Project : Problem
CS179M: Senior Project Artificial Intelligence, Dr. Eamonn Keogh

Team: Class of 2023
Student names: Chinkyung Choo, Nishitha Gouravelli, Marcell Fulop
Email and SID: cchoo003@ucr.edu(862215214), ngour001@ucr.edu (862183759), mfulo001@ucr.edu (862181078)
Date: March, 28th,2023

To run:
1.
clone repo
cd into algorithm/backEnd folder
run python3 main.py

2. Goto branch: CS179M compile_Web_Application
And follow the instructions on README file.

1. **Introduction and Requirements Analysis**

- The problem to solve:

  Our customer Mr. Keogh owns a single port, in Long Beach. He runs a business that makes profits by loading and unloading containers from a ship to his port. When a ship arrives in his port, the operator of the port moves containers with a crane.
  The business makes profits by the number of containers moved from the ship to the port. Thus Mr. Keogh wants a system that calculates the optimal solution that moves as many containers from a ship to the port in the least amount of time with least cost of crane operation. To achieve this goal, containers need to be balanced and sifted as well to minimize operation time and crane operation. Therefore, our team mission is to create software that operates on algorithms which calculates optimal solutions for loading, unloading and balancing containers based on his business needs.

- Elicitation:
  - Our team interviewed our professor and a client Mr. Keogh to discuss desired technical features for our software .
  - Notes and feedback from the client interview and our product presentation.
    - A client wants a log file that displays the workflow of the software. For instance, the desired log file will display a timestamp of when an operator

logs in into the software, when manifest.txt file was uploaded in the system and log message from an operator.

- Log file system enables a user to comment on each other's operation through a log message creation feature.
- A client wants the weight of a container to be integer in decimal form.
- A client wants a notification alert system for instance, when unload/load operations are done, the system must notify the completion of the work with the result.
- A client wants us to keep the user logged in all the time once a user has logged in into the system.
- We have to be aware of stakeholders, who will be affected by this project such as a user, port of Long Beach, Ship crew, Crane operator, truck drivers, shipping companies, and Department of Homeland Security as stakeholder.

2. System Design and Program Design.

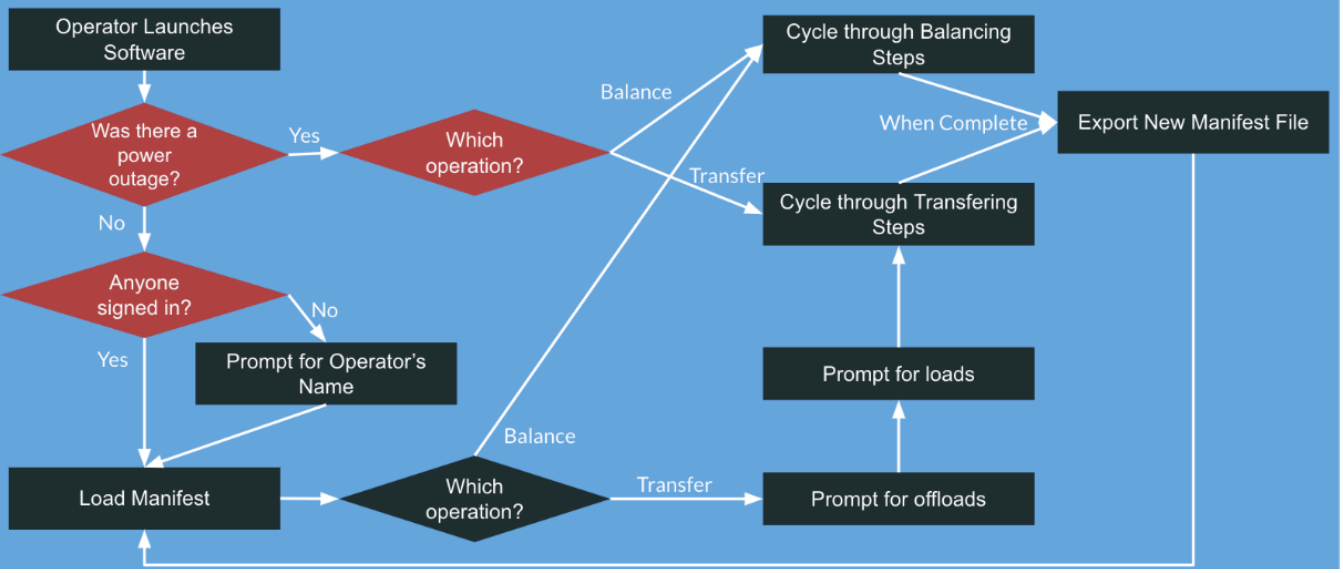
- **System Design**

System Design was delivered through an in person project pitch presentation to a client Mr. Kough on February 10th, 2023. There was a slight modification on the presentation powerpoint based on Mr. Keogh's feedback after the project pitch. Modified presentation power point is accessible through our submission. You can also access the detailed system design in the project pitch powerpoint via this link.

