MET cS665 Summer 2 2018

Abel Gancsos

Abel Gancsos productions | Hawthorne, NJ

AMGDeliveryDispatch

2018

Contents

1 Application Description 3

1.1 Implementation Details 3

1.2 Product Types 3

1.3 Requirements 3

1.3 Assumptions 3

1.5 Prerequisites 3

2 Assignment Tasks 5

2.1 Implementation Description 5

2.2 UML Class Diagram 6

2.3 UML Sequence Diagram 7

Bibliography 8

Appendices 9

Appendix A – Setup 9

Appendix A – Building project 9

Appendix B – Running executable 10

# 1 Application Description

## Implementation Details

*The following details are also available in the README.md file*

This utility, AMGDeliveryDispatch, is a system that helps assign delivery orders to appropriate vans based on certain criteria once the purchase has been completed. The system starts by taking in a certain number of orders containing various number of items from different customers at different shops. The assignment process then scans the type of items in the orders, checks if it's the customers birthday, calculates the total distance of the delivery, and then checks if there's an available driver. To ensure appropriate delivery time and proper state of the product upon delivery, the assignment process also checks for any traffic that could occur while in transit. If an order is not able to be dispatched due to lack of drivers or unavailable resources, the order waits until a proper resource has become available. Once the driver has been assigned, the customer is then notified of a scheduled delivery.

## Product Types

\* Food

\* Cold

\* Warm

\* Flower Bouquets

\* Gift chocolate boxes

## 1.3 Requirements

* If a cold meal is to be delivered more than 2 miles away, it must go to a van with a freezer.
* If it's a customers birthday and they have an order, they must receive a gift chocolate box.
* When there is a lot of traffic, ALL cold foods must go to vans with freezers.

## Assumptions

* The order has already been completed and paid for by the time the process enters into this utility.
* In addition to notifiying the customer of an expected delivery, the driver is told where to pick the order up.
* Although the system would use GPA coordinates to calculate driver location, the distance is calculated by random numbers in miles for simulation purposes.
* More product types can be added in the future.
* Food types can be of any type.
* Flower bouquets can include any flower combination.
* Chocolate boxes may contain different chocolate types.
* Drivers and products are predefined and the user doesn't control this data.
* One tick in the simulation is the same as travelling one mile in distance.
* An order can have multiple items, but no more than one item of the same type.
* The total distance of a delivery is considered the distance of the driver from the store plus the distance of the customer from the store.
* An order doesn't become a delivery until it is ready for dispatch.

## 1.5 Prerequisites

* C++ compiler with C++11 capability
* CMake version 3.11 or higher
* XCode command-line tools (MacOS only)
* Visual Studio 2017 or higher (Windows only)

# 2 Assignment Tasks

## 2.1 Implementation Description

*The following details are also available in the README.md file*

## 2.2 UML Class Diagram

## 2.3 UML Sequence Diagram

# Bibliography

Döring, A. (2016, 11 07). *minimal\_cmake\_example*. Retrieved 07 03, 2018, from GitHub Web site: https://github.com/krux02/minimal\_cmake\_example

Google Corporation. (2018, 06 19). *Google C++ Style Guide*. Retrieved 07 03, 2018, from Google GitHub styles: https://google.github.io/styleguide/cppguide.html

Kitware Corporation. (n.d.). *Download*. Retrieved 07 03, 2018, from CMake Web site: https://cmake.org/download/

Lucid Software Corporation. (2017). *Class Diagram Tutorial*. Retrieved 7 25, 2017, from Lucid Software Corporation Web site: https://www.lucidchart.com/pages/uml/class-diagram

Microsoft Corporation. (2010). Visio. (2010). Redmond, WA, USA.

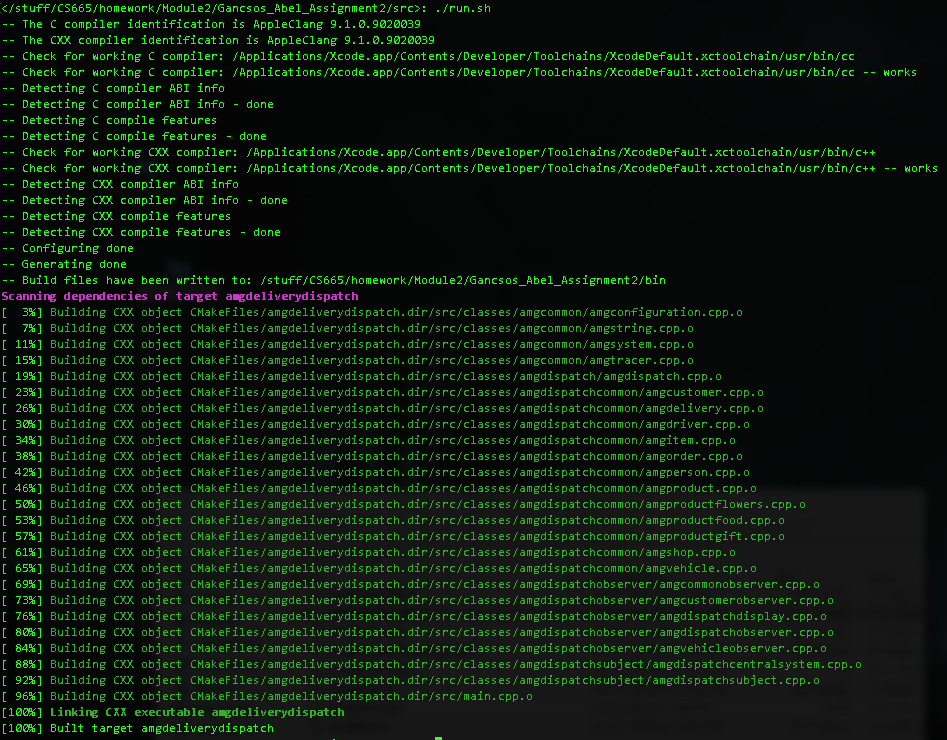
Oracle Corporation. (2017). VirtualBox. (5.1.18). Redwood City, CA, USA.

# Appendices

## Appendix A – Setup

The project can be build using the run.sh or run.bat scripts, depending on the platform. Both of these scripts will run cmake to generate the CMakeFiles and then run the make command in the bin directory. Alternatively, the project can be built manually using the CMakeLists.txt file that comes with the package.

## Appendix A – Building project



## Appendix B – Running executable

