**ICEBERG PROJECT**



**제출일: 2020-11-04**

**IT융합대학 컴퓨터공학과**

**팀명: 지구방위본부**

**팀원: 김민석 201635924**

**오윤택 201636006 (팀 리더)**

**최윤호 201636084**

기간: 2020년 8월31일 ~ 2020년 11월 10일 (11주)

1. **서론**
   1. 배경
   2. 간략 게임 정보
      1. 메인게임 정보
      2. 미니게임 정보
   3. 차별성
   4. 프로젝트 일정
2. **관련 지식 및 적용 기술**
   1. 관련 지식 및 적용 기술
   2. 적용 기술 (빈 칸)
3. **시스템 구축**
   1. 개발 환경
   2. 시스템 구성 (민석 파트)
4. **사용자 매뉴얼**
   1. 메인게임 매뉴얼
      1. 게임 오브젝트 및 실행화면
      2. 게임 진행 설명
   2. 미니게임3종 매뉴얼
      1. 게임 오브젝트 및 실행화면
      2. 게임 진행 설명
5. **결론 및 개선방안**
   1. 문제점 및 개선방안
   2. 결론

**1. 서 론**

**1.1 배 경**

현재 우리는 급격한 환경 변화 속에서 살고 있습니다. 올여름 비정상적인 52일의 역대 최장기 장마, 해를 거듭할수록 강렬해지는 폭염과 옅어져 가는 사계절의 모습은 단순히 시간의 흐름에 맞는 행보는 분명 아닐 겁니다. 또한 최근 코로나 19의 대유행으로 비대면 소비가 증가하였습니다. 이에 택배와 음식 배달이 전년 대비 약 30% 정도 상승하였습니다. 비대면 소비의 증가에 따라 일회용품 쓰레기 배출량 또한 상승하였습니다. 이제는 환경오염을 넘어서 환경위험이라고 표현할 정도로 지구의 환경오염 시계는 24시에 다가가고 있습니다. 이에 저희 팀은 환경의 현 실태에 심각성을 느끼게 되었습니다.

반면 코로나 19로 인한 언택트 시대에 맞게 사람들의 관심은 자연스럽게 디지털 컨텐츠에 집중되었습니다. 특히 우리나라 같은 경우는 스마트폰 보급률이 95%에 육박하고, 접근성이 매우 좋기 때문에 모바일 컨텐츠 이용자들의 모바일게임에 대한 수요와 관심이 상당히 많이 증가하였습니다.

이러한 이유로 저희 팀은 환경오염의 심각성을 모바일게임이라는 플랫폼을 이용하여 많은 사람에게 쉽고 넓게 접근하고, 환경에 대한 인식을 개선할 수 있는 게임을 만들고자 했습니다.

**1.2 간략 게임 정보**

**1.2.1 메인게임 정보**

게임은 크게 메인게임과 미니게임 파트로 나뉘어 있습니다. 메인게임의 장르는 시뮬레이션 장르를 표방하고 있습니다. 시뮬레이션 장르를 표방한 만큼 게임 속에 계절과 날짜 등 시간의 흐름이 존재합니다. 자칫 지루해질 수 있는 시뮬레이션 게임의 특성상 빠른 템포의 플레이를 위하여 플레이어는 임의로 시간의 흐름을 배속으로 조절할 수 있습니다.

메인게임에는 플레이어가 조종할 수 있는 캐릭터가 마련되어 있고 이 캐릭터는 기분, 허기, 피로도 등의 상태에 영향을 받습니다. 플레이어는 메인게임에서 캐릭터를 터치로 조작하여 플레이어의 집을 돌아다닐 수 있습니다. 플레이어는 집안의 가구와 같은 오브젝트와 상호작용을 할 수 있습니다. 현관문과 상호작용하여 출근하거나 마트를 가는 등의 행동을 할 수 있고 외출에서 한 행동은 모두 캐릭터 상태와 환경변수에 영향을 줍니다. 부엌, 침대, 샤워부스 또한 캐릭터와 상호작용하여 캐릭터의 상태 및 환경변수를 조절할 수 있습니다. 환경변수는 플레이어가 외출 혹은 다른 오브젝트들과 상호작용을 할 때 친환경적인 선택을 했는지, 반환경적인 선택을 했는지 계산하는 변수입니다. 이렇게 생성된 환경변수는 전날과 대비하여 어떤 선택이 우위 했는지 알려주고 이는 미니게임의 레벨디자인에 영향을 주게 됩니다.

