Software Design Document

for

InArt VR Project (Ver 2.0)

Version 2.0

Prepared by Joseph Chong, Jimmy Hernandez, Edwin Hernandez, Jaquan Jones, Alberto Landeros, Tony Lee, Jennelle Maximo, Eduardo Meza, Dean Nguyen, Anthony Viramontes

The Institute for Interactive Arts, Research, and Technology (*InArt*)

December 10, 2021

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
First Draft	11/19/2021	Initial Draft of Document	1.0
Second Draft	12/09/2021	Document refinements	2.0

1. Introduction

1.1 Purpose

The Software Design Document (SDD*) outlines the necessary information used to define the design and architecture of the InART* VR* project. This SDD document gives the development team guidance on the overall development goals of the project. This document is incrementally and iteratively being produced during the project development life cycle.

The intent of the InART VR project is to produce a digital storytelling experience that immerses an audience in an interactive game in order to experience various sensory stimuli that not only mimics physical reality but augments it in various ways.

1.2 Document Conventions

The typographical conventions used in this document are defined in this section.

Typographical Conventions	Style Usage
Regular text	Times New Roman font (12 point)
Bold Text	Title/Section headings
Numbered Lists	Ordered Lists
Bulleted List	Unordered Lists
Asterisk (*)	Definition is available in the glossary (Section 11)

1.3 Intended Audience and Reading Suggestions

This document is intended for:

- Developers - Developers can use this document to overview the expected outcome of this project.
- Advisor - Will be evaluating this document. Will be able to follow up with the development of this project to determine if we are successfully implementing system features.

1.4 System Overview

The InArt VR project is an interactive storytelling experience that is immersive and an engaging learning experience. The player is placed in a virtual world in which they can interact with

various objects in a controlled and uniquely defined space. Depending on the item the player interacts with, the item will send players into three different unique curated levels in which the player shall complete in order to progress through the game.

2. Design Considerations

2.1 User Assumptions and Dependencies

This software assumes the following:

- The user has access to VR hardware i.e Oculus Quest 2*
- The user has access to computer hardware meeting VR hardware minimum specifications
    - Assuming if they don't have access to an Oculus Quest 2
- The user has access to a stable internet connection.

2.2 Developer Constraints

- VR hardware required for gameplay
- Computer hardware requirements to run Unity*
- Software development languages limited to C# and alternatively C++
- 3D assets in the Unity Store are limited

2.3 Project Goals and Guidelines

- InArt VR software will produce specific gestures offering directions and objective reminders
- InArt VR software will produce gestures similar to gestures done in everyday life.
- InArt VR software will produce a gameplay experience to procreate cultural immersion with our level design
- InArt VR software will produce unique gameplay to emphasize cultural experience in individual storyline levels

2.4 Development Methods

Agile development approach:

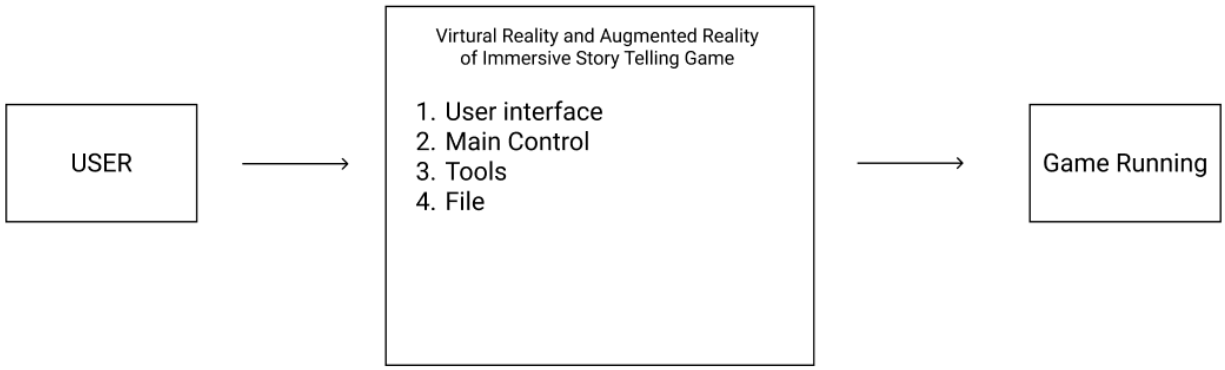
- Emphasis on group interaction and contribution to individual level gameplay storylines
- Gameplay as a whole broken down into smaller groups to prioritize specific level responsibility
- Small subsets of ideas within gameplay to support the immersive experience of the game as a whole
- Adaptability to changing requirements at any given point in the development progress
- Continuous progress delivered through storyline or resource development

