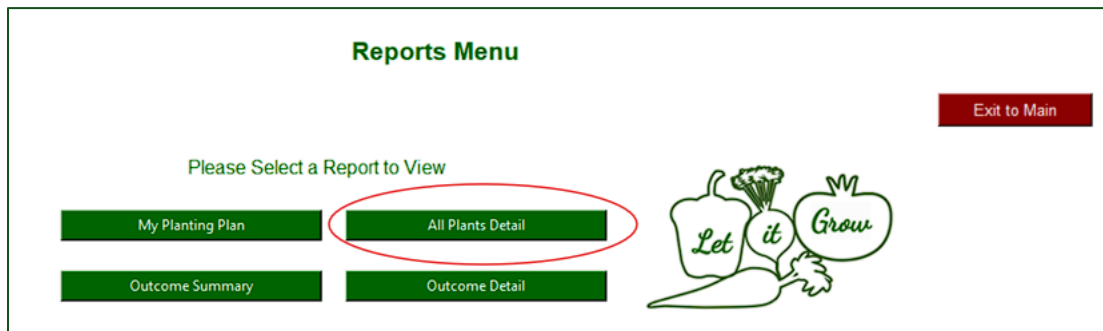
When done, click “Exit to Main.”

Add New Season

Select Season: ☐ Spring ☐ Fall

Enter Year:

6. To check which Plants are already in the database by default, click “Reports” followed by “All Plants Detail.”



7. To add your own Plants to the list...

From the Main Menu, go back to Garden Setup & Maintenance.

Click “**Add New Plants.**”

Enter the information about the Plant, then click “Add Plant.”

To add another Plant, type over the existing data and click “Add Plant” again.

When done, click “Exit to Main.”

The screenshot shows a web form titled "Add New Plant". At the top right is a red button labeled "Exit to Main". The form contains several input fields and checkboxes. "Plant Name:" is a text box with "Zucchini" entered. There are two checkboxes: "Plant in Spring" (checked) and "Plant in Fall" (unchecked). "Crop Rotation Group:" is a dropdown menu with "Marrow" selected. "Frost Tolerance:" is a dropdown menu with "Cold Sensitive" selected. "Sun Required:" is a dropdown menu with "Full: 6-8 hours" selected. "Soil Moisture Required:" is a dropdown menu with "Well Drained: 21-40%" selected. There are four numeric input fields: "Space per Seedling, in inches:" (4), "Space per Seed Pack, in inches:" (6), "Depth Required per Plant, in inches:" (4), and "Depth to Plant Seeds, in inches, will round to 2 decimal places (optional):" (1). "Days to Harvest (optional):" is a numeric input field with 60. "Watering Frequency:" is a dropdown menu with "Medium: 1x per week" selected. At the bottom right is a green button labeled "Add Plant".

8. This completes the initial application setup.

Continue reading if / when you are ready to start your first Garden Plan.

9. If / when you are ready to start your first Garden Plan, click “**Create Garden Plan**” from the Main Menu.

To use the auto-plan feature, complete only the top section for each set of plants you wish to grow, clicking “**Add Plant to Auto-Plan**” after each selection.

When you have finished selecting all the Plants, click “**Complete Auto-Plan**” to run assign plants to appropriate plots automatically.

The screenshot shows the 'Create Garden Plan' interface. At the top, there's a title 'Create Garden Plan' and a red 'Exit to Main' button. Below the title, a green instruction says 'Please complete the following for all Garden Plans:'. The form is divided into two sections. The top section for auto-planning includes dropdowns for 'Select Plant Name' (showing 'Select Value'), 'Set Type' (showing 'Select Value'), and a 'Quantity' input field. A 'Refresh' button is next to the plant name dropdown. To the right of these fields are two green buttons: 'Add Plant to Auto-Plan' and 'Complete Auto-Plan'. The bottom section, titled 'For Manual Plans Only', includes a 'Select your Season' dropdown (showing 'Select Value'), a text prompt 'Enter a Plot ID if known, or enter a Zone, Row and Column', and input fields for 'Plot ID Number' (showing '0'), 'Zone Name' (showing 'Select Value'), 'Row Number' (showing '0'), and 'Column Number' (showing '0'). A green 'Check Plot / Instant Add' button is positioned to the right of these fields.

To assign Plants to specific Plots manually, fill out the entire page, then click “**Check Plot / Instant Add.**”

With either method, you will see a confirmation message listing Plants that were added to the Garden Plan successfully, or an error message no Plants were added.

This screenshot shows the 'Create Garden Plan' form with a success notification dialog box open. The notification message reads: 'The following plants have been assigned to plots: Carrot, Garlic, Lettuce. If any are not listed, no suitable plot was found.' with an 'OK' button. The form fields are the same as in the previous screenshot, but the 'Add Plant to Auto-Plan' button is now disabled.

This screenshot shows the 'Create Garden Plan' form with an error notification dialog box open. The notification message reads: 'Failed checks, Pepper not added to plan' with an 'OK' button. In the form, the 'Select Plant Name' dropdown now shows 'Pepper', and the 'Set Type' dropdown shows 'Seedling(s)'. The 'Add Plant to Auto-Plan' button is disabled, and the 'Check Plot / Instant Add' button is highlighted in green.

10. Throughout the year, click “**Update Plant Set**” from the Garden Setup & Maintenance Menu to enter the planting and harvest dates and the outcome.

- Click “**Check and Save Changes**” to perform the plot suitability check before saving changes, such as if changing the Plot ID, Set Type, and Quantity.
- Alternatively, click “**Save without Checking**” to skip the checks, such as if they’re unnecessary or to force a save change into an otherwise ineligible plot.

