Date: __/__/__

UniSim

REQUIREMENTS

TEAM6 Game Studios

Work Undertaken by:

Thomas Koukouris
Adam Khan
Oliver Herron
Sam Jordan
Nathan Hopper
Fergus Irvine

SINGLE STATEMENT OF NEED

The main goal with **UniSim** is to create a game that is fun and simple to play while offering enough depth to challenge experienced players. The game should be intuitive to learn, immersive and engaging so that it encourages repeat play attracting a broader audience.

DOCUMENT PURPOSE

This document provides a detailed description of the game UniSim. It outlines user needs and system requirements, which are split under functional and non-functional requirements. The document will serve as guide throughout the development stages of the game and shall be regularly revised to meet updated client requirements.

REQUIREMENTS ELICITATION & NEGOTIATION

The initial phase of requirements elicitation for the UniSim game began with the client's **product brief**, which outlined the core elements of the game. Following this, Team6 held a group meeting where the project's requirements were discussed and a list of questions was compiled. A subsequent **meeting with the client** took place a few days later and further details and clarifications were acquired. Afterwards, Team6 had a second meeting to finalise the complete set of requirements. This document is the consolidation of these meetings and will serve as the foundation of the game development and will be shown to the client for confirmation.

GAME OVERVIEW

UniSim is a **single player simulation game** where players are tasked with building and managing a university campus. The goal of the game is to achieve maximum **student satisfaction score** by the end of the time.

The player begins with an empty map and are offered options to construct several buildings which cannot be altered after. Once the game begins, the player must use their resources available to them wisely to achieve the best student satisfaction possible by striking a balance between different factors. These factors include but are not limited to: building types, building proximity to relevant facilities, building and infrastructure maintenance, links between campus areas, strategic placement of natural elements.

The player is presented with special events that occur randomly. The events may be positive, negative or have no impact based on the players reaction (or non-reaction). The player must decide how to allocate and acquire resources during the game to achieve the best outcome.

The game has a time limit of **5 minutes**, then the game stops and the score is displayed as well as a complementary message reflecting performance. While the game cannot be "failed", the message will indicate how well the player did.

It is assumed that players will have basic familiarity with simulation games and how game interfaces work. A short tutorial or interface explanations should be provided highlighting certain aspects of the game in the beginning.

GAME PURPOSE & TARGET AUDIENCE

The aim is to provide an engaging game that will attract players primarily in the ages of **16-20+**. The game should encourage players to engage with Computer Science by highlighting that it was developed by CS students. This serves as a motivational element to inspire interest in the field. This game will serve as the initial minimal viable product (MVP). Depending on its success, the client will provide funding for the development of a more advanced version.

USER REQUIREMENTS (URs)

The following table outlines the user requirements, detailing the **key features players shall expect** in UniSim, such as an intuitive interface, map interaction, resources display, event response, and accessible design to ensure an engaging gameplay experience.

ID	Description	Priority
UR_GAME_LOADING	The player shall be able to swiftly load the game.	Shall
UR_MAIN_MENU	The player shall be presented with a main menu containing the main functionalities of the game along with settings and exit button.	Shall
UR_UX	The game interface shall offer a pleasant user experience. Buttons shall be clearly visible and visually pleasing. Text shall have an appropriate theme font and a suitable size and placement.	Shall
UR_DIFFICULTY	The player may be given 3 difficulty settings to choose for at the start of the game.	May
UR_START_GAME	The player shall be able to start the game by pressing a button.	Shall
UR_PAUSE_GAME	The player shall be able to pause the game at any point.	Shall
UR_MAP	The player shall be able to view a 2D map representing the university campus.	Shall
UR_MAP_INTERACTION	The player shall be able to interact with the map by placing buildings and infrastructure.	Shall
UR_GAME_INTUITIVENESS	The game shall be intuitive to play, ensuring that the player can easily understand and navigate the interface with minimal explanations.	Shall
UR_TUTORIAL	The player should be provided with a tutorial that explains the map's features and game mechanics.	Should
UT_EXPLANATIONS	The player should be offered "hover-over" explanations of interface elements.	Should
UR_INTERFACE	The player shall have easy access and visibility to all important game information, such as student satisfaction, resources, time, and building counter.	Shall
UR_BUILDINGS	The player shall be able to select from a list of building options to place on the map.	Shall
UR_MAINTENANCE	The player should be able to maintain the buildings, e.g. library, café.	Should
UR_MAINTENANCE_ALERTS	The player should be alerted when maintenance is required.	Should
UR_EVENT_ALERTS	The player shall be alerted of events at predetermined times throughout the game and may be prompted to make a decision that could impact the game in some way.	Shall
UR_TIMER	The player shall be able to see how much game time is left. The player shall be able to see the remaining game time at all times.	Shall
UR_COUNTER	The player shall be able to track how many buildings and subtypes are placed on the map at any time.	Shall
UR_MAP_ELEMENTS	The player shall be able to see all buildings and game elements on the map.	Shall