3. Architectural Strategies

- **3D environment**
    - The software we are utilizing is the Unity game engine
    - It provides a sophisticated 3D environment that will provide a stable and realistic product
    - The user shall be able to interact with objects and the environment around them.
    - We will be using assets from the Unity store to save us time, so we can focus on high priority tasks.
    - The environments we shall include will add diversity to our levels.
        - Cooking Level
        - Farming Level
        - Festival Level
- **Gestures**
    - We will need to create an accurate response of gestures into each level.
    - Gestures are the focal point of our game, so each real life gesture should give feedback to the user to give the immersive feeling.
- **Hardware**
    - We are utilizing Oculus Quest 2 to test the program.
    - Not every developer will have access to a VR to test the program.
    - Any VR system will be able to run our program.
- **Unity Teams**
    - We will be using Unity Student Plan so we are all working on the same project file.
    - This will allow us to have the same settings and imports needed to share the same file.
    - All assets will be available, so we don’t need to have each member import an asset.
- The system will be reusable with any 3D environment.

4. System Architecture

Figure 4-1. Level 0 DFD



DFD* diagrams have been prepared to help illustrate the system specifications. The first level of the modules is the user interface, and it will accept every input from the user. The user interface is bounded between the user and their input, so they can call any module with their input. We broke down each major component into modules, and each module will represent a specific feature.

4.1 The User Interface Module

- Refer to DFD 0 or DFD 1. A detailed description is in Section 6.

4.2 The Main Control Module

- Refer to DFD 0 or DFD 1. A detailed description is in Section 6.

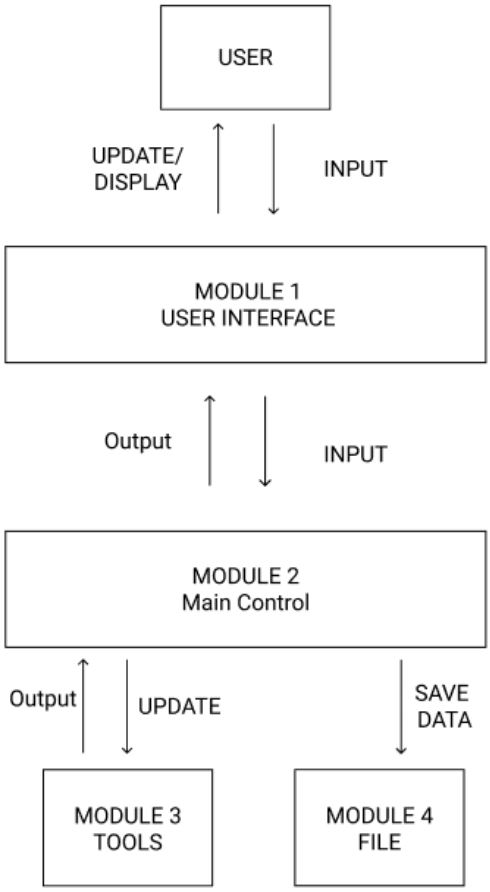
4.3 The Tools Module

- Refer to DFD 0 or DFD 1. A detailed description is in Section 6.

4.4 The File Module

- Refer to DFD 0 or DFD 1. A detailed description is in Section 6.

Figure 4-2. Level 1 DFD



5. Development Policies and Tactics

5.1 Choice of which specific development environment and tools

- IDE*: Visual Studio 2019
- Compiler: C# compiler
- Unity Plugins: XR Plugin
- VR Headsets: Oculus Quest 2

5.2 Plans for ensuring requirements traceability

- Requirements for documentation, plans, and meetings are recorded. Documentation is all written in other documents including Software Design Document. Requirements are discussed and reviewed in weekly meetings and also written in notes stored in Jira. Meetings that suggest new requirements are in our weekly meeting with the full group and subgroups. Most of these meetings are recorded on Zoom.