Update Plant Set

Select Plant Name: Season:

Plant Set ID: Set Type:

Plot ID Number: Quantity:

Plant Set Notes (Optional):

Date Planted:

First Harvest Date:

Last Harvest Date:

Outcome: ☐ Pending ☒ Success ☐ Failure

11. At the end of the last growing in a calendar year, click “**Complete Year**” from the Main Menu.

- This will update the nitrogen levels of plots in the database, which is one of the ways the algorithm determines plot suitability.
- It will also mark the Seasons as inactive so they will not be included in future Garden Plans.

Enter the year to close out and click “Go” to see the list of Plant Sets that will be impacted.

If the list looks correct, click “Complete” to run the updates.

You can click “Go” again to confirm the Season Active status has been updated to No.

Complete Year

Select Year to Finalize:

Season	Plant Set ID	Plant	Zone	Plot Number	Date Planted	Last Harvest	Outcome	Season Active
Spring24	2	Carrot	Long Garden Bed	13	2024-06-01	2024-09-13	Success	Yes
Spring24	4	Lettuce	Long Garden Bed	14	2024-05-10	2024-07-22	Success	Yes
Fall24	1	Garlic	Long Garden Bed	1	2024-09-27	None	Failure	Yes
Fall24	3	Garlic	Long Garden Bed	2	2024-09-27	2025-07-01	Failure	Yes
Fall24	5	Garlic	Long Garden Bed	3	2024-09-27	2025-08-29	Success	Yes
Fall24	6	Garlic	Long Garden Bed	4	2024-11-01	2025-07-28	Success	Yes

12. This completes the steps required to complete the full Garden Planning cycle.

APPENDIX D (DEVELOPER SETUP INSTRUCTIONS)

Let it Grow Garden Planner

Developer Setup Instructions

Complete the following steps to set up the Let it Grow Garden Planner software on a client system.

1. Install SQL Server 2022 Express on the target system if not already present:
<https://www.microsoft.com/en-us/sql-server/sql-server-downloads>
2. Download the LetItGrowAllFiles zip file from the GitHub repository and extract its contents:
<https://github.com/kristyvt/LetItGrowGardenPlanner/blob/859460c0d70424c9d78782aa787a7902db5f58c6/LetItGrowAllFiles.zip>
3. Open the CreateNewGardenDatabase script in SQL Server Manager or another similar tool. Confirm the “Filename” matches the client’s system configuration. If not, edit accordingly.

```

CreateNewGardenDa...LAPTOP\krist (54)  X
1  /***** Script to Create New Garden Database *****/
2  /***** Part of the Let it Grow Garden Planner *****/
3  /***** Kristy Stark SDEV-435-81 *****/
4  /***** Last Revised 7/18/2024 *****/
5
6  USE [master]
7  GO
8  CREATE DATABASE [Garden]
9      CONTAINMENT = NONE
10     ON PRIMARY
11         ( NAME = N'Garden',
12           FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL16.SQLEXPRESS\MSSQL\DATA\Garden.mdf' ,
13           SIZE = 73728KB ,
14           MAXSIZE = UNLIMITED,
15           FILEGROWTH = 65536KB )
16     LOG ON
17         ( NAME = N'Garden_log',
18           FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL16.SQLEXPRESS\MSSQL\DATA\Garden_log.ldf' ,
19           SIZE = 8192KB ,
20           MAXSIZE = 2048GB ,
21           FILEGROWTH = 65536KB )
22     WITH
23         CATALOG_COLLATION = DATABASE_DEFAULT,
24         LEDGER = OFF
25  GO
  
```

4. With the master system database selected, run the CreateNewGardenDatabase.sql script to create the new Garden database.
5. Install the following Python packages either via the Command Prompt or by searching for them under Python Packages if using PyCharm or another IDE with that option:

Package Name	Use
pip	Installer for other Python packages
pyodbc	Handles SQL data connection
tkinter	Handles all interface windows, frames and basic widgets
tkcalendar	Handles tkinter calendar objects
Babel	Required for tkcalendar to function properly

6. Open the data_connection.py file from the LetItGrowCodeFiles folder in a text editor or IDE.
7. Edit the “server” variable to match the client system server name. Verify the driver info in the connection string is correct, edit if needed.

```
class Connection:
    def __init__(self):
        server = 'KRISTY-LAPTOP\\SQLEXPRESS'
        database = 'Garden'
        self.status = 'pending'

        try:

            connection_string = \
                (f'DRIVER={{ODBC Driver 18 for SQL Server}};'
                 f'SERVER={server};'
                 f'DATABASE={database};'
                 f'trusted_connection=yes;'
                 f'Encrypt=No')
```

8. Verify that the client has a Downloads folder present. If not, open the export_query.py file and edit the name of the folder named in the “download_path” variable to the alternative.

```
class ExportQuery:
    def __init__(self):
        self.header = []

    def export_csv(self, query_name, header):
        this_connection = data_connection.Connection()
        cursor = this_connection.connection.cursor()

        download_path = os.path.expanduser("~") + '\\Downloads\\' + query_name + '.csv'
```

9. In the Command Prompt navigate to the LetItGrowCodeFiles folder, then type the following to install pyinstaller: `pip install pyinstaller`
10. Still in the Command Prompt and in the LetItGrowCodeFiles folder, use pyinstaller to create the Windows executable file by typing the following:


```
pyinstaller --onefile --noconsole --hidden-import babel.numbers --add-data Images\*.png;.\\Images" main.py
```
11. Navigate to the newly created “dist” folder, and rename: “main.exe” to “LetItGrow.exe”
12. Copy the LetItGrow.exe file to the client desktop or other convenient location.