Dev Ops Project Documentation

*by Bríana Lee, Ciarán Toman, Dylan Irwin, Kevin McGonagle, Kyle Doherty,*

*Rory Friel & Shaun Haugh.*

**Original Requirements For The Project**

**Use the following technologies in the development of your solution:**

* An automated pipeline MUST be demonstrated. Without this a maximum of 40% can be awarded.
* GitHub to provide version control for the project, each team member must commit his/her code at least once a week.
* Install jacoco for code coverage.
* Agile Scrum & Jira for the management of meetings and backlogs, Facebook cannot be used.
* Use junit tests to test your code; make a suite of tests to run before each commit to github.
* TestNG may be used if desired.
* Use Javadoc to comment your code.
* Pay careful attention to exceptions; how are they handled, which are caught and which are thrown.
* Ensure that at least one of your own api’s and one web service are included in the design to demonstrate an understanding of service composition.
* Consider secure coding, be clear in your write up.
* Consider performance – allow time to run a performance testing tool such as jconsole or jrat.
* The main documentation may be submitted by any one member of the team – however the conclusions section is unique to all team members and will be marked as such. This constitutes a large portion of the grades.

**Considerations:**

* Framework(s)
* http://zeroturnaround.com/rebellabs/top-4-java-web-frameworks-revealed-real-life-usagedata-of-spring-mvc-vaadin-gwt-and-jsf/
* Database
* Database persistence technology
* Presentation Layer (jsf, jsp)
* Define the Business Requirements
* Named queries and database triggers for security
* Regex for cleansing and validation of data before sending to the database

## **Customer Requirements:**

Group 3: Eduardo Perez The customer would like an on-line system to enter customer details for tanks and trains. The system would then calculate the cost based on number of evil minions and weight of rockets. The system should be clean and simple. The system needs to take into account the usual details about dimensions and weight. The client needs detailed management of the number of tanks and trains in any given destination at any time. Each destination can only take a maximum of 5 tanks or 2 trains at any time. The administrator should be able to access detailed information and edit as appropriate. Once the client enters details it should not be able to be changed by the client.

**Each of your customers are trying to take over Minion Island. They each need a system to manage their vehicles. Gru and Dr. Nefario are trying to protect the minions and their island.**

**Some Questions Asked By Kevin About The Project**

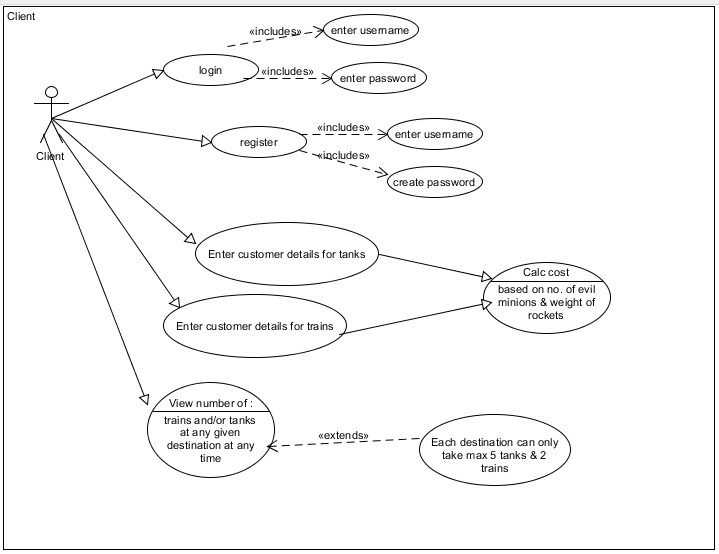
This is a number of questions I asked Ruth during class about or project, some questions may be useful during the design of some of your classes, others may not be necessary for the project.

* **Does the Cost or budget matter?** Not yet.
* **How does the user interact with the program?** Through a tablet or PC using a web interface.
* **How many docking location?** One for now.
* **Is there a time constraint?** Not at the moment.
* Tanks and Trains travels at the same speed, so transit is the same time for both, this might be changes later.
* **Do we care how long it takes to get to the island?** No unless this messes up docking
* **What does the tracking system have to be like?** It must show if the vehicle is either at the Source, in transit or at the destination.
* It takes an hour for from arrival until unloading is finished.
* Either five tanks or two trains can arrive at a single destination at the one time, not both.
* There are two types of minions tall thin ones and short thick, tall thins ones are half the width of the short ones. We are only using the tall thin ones for now.
* **Weight a tank/train can hold?** We are only looking at the dimensions for now, not sure if this will change to include weight.
* There are two types of rockets an easy rocket and a hard rocket. The easy rocket is half the dimension of the hard rocket.
* A train is three times the size of a tank.
* How many minions can fit in a tank or train? 30 can fit in a train 10 can fit in a tank, divide by two if the small minions are used.
* A tank requires 1 minion to drive, 1 per rocket to shot, 1 per rocket to load.
* A train only requires one driver.
* 3 minions = an easy rocket, 6 minions = hard rocket (cannot recall if this refers to dimensions or for loading and moving purposes).
* **Who determines when it was delivered?** Consider CA from the beginning.

