Landscape App Overview

By: Zach Murray

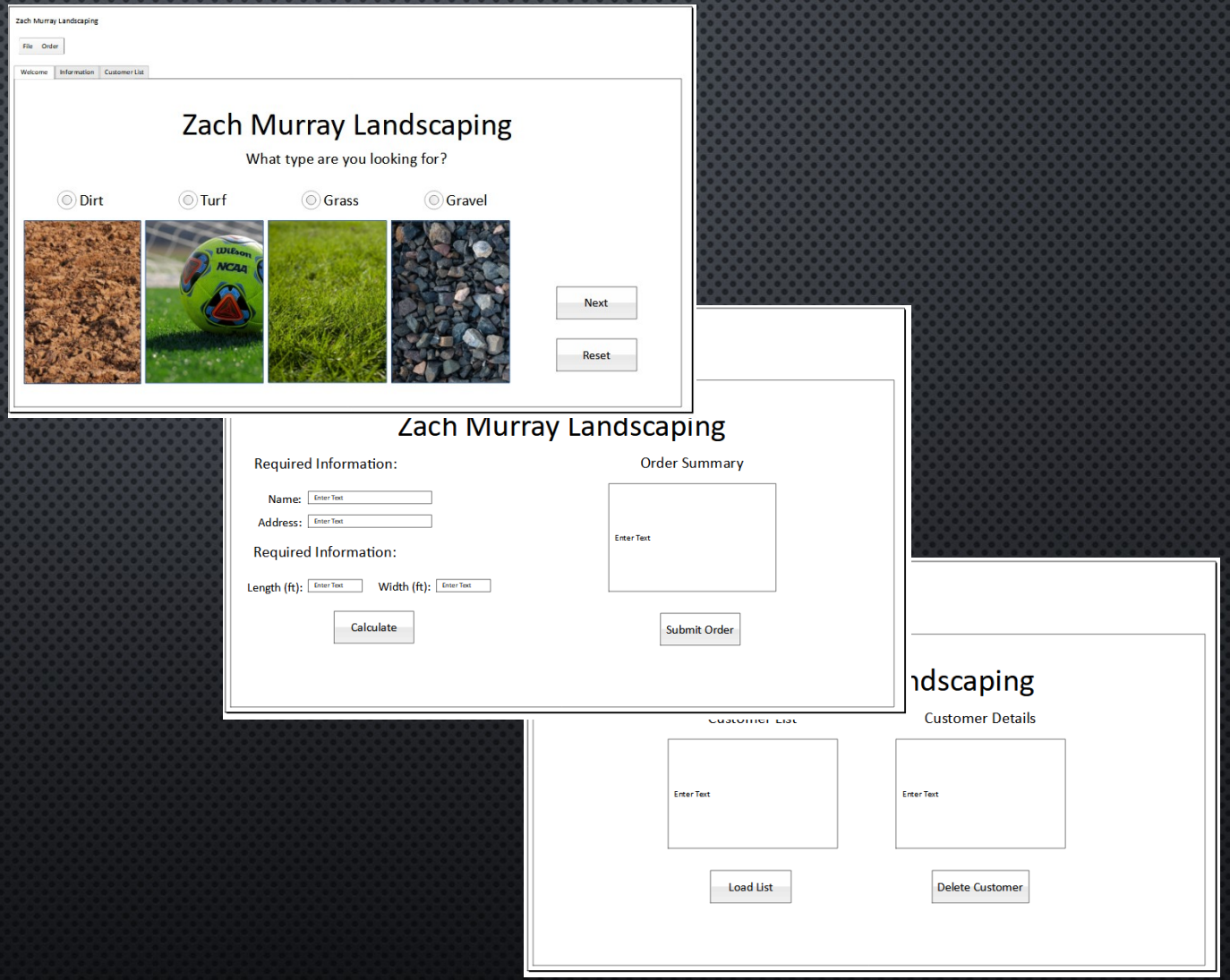
Introduction

- The Landscape App project is an example of basic GUI implementation with Java, using Netbeans and Swing. I also built a cloud database using Heroku and ClearDB.