**1.2.2 미니게임 정보**

미니게임은 메인게임 장면에서 플레이어가 TV 게임기와 상호작용할 경우 진입할 수 있습니다. 미니게임은 3종류의 게임으로 준비되어 있습니다. 장애물 피하기 게임, 아케이드 게임, TPS 게임으로 구성되어 있습니다. 각각의 게임은 플레이어가 메인게임에서 선택한 변수들로 난이도와 필드가 변화합니다. 친환경적인 선택이 많았다면 플레이어는 게임별로 고득점을 할 수 있는 기회가 많아질 것입니다. 하지만 반대로 플레이어가 메인게임에서 반환경적인 선택을 했다면 플레이어는 고득점을 할 수 있는 기회가 줄어들 것이고 전체적인 난이도가 상승할 것입니다. 미니게임이 종료되면 미니게임에서 획득한 점수가 나오고 메인게임에서 다른 플레이어들과 점수를 경쟁할 수 있는 랭킹을 확인할 수 있습니다.

**1.3 차별성**

Iceberg 게임은 크게 세 가지 특징으로 타 게임과 차별성을 두고 있습니다.

먼저, ‘월드 퀘스트와 간접적 상호작용’ 입니다. Iceberg를 플레이하는 모든 플레이어들은 하나의 공통적인 퀘스트를 가지고 있고, 모든 플레이어들은 이 퀘스트를 공유합니다. 메인 게임에서 선택한 각 플레이어들의 선택은 서버에서 취합되어 전날 대비 환경변화의 지표를 구성합니다. 플레이어들은 각자 따로 게임을 하고 있다고 생각하지만, 모든 플레이어의 게임 플레이들은 간접적으로 서로와 서로를 향해 이어져 있는 구조를 띠고 있습니다.

두 번째는 ‘선택과 필드 변화’ 입니다. 이것은 위쪽에 설명된 월드 퀘스트와 밀접한 연관이 있습니다. 플레이어들은 시뮬레이션을 진행하면서 많은 환경과 관련된 선택지들을 만나게 됩니다. 모든 선택지는 각각의 딜레마가 있습니다. 딜레마에서 선택한 선택지는 친환경적인지 반환경적인지 분류되어 월드 퀘스트에 반영됩니다. 월드 퀘스트에 반영되어 산출된 환경변수는 미니게임 필드 변화에 영향을 미치게 됩니다.

세 번째는 ‘환경 의식 고취’ 입니다. 저희 프로젝트의 궁극적 목표와 같습니다. 환경은 단순히 개인의 노력으로는 회복되거나 변화하지 않습니다. 하지만 개인의 노력이 모여 모두의 노력이 된다면 그 결과는 다를 것입니다. 위에 나열된 월드 퀘스트와 간접적 상호작용, 선택과 필드 변화를 통하여 플레이어는 자신의 선택이 게임플레이를 하는 데 어떤 영향을 미치는지, 어떤 결과를 초래하는지 시간이 지나면서 학습할 수 있을 것입니다.

**1.3 프로젝트 일정**

****

**2. 관련 지식 및 적용 기술**

**2.1 관련 지식**

**- Unity 3D**

Unity 3D는 비디오 게임의 개발 환경을 제공하는 게임 엔진이자, 3D 애니메이션과 건축 시각화, 가상현실 등 인터랙티브 컨텐츠 제작을 위한 통합 저작 도구입니다. 또한 멀티플랫폼을 지원하여 윈도우, 맥OS, IOS, 안드로이드, 플레이스테이션, 엑스박스, 닌텐도, 웹 브라우저 등 27개의 플랫폼에서 사용 가능한 컨텐츠를 만들 수 있습니다.