https://drive.google.com/drive/folders/1pO0yv-oNBx2hmLYcJqLZGiJhtUbx6TP7?usp=share_link

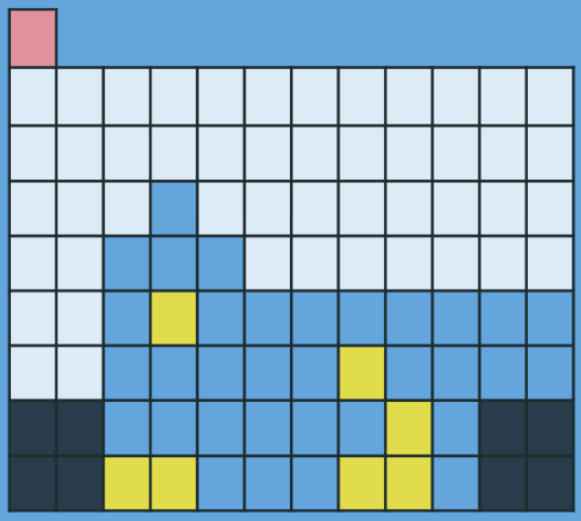
- **Flow chart of the Program Design.**

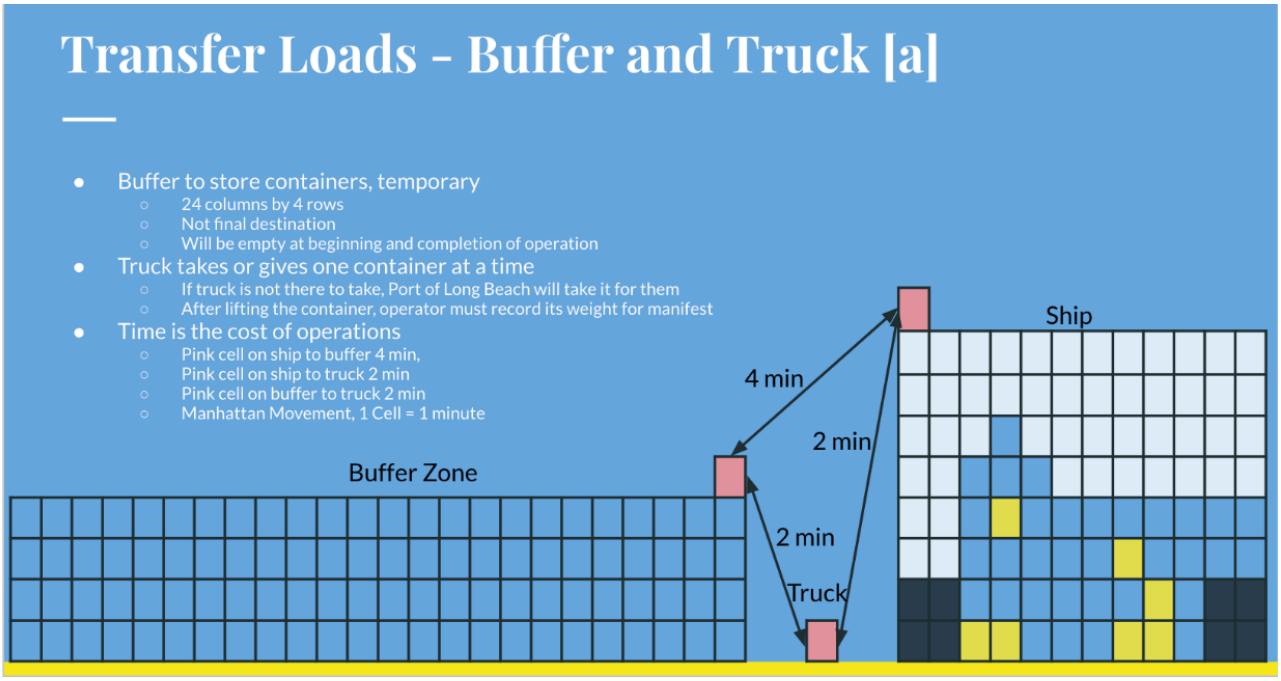
Basic Startup Overview Flowchart



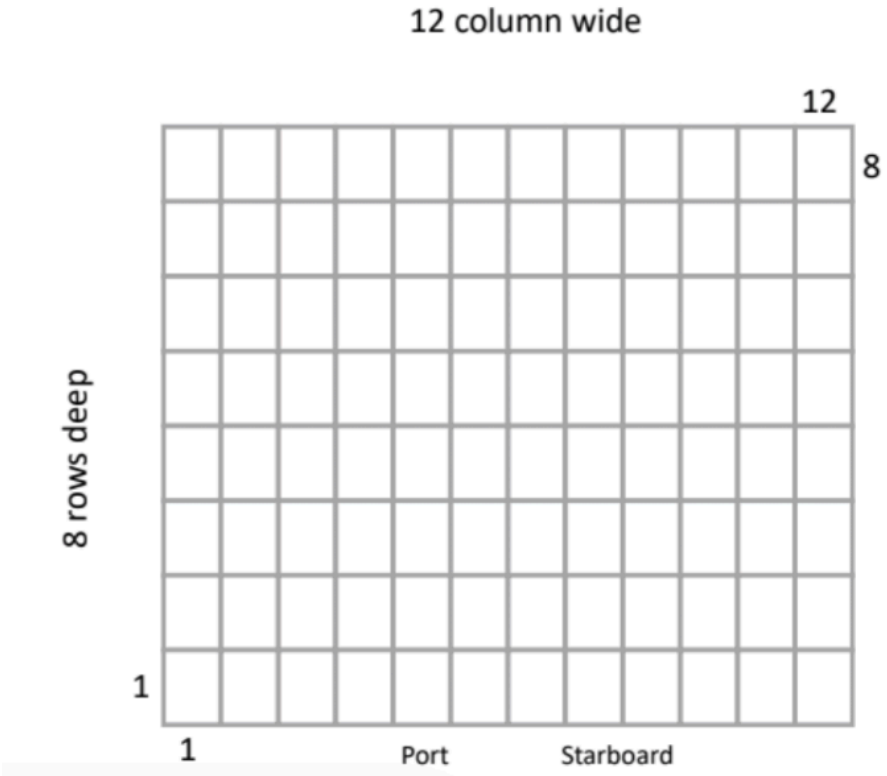
Transfer Loads - Ship Docked [a]

- Crane begins at location denoted by the pink cell
- Ship
  - 12 columns by 8 rows
    - One container depth
  - Some cells are part of hull and cannot be occupied by containers
    - Hull design is symmetric for both sides
    - Hull cannot "overhang"
- Top pink square represents the point where crane moves container to buffer or truck





Architecture of grid: The size of the ship's grid is 8 rows by 12 columns while the buffer is 4 rows by 24 columns.



4.Coding.

- **Algorithm:**
  The algorithm is split into three different problem sets, Transfer, SIFT, and Balance which all are derived from Port. They share similar functionality in that all use A* to find the optimal solution. In each type of problem, we first apply the operators to the initial state of the Port. Operators are like moving the crane with or without a container. We only want to do valid moves which excludes moving a container to be offloaded when it is not supposed to be or moving a container to be part of the ship's hull (or "NAN" in the manifest). We also want to limit the number of operators or else the diameter of the problem is going to be huge and take too long to calculate.
  We can make some optimizations to limit the number of operators while still maintaining optimality. We can reason it is pointless to move the crane twice without a container or to move the crane to an empty cell which means the number of operators for a lightly loaded ship is an average of around 5.
  We calculate A* cost which is the cost of getting to that node and the heuristic cost which is the estimated time to completion. Recall that so long as our heuristic never overestimates the time to completion, our search will always find the optimal or shortest route.
  After filtering out transfer ports that were already made, we add them to the queue for which then we sort them by the A* cost. We check if the lowest A* cost is a solved solution (solved solutions will have an A* value of 0).
  Afterwhich, we repeat the process with lowest A* value in the queue and apply our operators.