5.3 Plans for testing the software

5.3.1 Engineering trade-offs

- Extensive testing might be difficult as we are unsure mouse and keyboard controls will be compatible with VR controls
- Extensive testing will be applicable for those who have VR equipment
    - InArt does not have enough VR equipment for everyone in the team

5.3.2 Coding guidelines and conventions

- Coding Guidelines
    - Code must be as uniform indentations for readability
    - Frequent comments of scripts functions to describe purpose
- Conventions
    - Comments will be located in scripts to give an accurate representation of what the purpose of the script is
    - Proper naming conventions for variables so code is clear and readable

5.3.3 The protocol of one or more subsystems, modules, or subroutines

- All supporting packages, modules, and subsystems must be organized as dependencies

5.3.4 The choice of a particular algorithm or programming idiom (or design pattern) to implement portions of the systems functionality

- No set of particular algorithm or programming idiom to be implemented

5.3.5 Plans for maintaining the software

- Most textures and assets will be part of the aesthetic look and will usually not require frequent updating
- There will be no third party modifications that will break our software that will be our fault.
    - XR plugin is the only thing we might have to be careful because that is maintained by Unity and not by our developers
- Most scripts will be relatively simple to create and most likely not depreciate

5.3.6 Interfaces for end-users, software, hardware, and communications

- Software Interfaces
    - Unity Editor
        - Help with the visual 3D visualization
        - Used to create objects barriers and specifics objects to implement certain features we may or may not want players to experience
        - Import assets
        - Create a camera for the user to show a more immersive experience of gameplay
    - Visual Studio 2019
        - Used to write scripts for objects. Scripts will be in C#
        - The scripts are to help a object to do something
- Hardware Interfaces
    - Oculus Quest 2
        - Used to be able to access into Unity and for extensive testing

5.3.7 Hierarchical organization of the source code into its physical components (files and directories)

- Most original source code will belong in the Assets folder where the Unity project is file is located
    - Most original code will belong in a folder called “Original Scripts”
        - Most imported scripts have their own hierarchy of files but most scripts folders are called “Scripts”. “Original Scripts” is to prevent confusion in searching for scripts.

- Assets source code will still belong in the Assets folder but in its own Assets folder where organization will be determined by who made the Asset
    - A simple example would be /”Name of Asset”/”Scripts”
    - More complex Assets will have deeper rooted directory path
- All script code will be .cs files or C# files alternatively can be done in C++ files

5.3.8 How to build and/or generate the systems deliverables

- Installing XR Plugin
    - Go to File in a new/existing project then Build Settings
    - Click on Android in the Platform tab then press on Add Open Scenes
    - Click on Player Settings
    - Click on Other Settings tab and switch the order of Vulkan and OpenGLES3 in the Graphics APIs tab
    - Change the Minimum API Level to “Android 6.0 ‘Marshmallow’ (API level 23)”
    - On the Left tab, click on XR Plugin Management and install by clicking the only button present
    - Once the initial download is finished click on the check box marked Oculus
- Importing Assets
    - Make sure you are logged in Unity
    - Click on the Window > Package > Manager located towards the top of the editor
    - In the Packages tab towards the top click on “Package: My Assets”
        - To add an Asset to your account you must go to the Unity Asset store and then click on the asset you want. Then click on “Add to my Asset”. This will add it to your account and make it importable in the Unity editor
    - Click on the Import button based on what asset you want to import
    - A new window will appear and ask what specifics you want from the asset. By default you can import all of it. Click on Import.

6. Detailed System Design

Most components described in the System Architecture section are detailed below. Each subsection of this section will refer to or contain a detailed description of a system software component.

6.1 User Interface

6.1.1 Responsibilities

The user interface module is what the user is presented with in terms of visuals and environment, and it works along with the Main module. It provides a simplistic interface that will allow the user to interact with the system.

6.1.2 Constraints

There will not be any options to change any visuals or controls within the User Interface. Other Games have options of allowing the user to change brightness or change the control scheme, but our group does not have the time to implement this.

6.1.3 Composition

The User Interface Module is the middle ground between the User and the Game. The User Interface will remain consistent throughout each level. Each level will have their unique User Interface.

6.1.4 Uses/Interactions

The user will be able to use the user interface to interact with the main menu and have a different interface for each level.

6.1.5 Resources

Unity will be used to create the user interface for each of the levels.