Week 1

- Phase one is graphic design. We can use Visio to quickly create an example of what the GUI should look like.

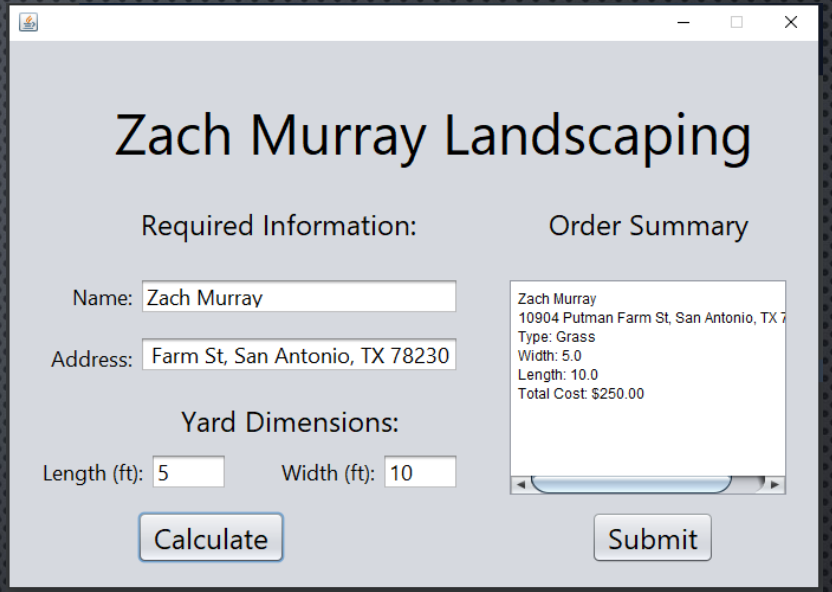
- A free alternative to Visio is LibreOffice Draw, which shows that design here. The design is not static, but allows you to navigate through each tab page. This is great for keeping all of the GUI designs under one file.

