**CS6310 – Software Architecture & Design**
**Assignment #5 [250 points]: Grocery Express Project – Group Implementation (v1)**
**Spring Term 2023 – Instructor: Mark Moss**

## Submission

- This assignment must be completed as part of an approved group.
- One member of each team must submit:
    1. ***A functioning version of the application*** in a format agreed upon by your evaluating TA (the format should have been agreed upon during the previous assignment);
    2. All source code (Java and otherwise) and links to external packages and other resources used to create your application in a file named ***source_code.zip***;
    3. The updated "as-is" UML Class and Sequence Diagrams, and any other accompanying documentation, in a file named ***design_docs.pdf***; and,
    4. An MP4 video (or other reasonably comparable format) with a ***total viewing time of 15 minutes*** or less named ***mts_video.mp4*** that presents an overview of your application. You are also welcomed to divide your work into smaller videos to keep the file sizes more reasonable.
- For this project, you should select reasonably named files, and parts (2) and (3) must be submitted via Canvas. The virtual machines and videos for parts (1) and (4) can be very large, so a link to an external but accessible storage site (e.g., Google Drive) should be submitted via Canvas for those deliverables.
- Submit your answers via Canvas.
- You must notify us via a private post on Piazza BEFORE the Due Date if you are encountering difficulty submitting your project. You will not be penalized for situations where Canvas is encountering significant technical problems. However, you must alert us before the Due Date – not well after the fact.

## Scenario

The clients are excited about your design proposals as we enter the final phase of the project. During this phase, you will: (1) finalize, implement and deploy your team's design; (2) ensure your design documents are consistent with the finished application; and, (3) prepare a short video presentation for the clients and evaluators that demonstrates how your application functions.

## Disclaimer

*This scenario has been developed solely for this course. Any similarities or differences between this scenario and any of the programs at Georgia Tech programs are purely coincidental.*

## Deliverables

This assignment requires you to submit the following items:

1. **Functioning Application [90 points]:** You must submit a working application that satisfies the client's requirements as presented throughout all the assignments, and as clarified further through all associated Office Hours and Piazza posts. Your system must incorporate all of the design requests as described in all of the previous assignments and below. You are allowed to submit your application in a format that has received prior approval from your evaluating TA.

From a grading perspective, the points for implementation are distributed as follows:

- 10 points for the clarity and "ease-of-use" (whether it's a GUI or a CLI) of your system, as well as the extent to which the interface supports the normal flow of operations during each simulation run.
- 20 points for the overall correctness of your monitoring system to include the accuracy of the simulation state.
- 60 points for the correctness of all your proposed modifications.

2. **Source Code [50 points]:** You must also provide copies of the actual source code that your team has developed, along with references to all of the external packages, libraries, frameworks, services and systems used to develop your application. The overall architectural and design quality of your system will be evaluated, to include factors such as good use of abstraction of modularity, reasonable documentation and descriptive class, variable and method names, etc.

3. **Design Documentation [70 points]:** You have already developed numerous design documents: Class Diagrams, Sequence Diagrams, Object Diagrams, and possibly other forms such as State Charts and/or Collaboration Diagrams. For this assignment, you must select and provide the most appropriate UML-compliant Structural and Behavioral Diagrams that describe the final design of your application. There are no more pending or additional requirements after this assignment, so your final design documents should be as accurate and consistent with the final implementation of your application as possible. And especially since you have permission to use languages and system stacks other than Java, you need to ensure that your design documentation is consistent with your actual implementation.

   Please clearly designate which version of UML you will be using – either 2.0 (preferable, and the latest OMG-accepted version: https://www.iso.org/standard/52854.html) or 1.4 (the latest ISO-accepted version). There are significant differences between the versions, so your diagrams must be consistent with the standard you've designated. The design documents can be submitted as different documents but must be named clearly and accurately.

