Requirements

Team 8
OctaGame

Lisandro Nascimento
Thomas Nash
Letam Patrick-Bienwi
Nathan Peacock
Will Peel
Harrison Pegg

Introduction

The UniSim game purpose is to build a single-player game that allows the player to build their own university campus.

The requirements for the UniSim game were obtained using the product brief and negotiated through a detailed conversation with the stakeholder.

From the product brief, the team was given direction on specific features, primary objectives, must haves, win-condition, and duration of the game. Additionally, the team was encouraged to use appropriately-licensed 3rd-party libraries, tools and assets, instead of implementing from scratch.

During the session with the stakeholders, relevant questions were asked to understand the customer's vision for the game, his preferences regarding gameplay difficulty, game events, environment themes and performance specifications. The customer suggested a straightforward gameplay baseline, with a static map, basic 2D graphics, and deterministics events for the first iteration. Additionally, the customer stated his desire for a cross-platform compatibility and low hardware requirements for the game. These requirements were realistic and achievable, balancing the desired game experience with available resources and timelines.

This conversation helped to establish a clear roadmap for the UniSim project, ensuring that each requirement was aligned with the customer's vision.

User Requirements Table

ID	DESCRIPTION	PRIORITY
UR_GAMEPLAY	The game should be easy to understand and play.	HIGH
UR_GAME_COMPATIBILITY	The game should be compatible with various operating systems.	HIGH
UR_GAME_PAUSE	The game should start paused and allow the user to pause when needed.	MEDIUM
UR_EVENTS	There should be at least three events that take place during the game.	LOW
UR_DIFFICULTY_LEVELS	The game should have a single difficulty level for the first iteration.	MEDIUM
UR_WIN_CONDITION	No strict win condition. The game should show positive indicators.	LOW
UR_AUDIENCE	The target audience are those interested in or likely to attend university.	HIGH

UR_GRAPHICS	Clear and basic 2D graphics should be used.	HIGH
UR_MAP	There must be a map to build the campus on.	HIGH
UR_BUILD	Players should be able to place buildings on map permitted areas.	HIGH
UR_SLEEP_PLACE	There must be at least one place to sleep.	HIGH
UR_LEARN_PLACE	There must be at least one place to learn.	HIGH
UR_EAT_PLACE	There must be at least one place to eat.	HIGH
UR_RECREATIONAL_ACTIVITY	There must be at least one place for students to participate in fun activities.	HIGH
UR_TIME_TRACKER	The game should last up to 5 minutes.	HIGH
UR_BUILDING_COUNTER	There should be a counter for each placed building.	HIGH
UR_BACKGROUND_MUSIC	There should be a theme song played during the course of the game.	MEDIUM
UR_VISUAL_AND_SOUND_EFFECTS	There should be visual/sound effects to indicate interaction.	MEDIUM
UR_GAME_UPDATES	The game should support updates, without requiring major changes.	MEDIUM
UR_REMOVE_BUILDING	The user should be able to remove a building that they have selected	MEDIUM

Functional Requirements Table

ID	DESCRIPTION	USER REQUIREMENTS
FR_MAP	Provide a static map with set paths.	UR_MAP
FR_BUILDING_PLACING	Allow to place buildings only in green areas or where there is no building, paths, nature, or lakes.	UR_BUILD, UR_VISUAL_EFFECTS
FR_CLOCK	The clock stops, when time is paused.	UR_TIME_TRACKER
FR_COUNTER_INCREMENT	Every time the same building is selected the counter should increase the count for the building.	UR_BUILDING_COUNTER
FR_VERSION_CONTROL	GitHub should be used for version control, tracking and collaboration.	UR_GAME_UPDATES

Non-Functional Requirements Table

ID	DESCRIPTION	USER REQUIREMENTS	FIT CRITERIA
NFR_CROSS_PLATFORM_SUPPORT	The game should be built on a platform that is compatible with Windows, Mac, and Linux.	UR_GAME_COMPATIBILITY	The game was developed in libGDX, a cross_platform Java game development framework that works on Windows, Linux, MacOS.
NFR_GAME_INTERFACE	The game should have a fun, intuitive interface that is easy to understand.	UR_GAMEPLAY, UR_GRAPHICS	Any targeted user will be able to play the game, as it is very intuitive.
NFR_GRAPHIC_ASSETS	Assets used do not jeopardise performance on low-end hardware.	UR_GRAPHICS	The game is light and performs very well.
NFR_BUILDING_INTERACTIONS	Selected buildings will turn red to indicate that the building cannot be placed in a certain area.	UR_VISUAL_EFFECTS	All interactions are fully functional.
NFR_EVENTS_GENERATOR	Generate environment-specific events at specific times.	UR_EVENTS	Not implemented yet.
NFR_GAME_DESIGN	The design of the game should be able to accommodate future updates.	UR_GAME_UPDATES	The first iteration is simple and ready for potential updates.
NFR_GAME_END	When the game hits 5 minutes, the game is over.	UR_TIME_TRACKER	The game will always end as soon as it hits 5 minutes.
NFR_SATISFACTION	Display a score/ranking at the end of the game.	UR_WIN_CONDITION	Not implemented yet.