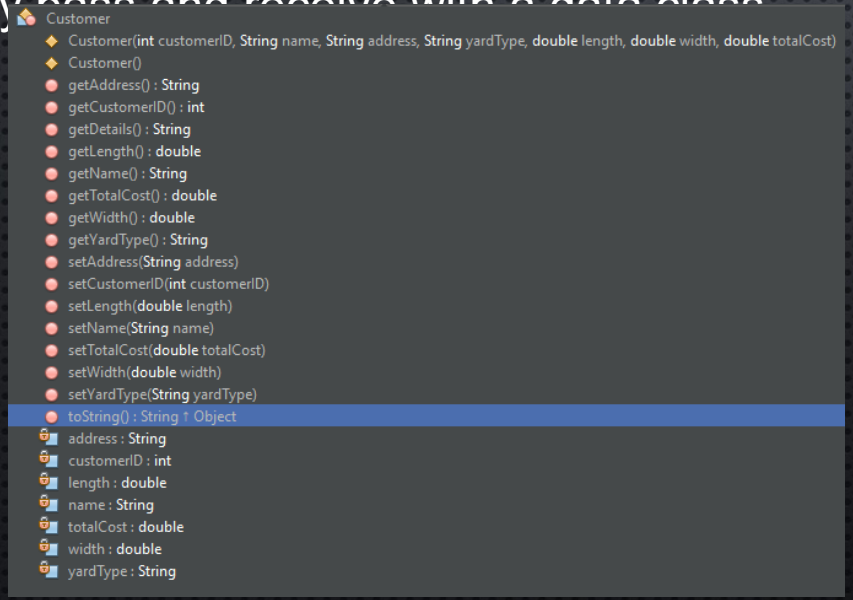


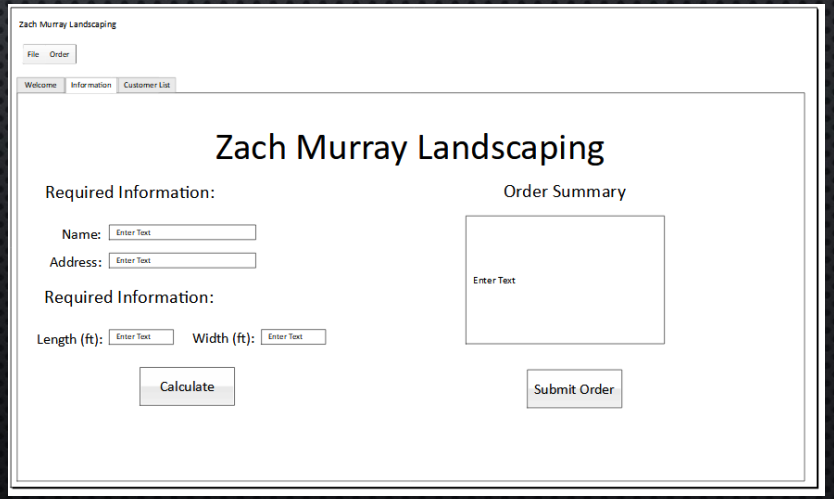
Week 2

- This phase is building the GUI in the Netbeans graphic editor. The second tab will be a good start, so that we can test functionality before moving forward.

Above is the Java GUI designed with Swing components. It will of course like slightly different. But the arrangement and functionality should closely represent the Visio design. We have also implemented the backend methods in the landscapingSummaryGUI class. Another class, Customer, receives the entry form data and will eventually pass and receive with a data class