  Some complexity happens with balance because there are two ways to balance. Legal balance where the heavier side no greater 10% of the lighter side. SIFT is a formatted way of laying down containers. Typically, legal balance is easier to achieve than SIFT but sometimes that is not possible so we must use SIFT.
  That means we calculate SIFT and find out the fastest time and then we try legal balance. Once the cost to run legal balance starts to go over SIFT's best time, we quit legal balance and use the SIFT solution otherwise we go for legal balance.
- **User Interface:** Integration of web application and python applications is not fully integrated. However, we will integrate python application features into our Web application.
  - **Command line**
    - The user compiles and runs the application through the command line. They will be prompted to enter their first and last name to login to the system. They are then greeted and asked to upload the manifest file. They

can then begin either transfer or balancing actions. For the transfer operation, the user can select which containers to offload and which to onload. The user has the opportunity to enter any log comments about the containers. Once the user chooses to begin the calculation, the console outputs each step as a series of coordinate moves. These are the moves the crane follows on a coordinate grid. There is also a text description of the move summarizing where the crane is moving (truck, buffer, ship), its source and destination coordinates, and if it is picking up any containers. The command line allows the user to enter log comments and sign out regularly. They also have the option to view an action step again. At the end of the operation, the total time of movement is calculated and the updated manifest file is saved onto the user's desktop.

- **Python application**
  - Our python application makes use of the Tkinter library to display an animation of the steps on the screen. The program displays a 9x12 grid representing the ship. The ship's manifest is read and the container names are written to the grid boxes accordingly for the user to see the initial ship configuration. The top row in the grid is representative of a buffer for the crane to move. A sliding box representing the crane will slide along the grid, following the movement path returned by the transfer function. The box will only travel vertically and horizontally, but not diagonally. When the slider box overlaps with an existing container on the grid, the container will be moved along with the slider to its designated destination.