UR_BUILDING_CONDITION	The player should be able to see the condition of buildings reflected visually on the map.	Should
UR_EVENTS	The player shall be offered actions to handle events as they occur.	Shall
UR_EVENT_MAP_DISPLAY	The player may be able to view events taking place on the game map.	May
UR_RESOURCES	The player shall be able to spend their resources on game elements.	May
UR_RESOURCES_DISPLAY	The player shall be shown the resource cost for each game element, such as construction, upgrades, maintenance, and event handling.	May
UR_COUNTER_LIMIT	The player shall be made aware when they have reached the limit for the number of each building type.	Shall
UR_TIMER_ALERT	The player should be alerted of the time when the timer is at the 1-minute mark.	Should
UR_GAME_END	An ending screen shall be displayed to the player when the timer is over.	Shall
UR_PERFORMANCE	The player shall be shown their student satisfaction score and a complementary message.	Shall
UR_GAME_REPLAY	The player shall be offered the option to replay the game.	Shall

SYSTEM REQUIREMENTS

The following table outlines the system requirements, which are divided into functional and non-functional. **Functional** requirements specify the **technical features** necessary to ensure UniSim operates smoothly, including performance standards, hardware and software compatibility and network requirements. **Non-functional** requirements describe **how the system performs its functions**, focusing on constraints and performance criteria such as speed, security, usability, scalability, reliability, and maintainability.

❖ FUNCTIONAL REQUIREMENTS (FRs)		
ID	Description	User Requirements
FR_UX	The buttons in the interface shall provide visual and audio feedback when clicked.	UR_UX
FR_UX_RESPONSIVENESS	The system shall provide instant feedback when something is pressed	UR_UX
FR_DIFFICULTY	The game shall implement a challenging version of easy mode	UR_DIFFICULTY
FR_MAP_MOVEMENT	The map will allow movement in all directions but within map limits.	UR_MAP_INTERACTION
FR_EVENT_IMPACT	The player shall be made aware of the impact of their reactions on events.	UR_EVENTS
FR_BUILDING_PLACEMENT	The system will allow the player to construct and upgrade buildings.	UR_BUILDINGS
FR_RESOURCES	The player shall start with some initial resources.	UR_RESOURCES
FR_TERRAIN_MODIFY	The player shall be able to modify certain terrain types (e.g., clearing	UR_MAP_INTERACTION

	grassland, removing trees) to optimise space.	
FR_TERRAIN_IMPACT	The terrain type shall have an increase/decrease/not affect student satisfaction.	UR_MAP_ELEMENTS
FR_BUILDING_MAINTENANCE	Student satisfaction buildings generate will reduce over time, maintenance will restore, building upgrades will enhance rate of increase.	UR_MAINTENANCE
FR_TERRAIN_MAINTENANCE	Terrain will degrade to other types of terrain without maintenance, requiring more resources to maintain later.	UR_MAINTENANCE

❖ NON-FUNCTIONAL REQUIREMENTS (NFRs)			
ID	Description	User Requirements	Fit Criteria
NFR_EVENT_IMPACT	The system will respond to player actions on events by initiating other events, manipulating resources.	UR_EVENTS	
NFR_MAP_GENERATION	The system shall generate the map within a few seconds.	UR_MAP	Within 2 seconds
NFR_UX_RESPONSIVENESS	The system shall provide instant feedback when something is pressed	UR_UX	Within 1ms
NFR_GAME_END_TIMER	The system must terminate the game as soon as the timer reaches 0.	UR_GAME_END	Within 1ms
NFR_SCALABILITY	The game should support future expansion.		
NFR_SECURITY	The system should encrypt		
NFR_RELIABILITY	The game should not crash	UR_PERFORMANCE	In case of crash, should display an error message

REFERENCES

Guides on information to include and format:

https://www.designrush.com/agency/software-development/trends/software-requirements-specification

https://asana.com/resources/software-requirement-document-template

Guide on distinction between functional/non-functional requirements:

https://www.altexsoft.com/blog/functional-and-non-functional-requirements-specification-and-types/

Requirements feedbacks:

- Needs to be less specific in places - could combine some requirements into one
- Make some requirements clearer - e.g. “maintenance” is a bit vague
- Game difficulty should be a functional requirement