6.2 Main Control

6.2.1 Responsibilities

The Main Control Module will be the main function of the game, because it is the foundation we are going to build the other modules on. It will handle the execution of the project, will handle input and output for the user, and it will handle the saves/checkpoints for the user.

6.2.2 Constraints

There is not enough development on the project to determine the constraints we may face.

6.2.3 Composition

The Main Module will hold most of the functionality of the game, including the user interface, files, and tools. The main Control is the foundation of the project.

6.2.4 Uses/Interactions

The user will be able to run the game. Once the game is running they will be able to experience all the levels we have to offer. If the user is not in a cut scene, the user interface will allow the user to interact with objects. The user will be able to use the tools funcion to fix any settings we have to offer. If the user fails a level or wants to restart a level, the file system will allow them to restart the level.

6.2.5 Resources

We will be utilizing the Unity asset store to use any 3D models that apply to our theme and environment. We may also use assets from the Steam store because they have free 3D models we can use.

6.3 Tools

6.3.1 Responsibilities

There will be some third party tools that we will import and integrate during development. The software will bundle Unity Plugins* that implement game functionality or control.

6.3.2 Constraints

Tools will only enable specialized actions and functionality and will not interfere with other systems or modules.

6.3.3 Composition

Unity plugins may have co-requisite dependencies, and they are auto imported by the Package Manager.

6.3.4 Uses/Interactions

The developer will be able to select individual plugins on an as needed basis in each Unity workplace scene.

6.3.5 Resources

Unity plugins can be acquired through the Asset Store and are organized into the Package Manager.

6.4 Gesture

6.4.1 Responsibilities

Gestures will be used to allow the user to trigger a certain action depending on what gesture is used.

6.4.2 Constraints

Certain gestures cannot be performed due to certain VR headsets having limited gestures.

6.4.3 Composition

VR controllers will generate the control signals that will be interpreted by the game engine as user input, which in-turn will be used to identify the different gestures being performed and execute the desired user command within the game environment.

6.4.4 Uses/Interactions

Gestures are performed through the VR controllers or hand tracking. The vr controllers will be able to track the position of your fingers and output a certain gesture depending on how you position your fingers. Hand tracking uses the position of your hands to make gestures.

6.4.5 Resources

VR controller and VR headset.

6.5 File

6.5.1 Responsibilities

The file module will allow the user to save their progress data onto a file and allow them to load and resume their progress from the file.

6.5.2 Constraints

Saving and loading can only be done through the main menu.

6.5.3 Composition

The option to either save or load will be contained inside of the main menu.

6.5.4 Uses/Interactions

To save or load a file, the user can interact with the main menu which will have an option to either save or load.

6.5.5 Resources

Will require the game data and the location of where the file will be saved/loaded.

7. Detailed Lower level Component Design

7.1 Name of Class or File

7.1.1 Classification

The components and classification that will be used will be the Unity cross platform. There will be multiple classes, packages, files (etc) for this project due to many implementations for the multiple gameplays and stories told for the user to interact with. Each story individually will have its own set of packages and files, so there is no conflict with each set of groups of developers and prevent confusion within the project.
As of now further studies into the project are needed for additional information.

7.1.2 Processing Narrative (PSPEC*)

When the user first interacts with the project it will be presented in a classroom. In the classroom there will be three items displayed on a table for the player to interact/select to pick as a story to experience. Each story has its own narrative interaction and will be told differently from one another, but will have the similar interfaces like the other stories.
As of now further studies into the project are needed for additional information

7.1.3 Interface Description

The user interface (UI*) for the following project will need the player to have a virtual reality (VR) headset or a computer with compatible components with the program. If the user chooses to wear a VR headset they would need to adjust the headset size to fit comfortably along with two controllers that are compatible with the headset as well. If the user chooses to use a computer as an interface for the program/project they would need to check if the components are compatible with the computer software components itself as well.

7.1.4 Processing Detail

The processing details have not been constructed in programming yet, but it eventually will. There have only been ideas on what is expected, although there are a lot of ideas to implement. There are very limited details that can be implemented to the project and with the limitations of the hardware that will be used and software to be built/programmed. It limits the developers to construct and implement the ideas of what was expected.

7.1.4.A Design Class Hierarchy

The design of the classes will all have a parent class since there is a starting point for each of the stories. The main class will be set in the classroom class. This will be the start and the main parent class for the project. The child classes will be the three stories that are included. These three child classes will be parent classes for a more detailed design of the project for the user interface and programming visuals.