- **Web application**

- Our web application was built with HTML, CSS, and Javascript. They are script languages which are suitable to build web applications with user interaction with graphical visualization. Javascript is an object oriented script language thus it is suitable to handle this kind of large scale program.
- Web application runs on localhost port number 3000. The server connection was written in Javascript. The client side of our application was built in Javascript, HTML, and CSS.
- A user can login, upload manifest files, export manifest text files to their desktop and phone, log messages and read log messages in our web application.

5. Unit and Integration Testing.

We will consider the transfer operation to be successful if:

1. It can unload containers from the ship to the truck while following the shortest path
2. It can load containers onto the ship while following the shortest path
3. It can unload two containers that are stacked on top of one another, by removing the top one first
4. It can choose the most optimal container to unload when there are 2 or more duplicates.
5. It can successfully complete the transfer process with numerous load/unload instructions.
6. It can find the optimal solution for any calculation in less than 1 minute.

1. ***unload "cat" from ShipCase1***

```
2
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
0
C00000000
00000000U
00000000S
000000000
000000000
000000000
000000000
000000000
000000000
000000000
000000000
000000000

Found solution

(0, 0) (0, 0)

Moving crane only from Ship (0, 0) to Ship (1, 8)
(0, 0) (0, 0) (1, 0) (1, 1) (1, 2) (1, 3) (1, 4) (1, 5) (1, 6) (1, 7) (1, 8)

Moving container Cat (weight: 99 kg) from Ship (1, 8) to Truck Bay (0, 0)
(1, 8) (1, 7) (1, 6) (1, 5) (1, 4) (1, 3) (1, 2) (1, 1) (1, 0) (0, 0) (-1, 0)
20 minutes
1 3
```

2. ***load batteries “bat” onto ShipCase1. weight 431***

```
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
0
C00000000
00000000S
00000000S
000000000
000000000
000000000
000000000
000000000
000000000
000000000
000000000
000000000

Found solution

(0, 0) (0, 0)

Moving crane only from Ship (0, 0) to Truck Bay (0, 0)
(0, 0) (-1, 0)

Moving container bat (weight: 431 kg) from Truck Bay (0, 0) to Ship (0, 7)
(-1, 0) (0, 0) (0, 1) (0, 2) (0, 3) (0, 4) (0, 5) (0, 6) (0, 7)
11 minutes
15 17
```

3. ***Unload batteries “cat” and “ewe” from ShipCase3***

```
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
0
C000000UU
0000000SS
00000000S
00000000S
000000000
000000000
000000000
000000000
000000000
000000000
000000000
000000000

Found solution

(0, 0) (0, 0)

Moving crane only from Ship (0, 0) to Ship (0, 7)
(0, 0) (0, 0) (0, 1) (0, 2) (0, 3) (0, 4) (0, 5) (0, 6) (0, 7)

Moving container Cat (weight: 9041 kg) from Ship (0, 7) to Truck Bay (0, 0)
(0, 7) (0, 6) (0, 5) (0, 4) (0, 3) (0, 2) (0, 1) (0, 0) (-1, 0)

Moving crane only from Truck Bay (0, 0) to Ship (0, 8)
(-1, 0) (0, 0) (0, 1) (0, 2) (0, 3) (0, 4) (0, 5) (0, 6) (0, 7) (0, 8)

Moving container Ewe (weight: 10001 kg) from Ship (0, 8) to Truck Bay (0, 0)
(0, 8) (0, 7) (0, 6) (0, 5) (0, 4) (0, 3) (0, 2) (0, 1) (0, 0) (-1, 0)
36 minutes
```

4.***offload “cat” from CustomCase1.txt***

```
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
00000
0
C00000000
0000000US
00000000S
000000000
000000000
000000000
000000000
000000000
000000000
000000000
000000000
000000000

Found solution

(0, 0) (0, 0)

Moving crane only from Ship (0, 0) to Ship (1, 7)
(0, 0) (0, 0) (1, 0) (1, 1) (1, 2) (1, 3) (1, 4) (1, 5) (1, 6) (1, 7)

Moving container Cat (weight: 0 kg) from Ship (1, 7) to Truck Bay (0, 0)
(1, 7) (1, 6) (1, 5) (1, 4) (1, 3) (1, 2) (1, 1) (1, 0) (0, 0) (-1, 0)
18 minutes
```

