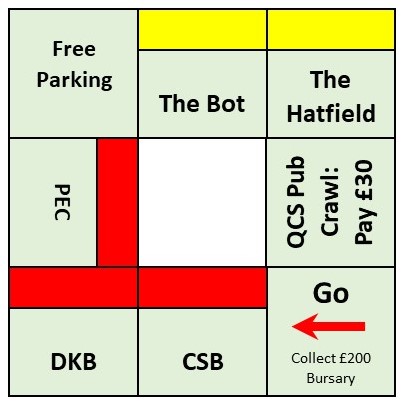
****

**Monoversity: An Undergraduate’s Experience at Queen’s University, Belfast**

Project Design and Implementation Summary Report

Prepared by Queen’s University, Belfast

Group G19: Compiler

March 2018

For Module Ref.: CSC7053 Software Engineering

Document Control

|  |  |  |  |
| --- | --- | --- | --- |
| **Document:** | Project Design and Implementation Summary Report | | |
| **Project:** | Monoversity: An Undergraduate’s Experience at QUB Monopoly Game | | |
| **Module:** | CSC7053 Software Engineering | | |
|  |  | | |  |
| **Revision:** | For Issue | | | |
| **Date:** | 05 March 2018 | | | |
| **Report prepared, reviewed and approved by:** | | | |  |
| **Signature:** | | **Name and**  **Student No.:** | **Signature:** | **Name and**  **Student No.:** |
|  | | Paddy Burns  18596037 |  | Colm McGoldrick  40006755 |
|  | | Conor Leonard  40106357 |  | Justin Kelly  40214842 |
|  | | Catherine O’Hare  40108703 |  |  |

Contents

[1.0 Introduction 1](#_Toc508046865)

[1.1 Project Overview 1](#_Toc508046866)

[1.2 Scope of Work 1](#_Toc508046867)

[1.3 Project Delivery Programme 1](#_Toc508046868)

[2.0 Requirements Analysis 2](#_Toc508046869)

[2.1 Project Kick-Off 2](#_Toc508046870)

[2.2 Determination of Core System Functionality Requirements 2](#_Toc508046871)

[2.3 Game Guide 3](#_Toc508046872)

[2.3.1 Smickopoly Basic Rules: 4](#_Toc508046873)

[2.3.2 Gameplay: 4](#_Toc508046874)

**Appendices**

Appendix A Project Delivery Programme

Appendix B Property Values and Rent Costs

# Introduction

## Project Overview

The project team has been commissioned to develop a monopoly-based game with a theme that is connected to Queen’s University, Belfast (QUB). This document provides an overview of the processes implemented in the production of the required monopoly-based game, and the outputs from such during the development process.

## Scope of Work

The overarching scope of work has been derived from the Project Specification which outlined a number of key phases of the development process that are required to be followed, and subsequently demonstrated in the Summary Report. This specifically involves the following:

* Requirements Analysis: incorporating a game guide, and subsequent Use Case Diagram and associated Use Case Descriptions based on Unified Modelling Language (UML) principles;
* System Realisation: conceptualised using UML Sequence Diagrams;
* System Design: idealised through a UML Class Diagram;
* Design Process: providing details on the software testing process implemented, and a record of team meetings;
* Game Development: essentially involves the delivery of a working system that meets the specified requirements.

The aforementioned is subsequently detailed in the following sections of this report.

## Project Delivery Programme

Insert details regarding the project management aspects.

Appendix A: Programme.

# Requirements Analysis

In undertaking an analysis of the core project requirements, the following key sequences were carried out:

* Initial project kick-off meeting to conduct a high-level discussion on the Monoversity Project Specification, agree on a game theme, and initial project progression;
* Analysis of the customer requirements as detailed in the Monoversity Project Specification to determine core system functionality requirements;
* Development of a “Game Guide” that included a conceptualised board layout and game rules;
* Production of a Use Case Diagram and associated Use Case Descriptions.

## Project Kick-Off

The key outcome from the project kick-off meeting with respect to the Requirements Analysis process was the agreement among the team on a theme for the game. The agreed approach was to develop a light-hearted game with the core theme of a student’s experience at attending QUB, including the social aspects of the experience. The working title “Monoversity: An Undergraduate’s Experience at QUB.