7.1.4.B Restrictions/Limitations

Depending on what platform the user chooses to use as an interface there will be certain limitations for each.

7.1.4.C Virtual Reality Headset

When using a VR headset the player is limited to using the remotes with the compatible headset. The headset itself should be compatible with the software. Certain gestures will have its limits to what is possible to do for the user interface. The usage of fingers is not included to be able to do small gestures with fingers such as pinch, writing, poking, or even very specific sign language signals using fingers.

7.1.4.D Computer

When using a computer the user must know if the components of the computer's software are compatible with the project's programming software as well. The user will not interact with a VR headset but will be using a monitor as a screen and a mouse and keyboard for remotes. The user will need to adjust the keyboard to their own comfortable settings as they wish to interact with the stories/gameplay.

7.1.4.E Performance Issues

There are no performance issues with the project yet due to the delay of constructing the project. There will be project test runs to observe if the project is running well and does not have any issues or minor issues that should be repairable. There are some performance issues expected while constructing the project but as for the completion of the project there should not be any performance issues.

7.1.4.F Design Constraints

Due to many of the designs for each of the levels we may not be able to implement all that is expected. Since there is a time limit to the project, some designs of the project might be delayed due to the main stories needing to be completed first before going into details of each of the stories' gameplay.

7.1.4.G Processing Detail For Each Operation

There are no final processing details for each operation for the stories that are expected for the project. Further explanation, each of the stories will need to be implemented to discuss the processing details for each story. With all the details of each story that has been discussed with each of the groups, the implementation of each operation and main stories will first need to be completed first. Once the story has been completed with all the main implementation that is expected, the groups should then work on more explicit details and implementation that is expected to be in each part of the project.

8. Storytelling/Gameplay Design

The sections below contain descriptions of the VR gameplay scenarios that will be available in the game, and outlines the experience design.

8.1 Street Food Vendor

8.1.1 Summary:
Street food vendor must earn enough cash to exit Vietnam and reunite with his family abroad.
8.1.2 Setting:
Set around the 1970's, the end of the Vietnam War.
8.1.3 Player experience:
Gameplay is to perform several cooking tasks such as preparing pho bowls for customers. The level is intended to evoke an emotional response by forming connections with people and culture, and to understand the background and results of emmigration.

8.2 Farm/Jail Scenario

8.2.1 Summary:
Mexican immigrant working on a farm for a harsh and unsympathetic boss.
8.2.2 Setting:
The level has two stages, the initial gameplay takes place in the farm field, and the player is eventually incarcerated.
8.2.3 Player experience:
The player begins the level behind a pickup truck, where they must select a farm tool from the truck bed to perform agricultural duties. The player can perform gestures to execute simple labor tasks. As the level progresses, the player is captured by immigration officers and placed in confinement with other individuals.

8.3 Jail Scenario

8.3.1 Summary:
The player character is taken to jain and questioned.
8.3.2 Setting:
The player will experience interactions with other NPCs in a jail cell.
8.3.3 Player experience:
The player will not understand English speaking and will instead hear an undistinguishable alien language; however, the player will be able to use hand signaling gestures to communicate with the officers. The player will understand his native language when spoken to by the other inmates.

8.4 Festival Scenario

8.4.1 Summary:

Raising awareness of refugee groups through a joyful celebration of the global multicultural community, featuring music, food, art, and fun.

8.4.2 Setting:

Outdoors, assortment of booths and activity areas.The time is night or sunset with many lights to promote a lively atmosphere.

8.4.3 Player experience:

The player is free to roam the environment and explore the festival offerings; however, all interactions are limited to the scope of the following storyline narrative interactions.

1. **Arrival:** The level starts with the player in the middle of an interaction with a ticket booth attendant near the entrance of the festival grounds. The attendant welcomes the player to the event and recommends that the player visit the three most popular attractions at the festival. (The three attractions will be the interactive elements of this festival level.)
2. **Minigame A, Flyer (Syrian kites):** The player guides a flying object while shooting (wind) to elevate kites that are slowly descending in elevation. The player gains points depending on how high and how many kites are flying after a certain amount of time.
3. **Minigame B, Maze (Train yard):** The maze is a top-down-view game that uses a combination of head tracking and button input to guide a character to safety out of a train yard without being caught by a stationary enemy.
4. **End:** The end of the festival is marked by a fireworks display.