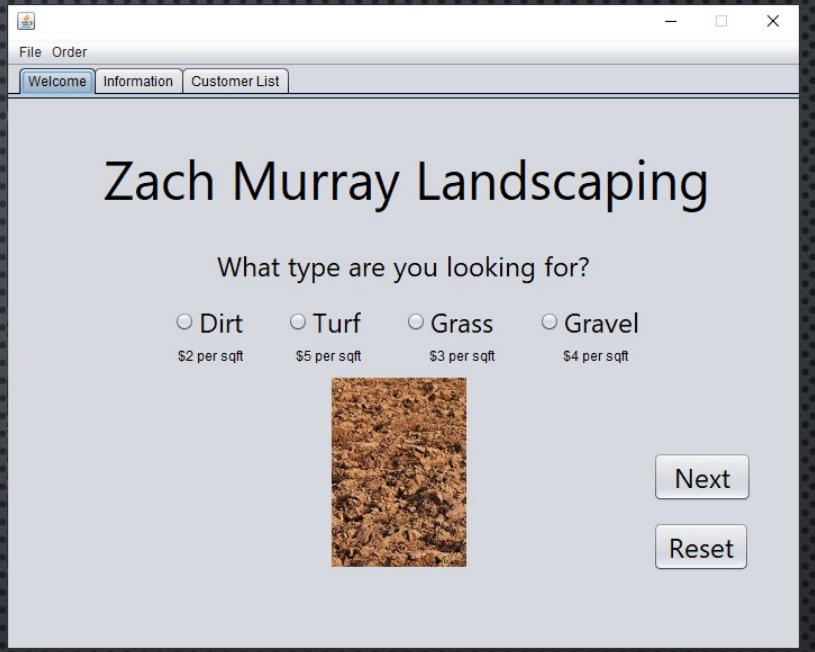


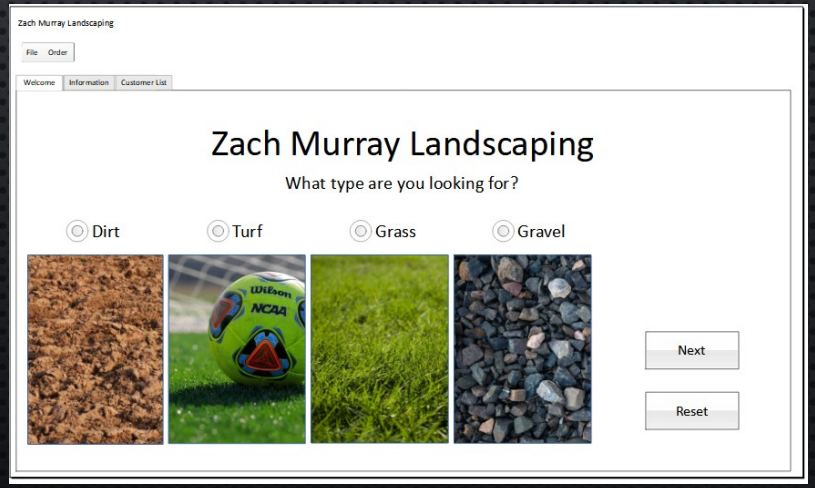


Week 3

- We can now add the first tab, which has some allows the user to select the type of material they want. We have to add some graphics and format them to the desired size before placing them in the GUI.

The first page will look slightly different from the design because we can add some dynamic functionality to the pictures. When each radio button is clicked, the picture changes accordingly. This page is is added to the GUI class in a second tabbed pane.

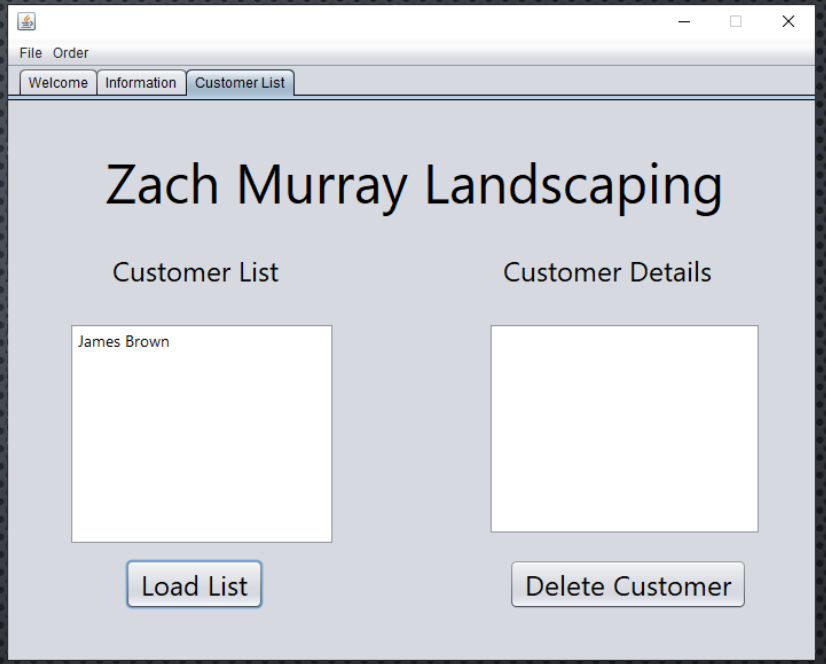


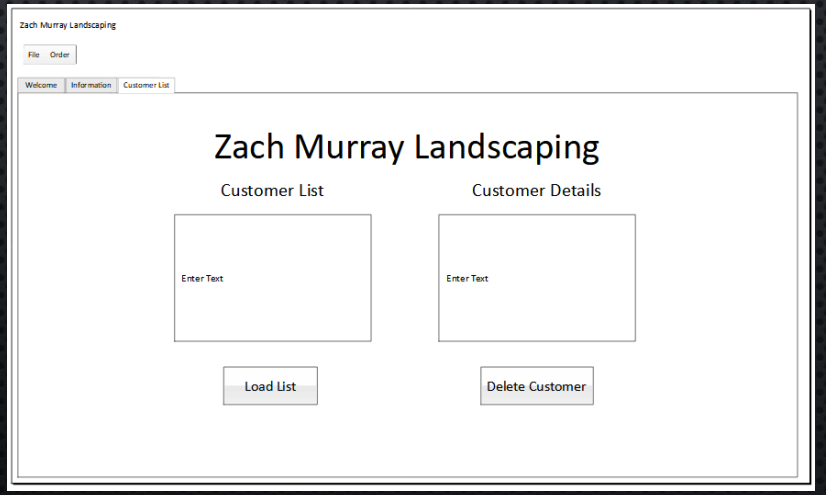


Week 4

- We can add the last tab in this phase, and program some Jlists to display multiple users.