## Determination of Core System Functionality Requirements

The Project Specification was analysed in order to identify the core requirements in terms of the overall function of the system that would subsequently inform the system development process. Outlined in Table 2.1, these core requirements would also form the basis against which the operational system would be benchmarked against during the testing phase(s) of the development process.

**Table 2.1: Summary of Core System Functionality Requirements**

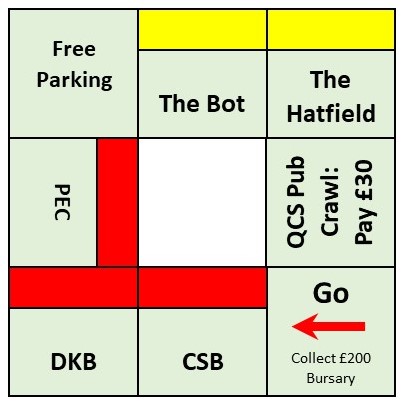
| **Ref.** | **Requirement** |
| --- | --- |
| R1 | The game has two players, who start with the same amount of money. Their names should be entered. |
| R2 | The players take turns. |
| R3 | They throw a virtual die. |
| R4 | There is a start square, which players land on or pass to pick up their salary/grant money/funding. |
| R5 | There are two property groups, one comprising two properties and another comprising three. |
| R6 | The properties in the two-property group are more expensive than the ones in the three-property group. |
| R7 | Before you can develop a group by building on it, you must own the *whole* group, |
| R8 | The game will have a university- and specifically a Queen’s-related theme. |
| R9 | Three ‘houses’/*labs* are needed before you can build or establish a ‘hotel' (or whatever you decide it is). |
| R10 | Players taking a turn are told where they have landed and what their obligations or opportunities are. |
| R11 | Where appropriate, they may indicate their choice of action. |
| R12 | If a player’s finances have changed, the system indicates the reason for the change and announces the player’s new bank balance. |
| R13 | When one player goes bankrupt, the other is declared the winner. If one player no longer wants to play, the game ends. In both cases, the players’ bank balance is given. |

## Game Guide

### Board Layout

The development of a board layout and associated game guide/rules was carried out in context of the key project requirements outlined in Table 2. 1, with the board layout presented in Inset 2.1 (below):

**Inset 2.1: Conceptualised Game Board Layout**

****

The project specification required the provision of a Go Square and two distinct property groups (one comprising two property squares, the other three property squares). The QUB theme is enforced when complying with this requirement through the inclusion of QUB property squares (Red squares), namely the Computer Science Building (CSB), David Keir Building (DKB) and the Physical Education Centre (PEC). The more expensive yellow property group represents bars, the social aspect of attending university.

A ‘Free Parking’ and ‘Queens Computing Society (QCS) Pub Crawl’ square (replacing the traditional ‘Income Tax’ square in Monopoly) were included, under the terms of the project specification (i.e. “*Your game will have many of the features that players of the original Monopoly would expect, but in a much simpler form”)* to give a more rounded game experience.

### Smickopoly Basic Rules

* The aim of the game of the game is to become the wealthiest player through buying, renting and selling property.
* The game is for two players, with each player prompted to enter their name before the start of the game.
* Each player starts off with £500 in the bank.
* Before the game starts both players roll the virtual dice. The player who rolls the highest number gets to move first and the game is started.
* The players then take turns to roll the virtual dice. One dice is rolled, and the moves of the players are made by the system.
* The game is over either when one player is bankrupt (can’t pay rent or a fine) or when one player decides to terminate the game when they are asked if they wish to roll the dice when it’s their turn.

### Gameplay

*Landing on Squares (General)*

* When the player lands on a square the system will display the name of that square and any functions/details associated with it.

*Pass Go*

* Both players start the game on the ‘Go’ square.
* Each time a player lands on or passes ‘Go’ following the start they receive £200.00.

*Pub Crawl*

* When a player lands on the ‘pub crawl’ square they have to pay £30.00 to the bank.

*Free Parking*

* No changes are made to the players balance or status.