9. User Interface

9.1 Overview of User Interface

The user will be greeted with a home menu upon starting up the game. There will be a “start game” button, an “options menu” button, and a “credits” button. Pressing the “start game” button will put the player in the first environment of the game, being the school. From here, there will be a pop up that will act as a tutorial for the player on how to move around and interact with objects. The player will be able to freely move around and there will be three distinct objects that upon interacting with, will transport the player to one of the three levels of the game. There will be a pop up when interacting with the aforementioned objects that will have a confirmation prompt and an indication of which level that object leads to. Each level will have different UI elements based on the plans and designs of each of the three sub-teams.

9.2 Screen Frameworks or Images

These can be mockups or actual screenshots of the various UI screens and popups.

9.3 User Interface Flow Model

A discussion of screen objects and actions associated with those objects. This should include a flow diagram of the navigation between different pages.

10. Requirements Validation and Verification

10.1 External Requirements

External Interface Requirements	**Component Modules/UI Elements That Satisfy Requirements**
School level - Upon accessing the VR experience, the user interface will display a school classroom: - Preview menu system on chalkboard - Be able to change settings - Include objects to transfer player to the other three game levels	Start menu UI element: - Has button to start game - Options tab Object components: - teleports user to the other three levels upon interacting with one of the three objects
Kitchen level - bowl with chopsticks as artifact: - Starts user off in kitchen - Access to menu at any time - Checklist to view objectives	- Menu UI element - Checklist UI element
Farm level - scythe or family photo as artifact: - Access to menu at any time - Checklist to view objectives - HUD for boss - Transfer player to jail upon completion of objectives - Gestures to NPCs - Interrogation room environment after jail sequence	- Menu UI element - Checklist UI element - HUD element for boss - Gesture components
Festival Level - poster of festival as artifact: - Access to menu at any time - Starts user off at entrance to festival - HUD for different mini games - Checklist to view objectives	- Menu UI element - Checklist UI element - HUD elements corresponding to each minigame - Checklist UI element
Hardware Interfaces:	- VR headsets supported
Software Interfaces:	- Unity Game engine required

	- APIs used - Computer with Windows or MacOS
Communications Interfaces	Facebook account required

10.2 Functional Requirements

Functional Requirements	**Component Modules/UI Elements That Satisfy Requirements**
System shall run smoothly on hardware used	Component modules and UI elements will be light enough for hardware to handle
System should have two modes of in-game locomotion	Component for each method of locomotion: - Joystick - Real life walking
System shall be free of game-breaking errors/bugs	Component modules and UI elements will work together with no issues
System shall have sound effects and music	Sound component
System shall have three levels	Module for each level
System shall have options menu	Options UI element
System shall allow player to interact with objects in environment	Object module and components to allow interacting
System shall allow user to adjust volume	Volume component
System shall allow user to store items	Inventory component

11. Glossary

- **DFD:**
  - Data flow diagram
- **IDE:**
  - Integrated development environment
- **InArt:**
  - Our senior design project sponsor, the Institute for Interactive Arts, Research, and Technology (InArt) is an interdisciplinary institute at CalState LA that advances artistic research, digital narrative, and game design.
- **Oculus Quest 2:**
  - Headset and controller bundle that enables a player to interact with a VR game.
- **PSPEC:**
  - Process specification
- **SDD:**
  - Software design document
- **SRS:**
  - System requirements specification
- **UI:**
  - User interface, graphical user interface
- **Unity:**
  - The cross-platform game engine by Unity Technologies that we used to develop our VR experience.
- **Unity plugin:**
  - First or third party software add-in that implements additional functionality to a Unity development project.
- **VR:**
  - Virtual reality

12. References

Information about refugees

- Global refugee crisis | Doctors Without Borders
- The Most Urgent Refugee Crises Around the World | World Vision Canada
- Refugee crises around the world - InfoMigrants

Flyer (Syrian kites) background info

- https://tinyhand.net/making-kites-the-way-syrian-refugees-do/
- https://youtu.be/uysmSJ8xx9E

Maze (Train yard) background info

- Migrant caravan: What is it and why does it matter? - BBC News