5. ***offload “pig”, “owl” and “dog” from ShipCase4.txt. Load “lol” weight 21 and “hah” weight 211.***

```
Moving crane only from Ship (0, 0) to Ship (4, 1)
(0, 0) (0, 0) (1, 0) (2, 0) (3, 0) (4, 0) (4, 1)

Moving container Pig (weight: 3044 kg) from Ship (4, 1) to Truck Bay (0, 0)
(4, 1) (4, 0) (3, 0) (2, 0) (1, 0) (0, 0) (-1, 0)

Moving container Hah (weight: 211 kg) from Truck Bay (0, 0) to Ship (0, 6)
(-1, 0) (0, 0) (0, 1) (0, 2) (0, 3) (0, 4) (0, 5) (0, 6)

Moving crane only from Ship (0, 6) to Ship (4, 2)
(0, 6) (0, 5) (0, 4) (0, 3) (0, 2) (0, 1) (0, 0) (0, 0) (1, 0) (2, 0) (3, 0) (4, 0) (4, 1) (4, 2)

Moving container Doe (weight: 1100 kg) from Ship (4, 2) to Ship (3, 7)
(4, 2) (4, 1) (4, 0) (3, 0) (3, 1) (3, 2) (3, 3) (3, 4) (3, 5) (3, 6) (3, 7)

Moving crane only from Ship (3, 7) to Ship (4, 3)
(3, 7) (3, 6) (3, 5) (3, 4) (3, 3) (3, 2) (3, 1) (3, 0) (3, 0) (4, 0) (4, 1) (4, 2) (4, 3)

Moving container Owl (weight: 2020 kg) from Ship (4, 3) to Truck Bay (0, 0)
(4, 3) (4, 2) (4, 1) (4, 0) (3, 0) (2, 0) (1, 0) (0, 0) (-1, 0)

Moving container Lol (weight: 21 kg) from Truck Bay (0, 0) to Ship (4, 3)
(-1, 0) (0, 0) (1, 0) (2, 0) (3, 0) (4, 0) (4, 1) (4, 2) (4, 3)

Moving crane only from Ship (4, 3) to Ship (3, 7)
(4, 3) (4, 2) (4, 1) (4, 0) (3, 0) (3, 1) (3, 2) (3, 3) (3, 4) (3, 5) (3, 6) (3, 7)

Moving container Doe (weight: 1100 kg) from Ship (3, 7) to Ship (4, 2)
(3, 7) (3, 6) (3, 5) (3, 4) (3, 3) (3, 2) (3, 1) (3, 0) (3, 0) (4, 0) (4, 1) (4, 2)

Moving crane only from Ship (4, 2) to Ship (0, 6)
(4, 2) (4, 1) (4, 0) (3, 0) (2, 0) (1, 0) (0, 0) (0, 1) (0, 2) (0, 3) (0, 4) (0, 5) (0, 6)

Moving container Hah (weight: 211 kg) from Ship (0, 6) to Ship (1, 7)
(0, 6) (0, 5) (0, 4) (0, 3) (0, 2) (0, 1) (0, 0) (0, 0) (1, 0) (1, 1) (1, 2) (1, 3) (1, 4) (1, 5) (1, 6) (1, 7)

Moving crane only from Ship (1, 7) to Ship (4, 2)
(1, 7) (1, 6) (1, 5) (1, 4) (1, 3) (1, 2) (1, 1) (1, 0) (1, 0) (2, 0) (3, 0) (4, 0) (4, 1) (4, 2)

Moving container Doe (weight: 1100 kg) from Ship (4, 2) to Ship (3, 7)
(4, 2) (4, 1) (4, 0) (3, 0) (3, 1) (3, 2) (3, 3) (3, 4) (3, 5) (3, 6) (3, 7)

Moving crane only from Ship (3, 7) to Ship (4, 3)
(3, 7) (3, 6) (3, 5) (3, 4) (3, 3) (3, 2) (3, 1) (3, 0) (3, 0) (4, 0) (4, 1) (4, 2) (4, 3)

Moving container Lol (weight: 21 kg) from Ship (4, 3) to Ship (3, 6)
(4, 3) (4, 2) (4, 1) (4, 0) (3, 0) (3, 1) (3, 2) (3, 3) (3, 4) (3, 5) (3, 6)

Moving crane only from Ship (3, 6) to Ship (4, 4)
(3, 6) (3, 5) (3, 4) (3, 3) (3, 2) (3, 1) (3, 0) (3, 0) (4, 0) (4, 1) (4, 2) (4, 3) (4, 4)

Moving container Ewe (weight: 10000 kg) from Ship (4, 4) to Ship (2, 7)
(4, 4) (4, 3) (4, 2) (4, 1) (4, 0) (3, 0) (2, 0) (2, 1) (2, 2) (2, 3) (2, 4) (2, 5) (2, 6) (2, 7)

Moving crane only from Ship (2, 7) to Ship (3, 6)
(2, 7) (2, 6) (2, 5) (2, 4) (2, 3) (2, 2) (2, 1) (2, 0) (2, 0) (3, 0) (3, 1) (3, 2) (3, 3) (3, 4) (3, 5) (3, 6)

Moving container Lol (weight: 21 kg) from Ship (3, 6) to Ship (4, 4)
(3, 6) (3, 5) (3, 4) (3, 3) (3, 2) (3, 1) (3, 0) (3, 0) (4, 0) (4, 1) (4, 2) (4, 3) (4, 4)

Moving crane only from Ship (4, 4) to Ship (1, 7)
(4, 4) (4, 3) (4, 2) (4, 1) (4, 0) (3, 0) (2, 0) (1, 0) (1, 1) (1, 2) (1, 3) (1, 4) (1, 5) (1, 6) (1, 7)

Moving container Hah (weight: 211 kg) from Ship (1, 7) to Ship (4, 3)
(1, 7) (1, 6) (1, 5) (1, 4) (1, 3) (1, 2) (1, 1) (1, 0) (1, 0) (2, 0) (3, 0) (4, 0) (4, 1) (4, 2) (4, 3)
```

```
Moving crane only from Ship (4, 3) to Ship (2, 7)
(4, 3) (4, 2) (4, 1) (4, 0) (3, 0) (2, 0) (2, 1) (2, 2) (2, 3) (2, 4) (2, 5) (2, 6) (2, 7)

Moving container Ewe (weight: 10000 kg) from Ship (2, 7) to Ship (3, 6)
(2, 7) (2, 6) (2, 5) (2, 4) (2, 3) (2, 2) (2, 1) (2, 0) (2, 0) (3, 0) (3, 1) (3, 2) (3, 3) (3, 4) (3, 5) (3, 6)

Moving crane only from Ship (3, 6) to Ship (4, 3)
(3, 6) (3, 5) (3, 4) (3, 3) (3, 2) (3, 1) (3, 0) (3, 0) (4, 0) (4, 1) (4, 2) (4, 3)

Moving container Hah (weight: 211 kg) from Ship (4, 3) to Ship (3, 5)
(4, 3) (4, 2) (4, 1) (4, 0) (3, 0) (3, 1) (3, 2) (3, 3) (3, 4) (3, 5)

Moving crane only from Ship (3, 5) to Ship (4, 4)
(3, 5) (3, 4) (3, 3) (3, 2) (3, 1) (3, 0) (3, 0) (4, 0) (4, 1) (4, 2) (4, 3) (4, 4)

Moving container Lol (weight: 21 kg) from Ship (4, 4) to Ship (2, 7)
(4, 4) (4, 3) (4, 2) (4, 1) (4, 0) (3, 0) (2, 0) (2, 1) (2, 2) (2, 3) (2, 4) (2, 5) (2, 6) (2, 7)

Moving crane only from Ship (2, 7) to Ship (4, 5)
(2, 7) (2, 6) (2, 5) (2, 4) (2, 3) (2, 2) (2, 1) (2, 0) (2, 0) (3, 0) (4, 0) (4, 1) (4, 2) (4, 3) (4, 4) (4, 5)

Moving container Cow (weight: 2011 kg) from Ship (4, 5) to Ship (2, 6)
(4, 5) (4, 4) (4, 3) (4, 2) (4, 1) (4, 0) (3, 0) (2, 0) (2, 1) (2, 2) (2, 3) (2, 4) (2, 5) (2, 6)

Moving crane only from Ship (2, 6) to Ship (4, 6)
(2, 6) (2, 5) (2, 4) (2, 3) (2, 2) (2, 1) (2, 0) (2, 0) (3, 0) (4, 0) (4, 1) (4, 2) (4, 3) (4, 4) (4, 5) (4, 6)

Moving container Dog (weight: 2007 kg) from Ship (4, 6) to Truck Bay (0, 0)
(4, 6) (4, 5) (4, 4) (4, 3) (4, 2) (4, 1) (4, 0) (3, 0) (2, 0) (1, 0) (0, 0) (-1, 0)
24 minutes
```