**- C#**

C#은 마이크로소프트에서 개발한 객체지향 프로그래밍 언어로 닷넷 프레임워크의 한 부분으로 만들어졌습니다. C++ 와 자바의 문법과 비슷한 문법을 가지고 있습니다. 또한 C#은 수많은 라이브러리와 IDE를 이용하며 메모리 관리를 자동으로 해줍니다. 이는 프로그래머가 손쉽게 코딩할 수 있게 해주어 생산성을 높일 수 있습니다.

**- FireBase**

FireBase는 모바일 서버를 개발하기 위한 인증, 데이터베이스, 푸시 알람, 스토리지, API 등을 플랫폼 구축 시 자동으로 만들어줍니다. 인증 시스템을 지원하여 로그인과 관련된 일련의 과정을 지원합니다. Firebase는 NoSQL 기반의 3세대 데이터베이스입니다. 현재 많이 사용되는 오라클이나, mySQL 같은 관계형 데이터베이스보다, Document 형식의 빠르고 간편한 NoSQL 기반의 데이터베이스를 도입했습니다. 또한 Analytics를 제공하여 다수의 사용자가 모바일 앱을 어떻게 사용하는지 통계정보를 얻을 수 있습니다.

**2.2 적용 기술**

**- Component Based Design Pattern**

Iceberg Project는 게임으로 제작된 만큼 수많은 오브젝트를 가지고 있고, 또 오브젝트간의 상호작용이 굉장히 복잡하게 얽힐 수밖에 없습니다. 이러한 이유로 개발 과정에서 ‘무수한 커플링 현상, 무분별한 스파게티 코드 생성’이라는 문제점이 발생할 가능성이 컸습니다. 저희는 이러한 문제점을 인식하고는 개발 및 유지보수, 요구사항 변경 대응에 유연하게 대처할 수 있도록 Component based design pattern을 도입했습니다. Game Object 별로 독립적 기능이 하나의 Component가 되도록 설계하였고, 필요하면 싱글 톤 패턴을 활용하여 하위 객체를 컨트롤하는 등, 간결하고 직관적으로 코드를 짜도록 노력했습니다.

**장치이(가) 표시된 사진

자동 생성된 설명**

**- Object Pooling**

Iceberg의 미니게임인 Penguin Run과 Water Doggy는 게임 플레이 방식의 특성상 Game Object를 생성하거나 파괴하는 작업이 굉장히 빈번하게 발생합니다. 프로그래밍 내에서 Object의 생성 또는 파괴는 생성 시 메모리를 새로 할당하고 리소스를 로드, 파괴 이후에는 Garbage Collecting 작업을 수행하는 꽤 무거운 작업입니다. 실시간으로 상호작용하는 게임에서 Object의 파괴와 반복이 자주 일어나게 된다면 이는 곧 심각한 Frame Drop으로 이어지게 됩니다.

이러한 이유로 Object Pooling을 도입했습니다. 우선 Queue 형태의 Object Pool을 생성하여 게임에 필요한 Object들을 필요한 수 만큼 넣었습니다. 이후 게임이 Object를 생성해야 할 경우 Object Pool에서 Object를 de queue 하여 반환하고, 파괴할 Object는 Object Pool에 en queue 하는 방식으로 제작하였습니다.

Penguin Run과 Water Doggy 두 게임은 Object Pooling을 사용하기 전에는 50fps의 성능을 보여주었습니다. 하지만 Object Pooling을 적용한 후, Unity Profiler를 통해 분석한 결과 기존보다 20% 상승한 60fps의 성능을 보여주었습니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

**- FireBase Transaction**

게임 플레이 시 사용자들은 환경과 관련된 선택을 강요받게 됩니다. Iceberg는 이를 Firebase Cloudstore에 저장하게 됩니다. 하지만 이 과정에서 다수의 플레이어가 해당 Data를 동시 접근할 가능성이 굉장히 높으며, 이는 데이터 일관성을 무너뜨릴 수 있습니다.