When a user's data is submitted, it adds the Customer object to the DefaultListModel. The result is displayed on the Jlist. We can add and delete customers, but there's no persistence yet.



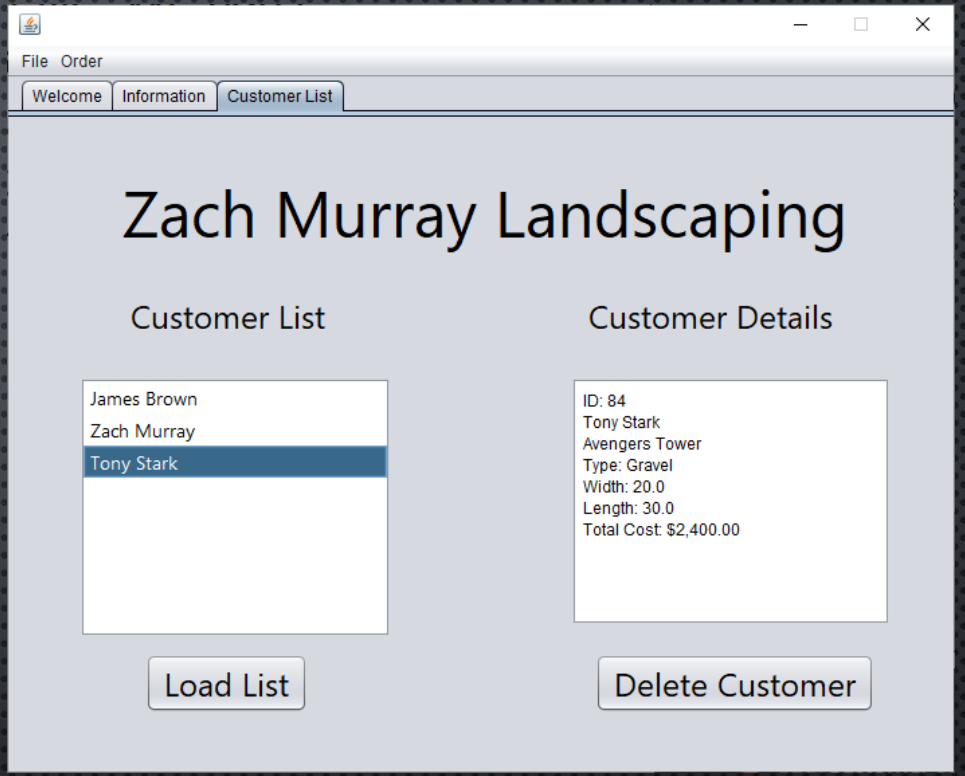


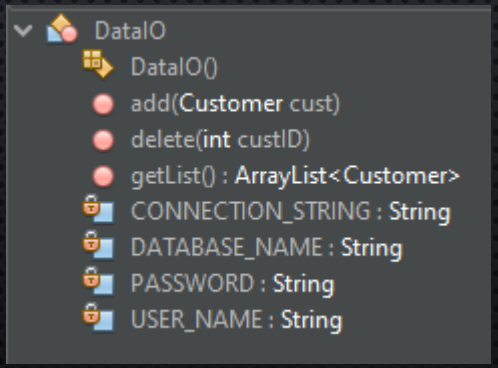
Week 5

- This week we will add some persistence by saving data to a text file. We will implement a DataIO class to handle read and writes.

We will write each customer as a line of data in the text file. We can't access each line at random as easily as a database. Thus it's easier to just delete the old file and recreate it with the changes.

Now we can recall the Customer list automatically when the program restarts. It will also reload when a customer is added or deleted.