6. We were not able to pass this test.

6.System Testing.

We will accept the system to be successful in this stage if:

1. User can log out in the middle of an operation and a new user can sign in successfully.
2. User can enter log comments at any point in the operation.
3. System should return an updated manifest file at the end of all ship operations. User should be reminded to do this.
4. If user is not able to complete the step before the animation ends, they should be able to view the step again.
5. Log file should not be deleted
6. System should be able to resume operation wherever it left off, even after the power goes out.
7. System should check that the manifest file uploaded is valid before proceeding with operations.

1.

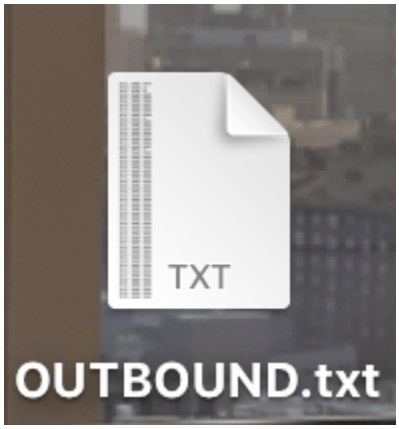
```
2023-03-29 21:49:53.642487 User One signed in
2023-03-29 21:49:53.642939 Manifest ShipCase3.txt was opened, there are 4 containers on the ship
2023-03-29 21:50:23.646153 User One signs out
2023-03-29 21:50:30.262822 User Two signed in
2023-03-29 21:50:39.225632 container Dog was successfully offloaded.
```

2.

```
2023-03-29 21:52:40.447013 Juice Monster signed in
2023-03-29 21:52:40.447397 Manifest ShipCase3.txt was opened, there are 4 containers on the ship
2023-03-29 21:53:17.802633 Juice Monster logged: This container has been breached. it's open
2023-03-29 21:53:32.641945 container Cat was successfully offloaded.
2023-03-29 21:53:32.641965 container Apple was successfully offloaded.
```

3.