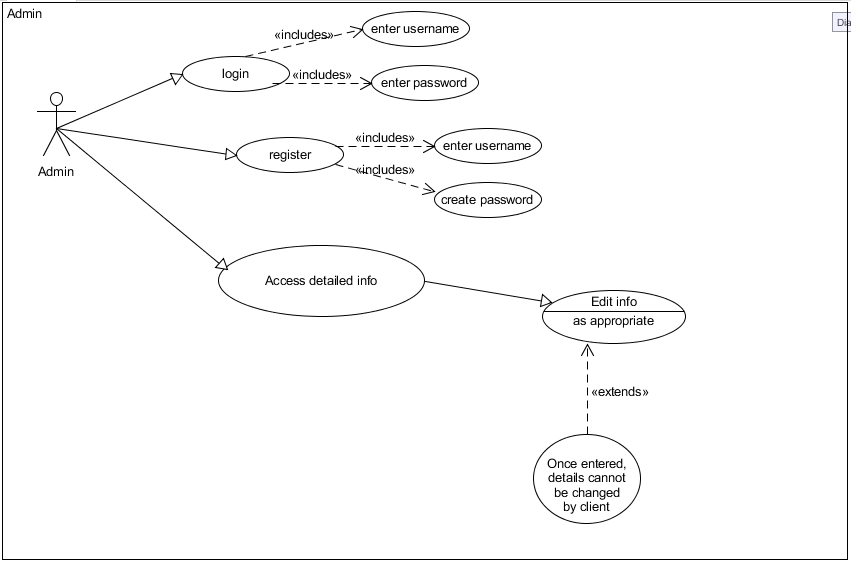
**Class Diagram**

Put class diagram and small description here.

**Use Cases And Descriptions**

**Client Use Case**

**Admin Use Case**



**Use Case Descriptions**

|  |  |
| --- | --- |
| Use Case | Login |
| Objective | To allow either the admin or client to login to the online system |
| Precondition | The user already has an account and login details are known |
| Main Flow | 1. Get username 2. Get password 3. User logged in |
| Alternative Flow | 1. User details does not exist. Inform the user and exit use case |
| Post Condition | User is logged into their account |

|  |  |
| --- | --- |
| Use Case | Register |
| Objective | To allow the admin or client to register for an account |
| Precondition | The user does not have an account |
| Main Flow | 1. User chooses a username 2. User chooses a password 3. The account is created and details are sent to the database |
| Alternative Flow | 1. Username already chosen. Inform the user and exit use case |
| Post Condition | User’s account is now created |

|  |  |
| --- | --- |
| Use Case | Client – Enter customer details for tanks and trains |
| Objective | To allow the user to enter the customer details for tanks and trains |
| Precondition |  |
| Main Flow | 1. User enters customer details for tanks 2. The system calculates the cost based on no. of evil minions and weight of rocket along with the details entered for the trains |
| Alternative Flow |  |
| Post Condition | Customer details are added for the tanks and the trains |

|  |  |
| --- | --- |
| Use Case | Client – View number of tanks and/or trains |
| Objective | To allow the user to view the number of trains and/or tanks at any given destination at any time |
| Precondition |  |
| Main Flow | 1. User chooses “To View” option 2. No of trains and/or tanks is displayed on screen depending on which type the user wishes to see |
| Alternative Flow | 1. Each destination can only take max 5 tanks and 2 trains |
| Post Condition | Number of trains and/or tanks is shown to the user |

|  |  |
| --- | --- |
| Use Case | Admin – Access detailed info |
| Objective | To allow the admin to access detailed info |
| Precondition |  |
| Main Flow | 1. User wishes to access detailed info 2. Detailed info is shown to the user 3. This info can be edited as appropriate by the admin |
| Alternative Flow | 1. Once the details have been entered, they cannot be changed by the client |
| Post Condition | Detailed info is displayed to the user |