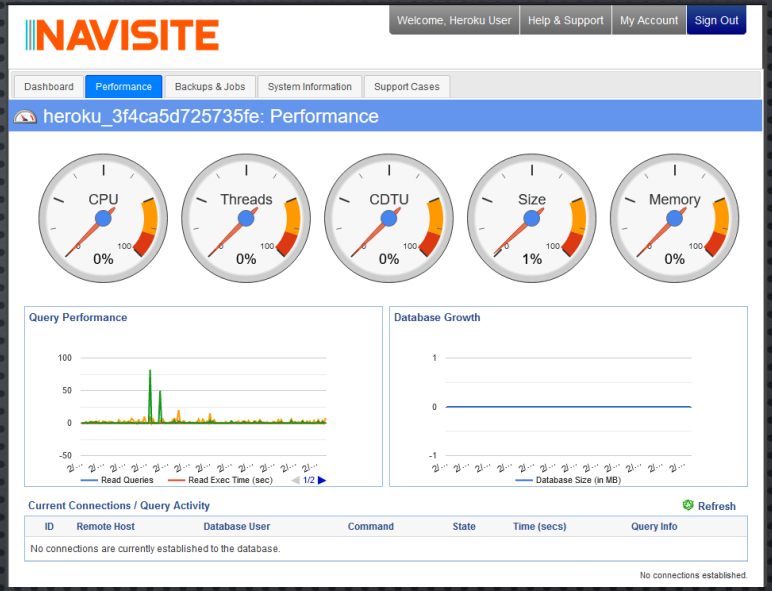
Week 6

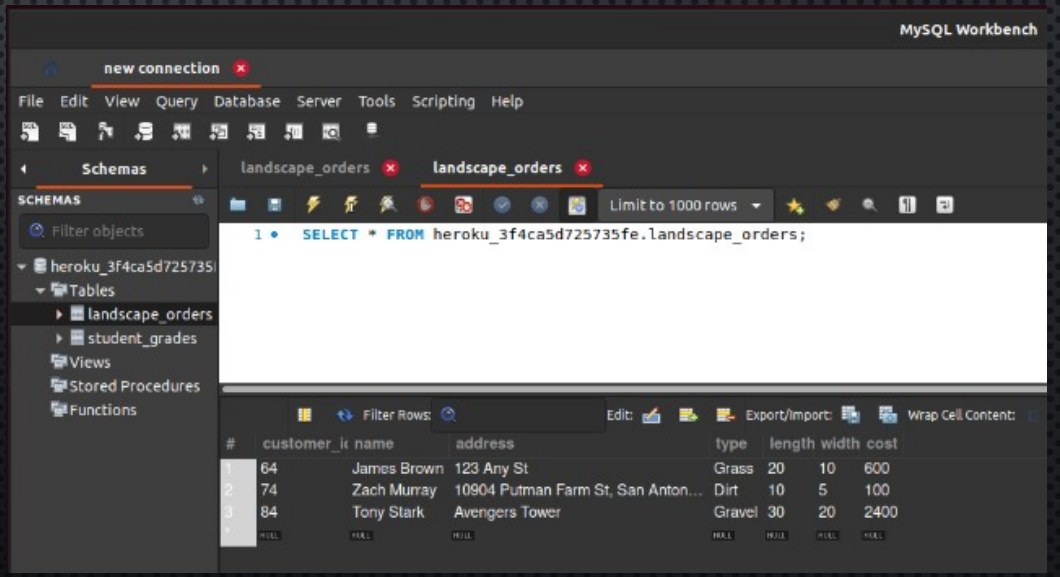
- This phase moves beyond saving to file and making calls to a database instead. This project will utilize MySQL. It’s simple data types are easy to implement with Workbench. We only need one table.

We need to build our database and update the DataIO class methods with a driver, connection, and statements for the add, delete, and getList methods.

This database is Hosted on Heroku using ClearDB. Instead of using localhost for database server, we can access it from anywhere!

It is very important to prepare queries with the Statement class to protect your program from SQL injections.





Challenges

- There were a lot of new concepts used for this project, such as the DefaultList model, read and write methods, and cloud database implementation.

Career Skills

- This project covers most of the Java skills needed to get started in the industry.
- Database management
- Front end development
- Object oriented programming
- Backend logic, view, and data class implementation.

Conclusion

- Although a simple project, it covers most of the important concepts for modern software development. It may serve as a useful reference as well. Future steps include learning frameworks and scaling into web development.