**OUTBOUND manifest is created and exported to Desktop*



```
[01,01], {00000}, UNUSED
[01,02], {00500}, Cow
[01,03], {00600}, Dog
[01,04], {00100}, Rat
[01,05], {00000}, UNUSED
[01,06], {00000}, UNUSED
[01,07], {00000}, UNUSED
[01,08], {00000}, UNUSED
[01,09], {00000}, UNUSED
[01,10], {00000}, UNUSED
[01,11], {00000}, UNUSED
[01,12], {00000}, UNUSED
[02,01], {00000}, UNUSED
[02,02], {00010}, Doe
[02,03], {00000}, UNUSED
[02,04], {00000}, UNUSED
[02,05], {00000}, UNUSED
[02,06], {00000}, UNUSED
[02,07], {00000}, UNUSED
[02,08], {00000}, UNUSED
[02,09], {00000}, UNUSED
[02,10], {00000}, UNUSED
[02,11], {00000}, UNUSED
[02,12], {00000}, UNUSED
[03,01], {00000}, UNUSED
[03,02], {09041}, Cat
[03,03], {00000}, UNUSED
[03,04], {00000}, UNUSED
[03,05], {00000}, UNUSED
[03,06], {00000}, UNUSED
[03,07], {00000}, UNUSED
[03,08], {00000}, UNUSED
[03,09], {00000}, UNUSED
[03,10], {00000}, UNUSED
[03,11], {00000}, UNUSED
[03,12], {00000}, UNUSED
[04,01], {00000}, UNUSED
[04,02], {00000}, UNUSED
[04,03], {00000}, UNUSED
[04,04], {00000}, UNUSED
[04,05], {00000}, UNUSED
[04,06], {00000}, UNUSED
[04,07], {00000}, UNUSED
[04,08], {00000}, UNUSED
[04,09], {00000}, UNUSED
[04,10], {00000}, UNUSED
[04,11], {00000}, UNUSED
[04,12], {00000}, UNUSED
[05,01], {00000}, UNUSED
[05,02], {00000}, UNUSED
[05,03], {00000}, UNUSED
[05,04], {00000}, UNUSED
[05,05], {00000}, UNUSED
```

4.

```
Moving crane only from Ship (0, 0) to Ship (2, 8)
(0, 0) (0, 0) (1, 0) (2, 0) (2, 1) (2, 2) (2, 3) (2, 4) (2, 5) (2, 6) (2, 7) (2, 8)
Did you complete this step? (Enter Y or N): Enter l to log a comment. N
Would you like to sign out? (Enter Y or N) N

 Repeating step for you.

Moving crane only from Ship (0, 0) to Ship (2, 8)
(0, 0) (0, 0) (1, 0) (2, 0) (2, 1) (2, 2) (2, 3) (2, 4) (2, 5) (2, 6) (2, 7) (2, 8)
Did you complete this step? (Enter Y or N): Enter l to log a comment.Y

Moving container Dog (weight: 100 kg) from Ship (2, 8) to Truck Bay (0, 0)
(2, 8) (2, 7) (2, 6) (2, 5) (2, 4) (2, 3) (2, 2) (2, 1) (2, 0) (1, 0) (0, 0) (-1, 0)
Did you complete this step? (Enter Y or N): Enter l to log a comment.
```