이를 해결하기 위해서 사용자의 동시 접근 가능성이 높은 Data들은 Transaction을 사용하여 값을 읽거나 저장하는 방식을 사용했습니다. 이를 통해 사용자들의 환경 선택 Date들은 안전하게 유지될 것입니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

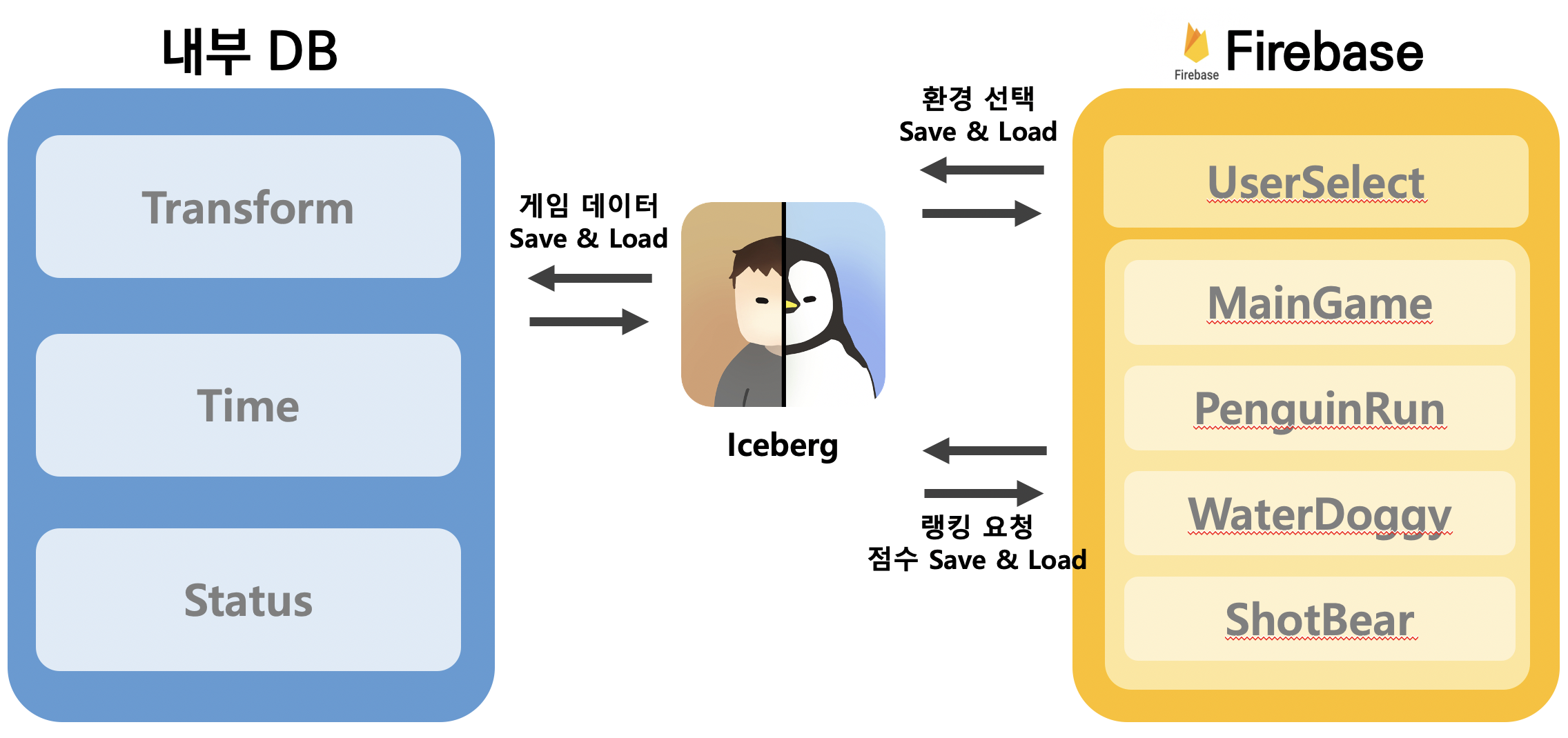
**3. 시스템 구축**

**3.1 개발 환경**

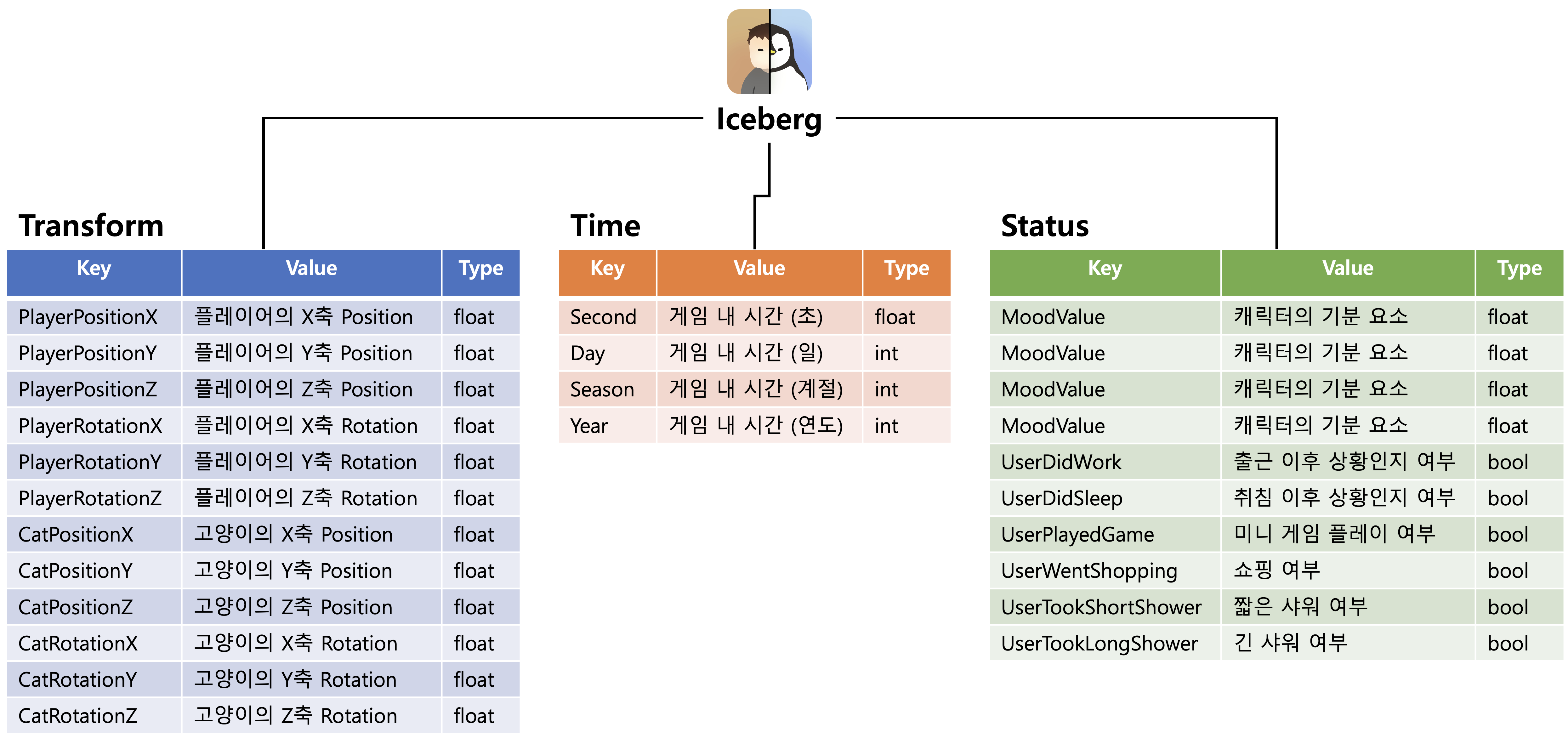
|  |  |
| --- | --- |
|  | **Device** |
| **H/W** | **CPU: Intel Core i7 -8700**  **RAM: 16GB** |
| **CPU: Intel Core i9**  **RAM: 32GB** |
| **CPU: Intel Core i9**  **RAM: 24GB** |
| **O/S** | **Windows 10** |
| **Mac OS Catalina** |
| **Development Tool** | **Unity 2020.1.4f1**  **Firebase**  **Adobe Illustrator**  **Adobe Photoshop** |
| **Editor** | **Visual Studio 2019, Visual Studio Code** |
| **실행환경** | **아이폰11 Pro, 아이폰12 Pro (IOS 14.1 이상)** |