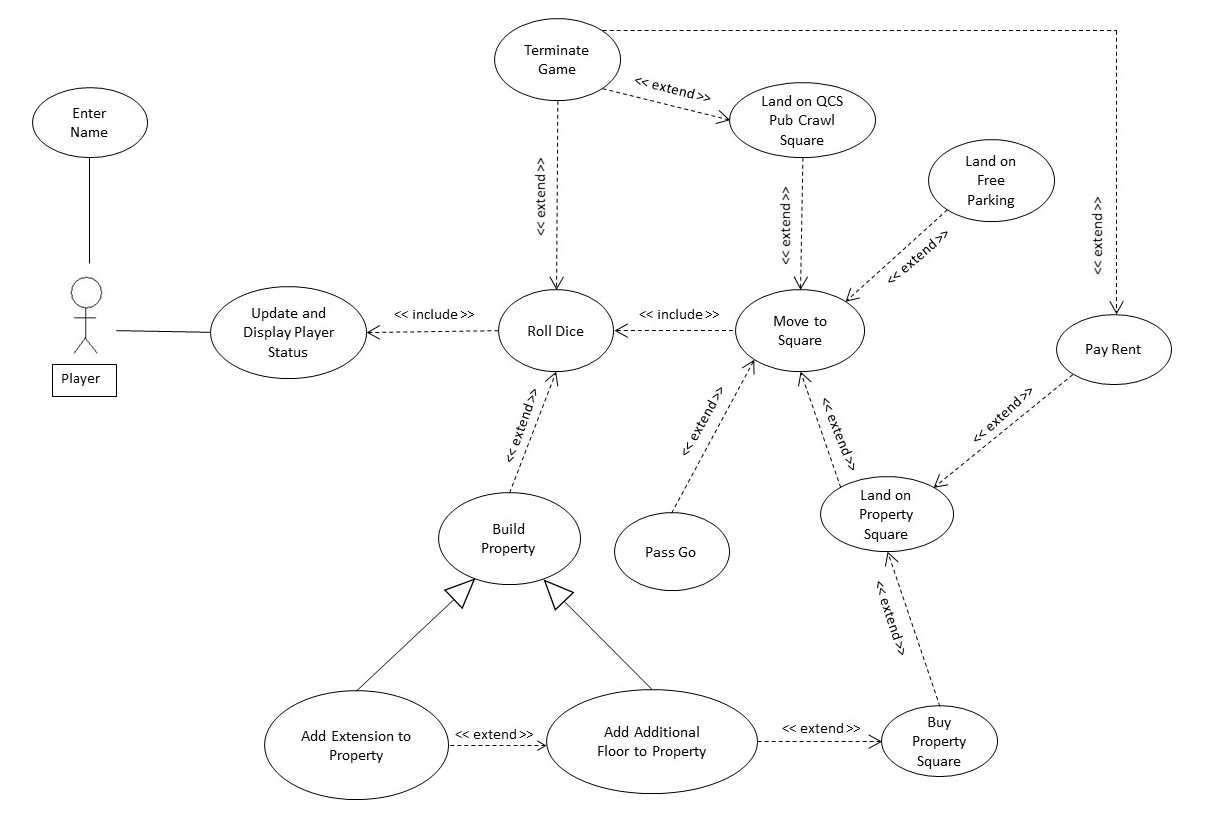
*Additional functions associated with Property Squares*

* When a player lands on a property square the system will confirm whether the property is owned or not.
* If a property square is not already owned, the player will have the option to buy that property square if they have the required amount of funds.
* If the property square is already owned, the system will display whether there is any development on the site (i.e. extra floors or extensions), the amount of rent due, the player’s current balance and the balance after the rent has been paid.
* The amount due for the rent is doubled when a player owns all the properties in that group and if there has been no development.
* If a player has insufficient funds, the system will declare that player bankrupt.
* Development is only allowed on the properties when the player owns all the properties in that group.
* Development is only allowed equally; a player cannot build more than one floor on any one property of any colour until one floor has been built on every property of that group. The player could then begin to build on the 2nd row of floors etc.
* Extensions can only be added when three floors have been added to each property square in that group.
* Costs of properties and rent values can be found in Appendix B.

## Use Case Diagram and Descriptions

The Use Case Diagram that outlines the system pathways which reflect the core requirements is outlined in Inset 2.2 overleaf.

**Inset 2.2: Use Case Diagram**



**APPENDICES**

**Appendix A: Project Delivery Programme**