5. Log file successfully maintains all updates since it was first created

```
2023-03-29 20:09:43.160490 Nish Gou signed in
2023-03-29 20:09:43.161041 Manifest ShipCase3.txt was opened, there are 5 containers on the ship
2023-03-29 20:12:04.609044 Hi Bye signed in
2023-03-29 20:12:04.609620 Manifest ShipCase3.txt was opened, there are 4 containers on the ship
2023-03-29 20:12:52.661507 a a signed in
2023-03-29 20:12:52.661935 Manifest ShipCase3.txt was opened, there are 4 containers on the ship
2023-03-29 20:14:27.163976 a a signed in
2023-03-29 20:18:17.295433 nish gou signed in
2023-03-29 20:18:17.295958 Manifest ShipCase3.txt was opened, there are 4 containers on the ship
2023-03-29 20:19:25.765974 a a signed in
2023-03-29 20:19:25.766640 Manifest ShipCase3.txt was opened, there are 4 containers on the ship
2023-03-29 20:20:50.482965 a a signed in
2023-03-29 20:20:50.483421 Manifest ShipCase3.txt was opened, there are 4 containers on the ship
2023-03-29 20:22:57.141426 nish g signed in
2023-03-29 20:22:57.141874 Manifest ShipCase3.txt was opened, there are 4 containers on the ship
2023-03-29 20:23:47.784180 hi nye signed in
2023-03-29 20:23:47.784703 Manifest ShipCase3.txt was opened, there are 4 containers on the ship
2023-03-29 20:24:53.461399 nish k signed in
2023-03-29 20:24:53.461813 Manifest ShipCase3.txt was opened, there are 4 containers on the ship
2023-03-29 21:17:55.620050 nis s signed in
2023-03-29 21:17:55.626377 Manifest ShipCase3.txt was opened, there are 4 containers on the ship
2023-03-29 21:18:20.483985 h h signed in
2023-03-29 21:18:20.484389 Manifest ShipCase3.txt was opened, there are 4 containers on the ship
2023-03-29 21:19:46.028296 a a signed in
2023-03-29 21:19:46.028705 Manifest ShipCase3.txt was opened, there are 4 containers on the ship
2023-03-29 21:20:02.997986 a a: i'm testing this
2023-03-29 21:21:18.556884 a a: akskkdkskdkd
2023-03-29 21:22:03.699160 h h signed in
2023-03-29 21:22:03.699722 Manifest ShipCase3.txt was opened, there are 4 containers on the ship
2023-03-29 21:22:14.717181 h h: dkkkff
2023-03-29 21:24:57.240560 a a signed in
2023-03-29 21:24:57.241022 Manifest ShipCase3.txt was opened, there are 4 containers on the ship
2023-03-29 21:25:12.198849 a a: it was heavier
2023-03-29 21:27:55.504473 gina joe signed in
2023-03-29 21:27:55.504901 Manifest ShipCase3.txt was opened, there are 4 containers on the ship
2023-03-29 21:28:13.138022 gina joe signs out
2023-03-29 21:28:19.179892 Joe Joe signed in
2023-03-29 21:47:42.925475 ugh tired signed in
2023-03-29 21:47:42.931561 Manifest ShipCase3.txt was opened, there are 4 containers on the ship
```

6. Were not able to implement this.

7. ***User enters a manifest file with only 90 coordinates.***

```
Hello t t
Type in the name of the manifest file: CustomCase1.txt
Not a valid manifest file. Our ships are 12x8. Please enter a new one:ShipCase3.txt
Ship has container named <Ewe>
Ship has container named <Cow>
Ship has container named <Dog>
Ship has container named <Rat>
Ship has container named <Cat>
Ship has container named <Doe>
```

7.Acceptance Testing.

1) A user can log into our web application in 5 seconds.
2) When a user is logged in, users can see theirs name on the top of ‘optimal solution calculation page’
3) When a user logs in, a web browser transfers a user to an optimal calculation web page in 5 seconds.
4) When a user logs in, log files automatically writes a time stamp that a user logged in to the system. A user can see the time stamp on the optimal calculation web page.
5) An optimal calculation web page enables a user to upload any text file in 5 seconds regardless of file size. It takes less than 5 seconds to upload 2 Kilo bytes manifest text file.
6) A user can update multiple manifest text files in the system.
7) A user can write a message into the log file.
8) A user can export a text file to their desktop and their phones from our program within 5 seconds for 2 kilo bytes size text file.

8.A Maintenance Plan.

- We will provide 10 year basic support
    - We ensure software updates based on user experience and user request.
    - Fixing critical bugs to ensure its business meets.

- For 20% of contract amount and a deadline of one year during basic support, can exercise one of the following.
    - Port software to a different platform or newer version of Windows e.g. iOS.
    - Upgrade software to handle different dimensioned ships.
    - Upgrade software to handle different time costs of movement e.g. moving one cell over costing ½ minute instead 1 minute
- Does not support any versions of Windows besides Windows 10/11

9.Pointer to github code and Demo video

1. GitHub: https://github.com/marcusphilips/ContainerTransferAndBalance

2. Demovideo:

https://drive.google.com/drive/folders/1e8cXErChl8E7EnrfIeI92U0eRGH4TmZd?usp=share_link

10. Improvement.

- There are several features we need to improve in the future.
    - Our web application has most of the features except optimal calculation of balancing , sifting, loading, and unloading. However, those missing features were completed on python program. We tried to integrate web and python, however it kept crashing with errors. Thus we decided to submit our work in this way.
    - We could not complete a successful animation of the steps. We were able to create a ship grid and populate the container values in the right locations according to the manifest (image attached in UI portion). However, we ran into issues when we tried to take input from the command line while running the animation steps. In the future, we would work on implementing multithreading so that issue does not arise. The attempt and work can be seen in the Github repository.
    - We would like to make either the Tkinter GUI fully functioning so that there is no need for the command line or use POST and GET functions in JavaScript to connect the frontend web application with the backend python algorithm.
    - We would ideally like to work on improving our transfer operation, as the time for completion is not currently under a minute as originally promised.
    - We were not able to fully implement the balance and SIFT operations. We worked to write the algorithms and our attempts/tests are located in the Github repository.

10.self-evaluation report.

All team members tried their best. We put the same amount of effort and time with passion.