**3.2 시스템 구성**

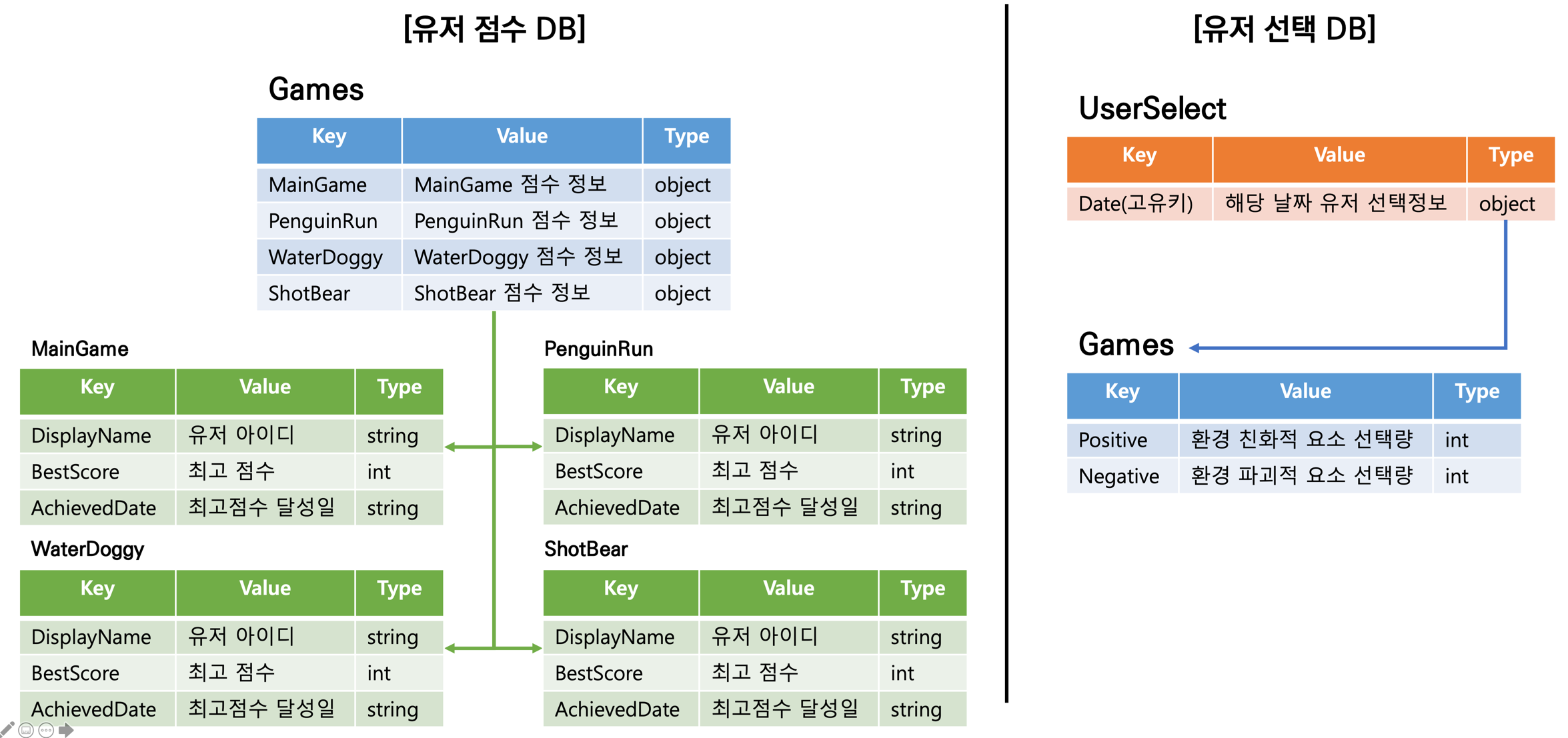
**- 데이터 흐름도**

