**Software Requirements and Design Document**

**For**

**Group <6>**

Version 1.0

**Authors**:

Jay Daniels

Marlan McInnes-Taylor

Keaun Moughari

Daniel Riley

Hayden Rogers

# Overview

The

*Give a general overview of the system in 1-2 paragraphs (similar to the one in the project proposal).*

# Functional Requirements

*List the* ***functional requirements*** *in sentences identified by numbers and for each requirement state if it is of high, medium, or low priority. Each functional requirement is something that the system shall do. Include all the details required such that there can be no misinterpretations of the requirements when read. Be very specific about what the system needs to do (not how, just what). You may provide a brief design rationale for any requirement which you feel requires explanation for how and/or why the requirement was derived.*

# Non-functional Requirements

*List the* ***non-functional requirements*** *of the system (any requirement related to security, safety, software quality, performance, reliability, etc.) You may provide a brief design rationale for any requirement which you feel requires explanation as to how and/or why the requirement was derived.*

# Use Case Diagram

*This section presents the* ***use case diagram*** *for the system under development. The use case diagram should contain all the use cases and relationships between them needed to describe the functionality to be developed. (If you discover new use cases between the two iterations, update the diagram for your second iteration).*

# Class Diagram

*This section presents a high-level overview of the anticipated system architecture using a* ***class******diagram****. It shows the* ***fundamental objects/classes*** *that must be modeled with the system to satisfy its requirements and* ***the relationships*** *between them. Each class rectangle on the diagram* ***must also include the attributes and the methods of the class*** *(they can be refined between iterations). All the* ***relationships between classes and their multiplicity*** *must be shown on the class diagram. The classes specified in this section are those directly derived from the application domain (the* ***entity classes****).*

# Operating Environment

*Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.*

# Assumptions and Dependencies

*List any assumed factors (as opposed to known facts) that could affect the requirements stated in this document. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project.*