   From a grading perspective, the points for documentation are distributed as follows:

   - 25 points for your UML Structural Diagrams, which should include (at a minimum):
     - A Class Diagram that describes system as it is actually implemented
     - A Deployment (or similar) Diagram that shows how your system is actually implemented, especially in integrating external packages, services and/or systems
   - 25 points for your UML Behavioral Diagrams which should include (at a minimum):
     - Sequence, State Machine (or similar) Diagrams the give a reasonable overview of how the system is implemented
     - Use Case Diagrams for all functionality added per your design proposals
   - 20 points for the overall quality and presentation of your design documentation, including the addition of various diagrams that add significant value for the clients in understanding and maintaining your system into the future. *Do not simply create lots of diagrams to attempt to*

*collect points – this is definitely a "quality over quantity" issue.* Ensure that new diagrams and/or documents are well formatted and add significant value.

4. **Demonstration Video [40 points]:** Your team must submit a video demonstration of the prototype that your all have developed. You can either submit a YouTube link (preferred) or an MP4 attachment in Canvas. If submitting a video attachment, try to ensure that the total size doesn't exceed ~200MB, which should cover approximately 20 minutes for a 720p (HD) with 1 Mbps bitrate (or 24fps) quality video. You are also welcomed to divide your work into smaller videos to keep the file sizes more reasonable.

   Please take these values as approximations – the real file sizes will depend greatly on other factors such as frame rates, encoding types, etc. Also, we realize that not everyone wants to "make it to the big screen": faces are welcome, but there's no requirement for your faces to be displayed during the video. A screen capture of your system in action, along with a clear, audible English-language audio track, will be sufficient.

   Your video should be organized to give a clear demonstration of your system in action, including a well-considered test case/scenario that allows you to show how your interface and supporting components operate; and, how your overall system meets the client's requirements. And your video should highlight your design modifications. There's no need to spend significant amounts of time demonstrating the capabilities that already existed in the system – focus on the aspects of your system that have been changed, upgraded, etc.

### Writing Style Guidelines

The style guidelines can be found on the course Udacity site, and at:
https://s3.amazonaws.com/content.udacity-data.com/courses/gt-cs6310/assignments/writing.html

The deliverables should be submitted in the appropriate formats (OVA, ZIP, JAR, PDF, etc.) with file names such as ***source_code.zip***, ***design_docs.pdf*** and ***mts_video.mp4***. You should also use a similarly clear and simple name structure if you need to submit a file that we haven't explicitly listed here. Ensure that all files are clear and legible; points will likely be deducted for unreadable submissions.

### User Interface and Error Checking

Your application must provide a reasonable interface, whether you've chosen to implement a Graphical User Interface (GUI) or a Command Line Interface (CLI) to support your application. An extremely complex interface is not required and is not the intent for this course. A simple interface that provides the required functionality will be sufficient.

- Even if you implement a GUI, you are still permitted to have Command-Line Interface (CLI) capabilities with your application to support development, maintenance, troubleshooting and execution functionality. Your GUI, however, must be able support the major functionality required for a user to manage – and view the state of – a simulation run. Your CLI is allowed to provide "behind the scenes" support for the GUI as needed.

- We are giving your team fairly wide latitude in selecting a framework to support the development of your user interface, given that you are responsible for installing and configuring that framework on the course-approved Docker Container (or an alternate pre-approved platform) for submission.

- Your application must support the requirements previously described, in addition to any updates listed below. This is absolutely critical to our ability to evaluate your application thoroughly.

**Closing Comments & Suggestions**

We (the OMSCS 6310 Team) will conduct Office Hours where you will be permitted to ask us questions in order to further clarify the client's intent, etc. Also, the TA who will evaluate your final submission will be pre-assigned to your team. You can communicate with them if you have questions related to application design, implementation, deployment, etc.

**Quick Reminder on Collaborating with Others**

Since this is a group project, you may (and should) communicate freely with all of your group members. However, your group is not allowed to communicate with any other groups while working on this project, including outside personnel or consultants. Please use Piazza for your questions and/or comments and post publicly whenever it is appropriate. If your questions or comments contain information that specifically provides an answer for some part of the assignment, then please make your post private (your group members and OMSCS6310 TAs/Instructors only) first, and we (the OMSCS 6310 Team) will review it and decide if it is suitable to be shared with the larger class. Best of luck on this assignment, and please contact us if you have questions or concerns.