Andersson Jacob, Johansson Tim, Wikström Leo, Åsbrink Anton

2018-04-09

Implemeation Plan

Assigment in the course PA1435 Objektorienterad Design

| **Author Name** | **Social Security Number** | **Thinking** | **Writing** |
| --- | --- | --- | --- |
| Andersson Jacob | 960221-8134 | 25% | 0% |
| Johansson Tim | 970718-3472 | 30% | 35% |
| Wikström Leo | 970523-6611 | 30% | 35% |
| Åsbrink Anton | 970428-0115 | 15% | 30% |

# **Story Points and Priority List**

Higher number = higher priority

#### **Lock or Unlock (UC9)**

Priority: 2

Useful to prevent users from manipulating the same object at the same time, but is not strictly needed for the program to function.

Story points: 10

Since this use case is only a status of a good, the system will only have to change the state the good is in. (pretty basic)

#### **Remove Warehouse (UC15)**

Priority: 6

A needed function if a warehouse is no longer available due to for example sale or destruction.

Story points: 11

This use cases will be pretty easy since the system only needs to remove these certain objects from a list.

#### **Remove Good (UC12)**

Priority: 7

A needed function if a good is no longer available due to for example sale, theft or destruction.

Story points: 11

This use cases will be pretty easy since the system only needs to remove these certain objects from a list.

#### **Remove Truck (UC21)**

Priority: 6

A needed function if any truck is no longer available due to for example sale, theft or destruction.

Story points: 11

This use cases will be pretty easy since the system only needs to remove these certain objects from a list.

#### **Track Trucks (UC4)**

Priority: 5

Important function for the program but not completely required since trucks could be manually observed by selection.

Story Points: 12

This use cases won’t need much from the system, some get and sets of information would suffice.

#### **Direct Truck (UC7)**

Priority: 4

Assigning trucks to warehouse is useful for truck organization but not required if there is only one warehouse or all warehouses has at least one truck.

Story Points: 12

The system needs to change a status of the truck and set a good or section of a warehouse as it’s targeted destination.

#### **Select Warehouse (UC2)**

Priority: 4

ID for warehouses can be remembered or written down and if there is only one warehouse then this function is not required.

Story Points: 13

This use cases won’t be too difficult since the system only needs an ID from the user and then it searches after the certain object.

#### **Select Truck (UC5)**

Priority: 6

Selecting which truck to use allows control of multiple trucks at the same time and is required if there are more than one truck.

Story Points: 13

This use cases won’t be too difficult since the system only needs an ID from the user and then it searches after the certain object.

#### **Select Good (UC6)**

Priority: 7

Selecting goods and view their information is needed to manipulate them with the trucks.

Story Points: 13

This use cases won’t be too difficult since the system only needs an ID from the user and then it searches after the certain object.

#### **Edit Good (UC8)**

Priority: 3

If there are no mistakes and no important changes are needed with any goods then this function is not required.

Story points: 14

Very similar to adding a good for the system, with the only difference being that the system won’t need to add it to a list again.

#### **Add Warehouse (UC13)**

Priority: 5

Useful for any new warehouses that might be used, but otherwise warehouses can be hardcoded.

Story points: 15

This case only need to add a warehouse to an array. But some extra work is required in memory management when the array is full.

#### **Add Good (UC3)**

Priority: 7

Core function for adding new goods coming in to warehouse and is therefore required for the program to function.

Story Points: 15

This case only need to add a good to an array. But some extra work is required in memory management when the array is full.

#### **Add Truck (UC17)**

Priority: 5

Adding trucks is a necessary function for any new trucks, but the trucks can be hardcoded.

Story points: 15

This case only need to add a truck to an array. But some extra work is required in memory management when the array is full.

#### **Write Message (UC10)**

This function is not needed to run the warehouse as it just provides communication between users.

Priority:1

Story points: 25

This use cases require that we build a complete chat system where our system saves the time of posting a message, the message itself and who posted it. The system also has to order every message depending on what time it was posted.

#### **Read Message (UC11)**

Priority: 1

The same as “Write message” this is not necessary for primary functions.

Story points: 25

This use cases require that we build a complete chat system where our system saves the time of posting a message, the message itself and who posted it. The system also has to order every message depending on what time it was posted.

#### **View History (UC16)**

Priority: 3

This function is useful for tracking where goods have been moved and when, but it is not needed for an MVP.

Story points: 30

In this case we need to store all user actions and then be able to present them in a readable fashion.

#### **Log In (UC18)**

Priority: 6

A needed function to distinguish between a manager and a trucker.

Story points: 55

For login we have to implement a few security features for password management such as hashing.

#### **Register User (UC19)**

Priority: 3

To register users is needed only to differentiate between managers and truckers and to determine who has sent a message. To distinguish between user roles a single manager account can be hardcoded into the program making this use case less of a priority.

Story points: 60

The system needs to save information in a way that people can’t read this information. The system will have to hash or encrypt the information.

#### **Move Good (UC20)**

Priority: 4

Functions as a way to organize goods in warehouse. This is important since the customer explicitly asked for this.

Story Points: 80

Apart from the work needed to implement the graphical aspects of this use case, moving a piece of goods in the warehouse requires lots of checks to make sure the item is not placed in an illegible place. For example if the place is too warm or too cold, or the item is too small or too large.

#### **Show Map (UC1)**

Priority: 3

Useful to give a visual representation of where warehouses are located and makes selection of them easier. However warehouses can be selected with only their ID and therefore this function is not critical to the program.

Story Points: 100

We have to create and display a map that shows everything inside a warehouse, a visualisation of every good in the warehouse and the possibility to select goods from this map. This also means that we need to add a positioning system for all the goods in the warehouses.

# **Iterations**

### **Iteration 1:**

Add Warehouse, Add Good, Add Truck, Select Warehouse, Select Good, Select Truck,  Remove Warehouse, Remove Good, Remove Truck, Track Trucks

MAX: 16

MIN: 11

AVERAGE: 13

NR OF STORY POINTS: 130

### **Iteration 2:**

Edit Good,  Direct Trucks, Show Map, Lock or Unlock

MAX: 100

MIN: 10

AVERAGE: 34

NR OF STORY POINTS:136

### **Iteration 3:**

Log In, Register User, View History

MAX: 60

MIN: 30

AVERAGE: 48.3

NR OF STORY POINTS:145

### **Iteration 4:**

Move Good, Write Message, Read Message

MAX: 80

MIN: 25

AVERAGE: 43.3

NR OF STORY POINTS: 130

For the first iteration we wanted to keep it as simple as possible while still creating a viable product. We concluded that in order to function, the program would need to be able to handle adding, selecting, and deleting of warehouses, goods, and trucks.

  For the second iteration we opted to include a bit more depth, with options to track and direct trucks, as well as to edit goods. At this point we would also like to implement the 2D map, as this was one of the things for which the company mentioned they were particularly interested in finding a solution. We feel that this iteration would require approximately the same amount of work as the first one, and would move the project forward significantly, keeping a good distribution of story points and a satisfactory development speed.

  Iteration three would include a login system for users as well as an option to view the history of actions the user has taken. This would allow the program to differentiate “truckers” and “managers”.

  Finally, the fourth iteration of the program would add the feature of moving goods around the map with the cursor keys, and the ability for users to write and read messages. These features would take up a lot of effort and the program would work without them, which is why we’ve placed them in the planned final iteration.

  All in all we feel that this division of use cases has divided the work in a good way between the different iterations of the program.

The velocity of the team (in story points/week): All the team members agree that the first implementation should not take more work than a week; to make some room for errors we have decided to set our velocity to 75 (~18 per person).