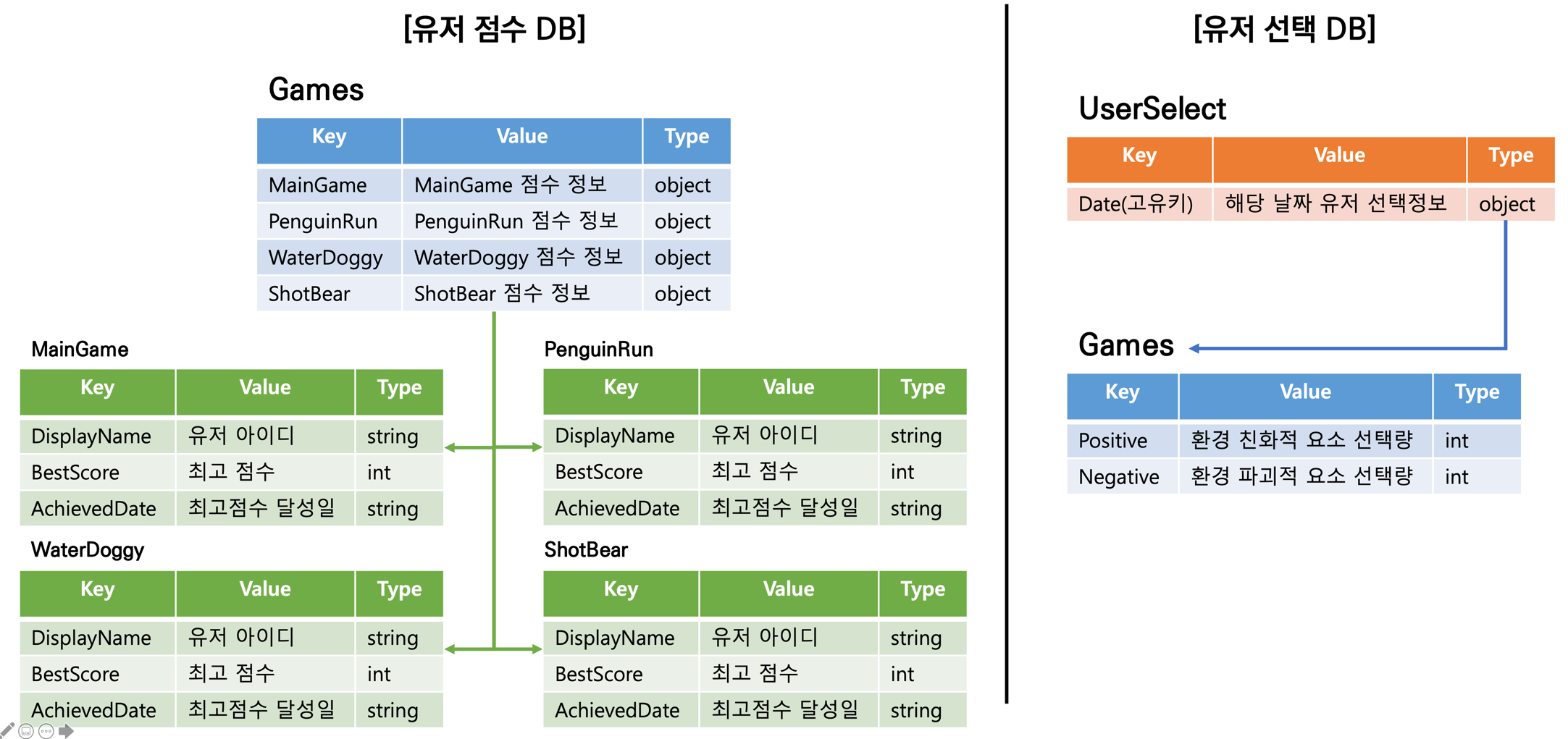
****

**- 내부 DB**

****

**- Firebase DB**

****

****

**4. 사용자 매뉴얼**

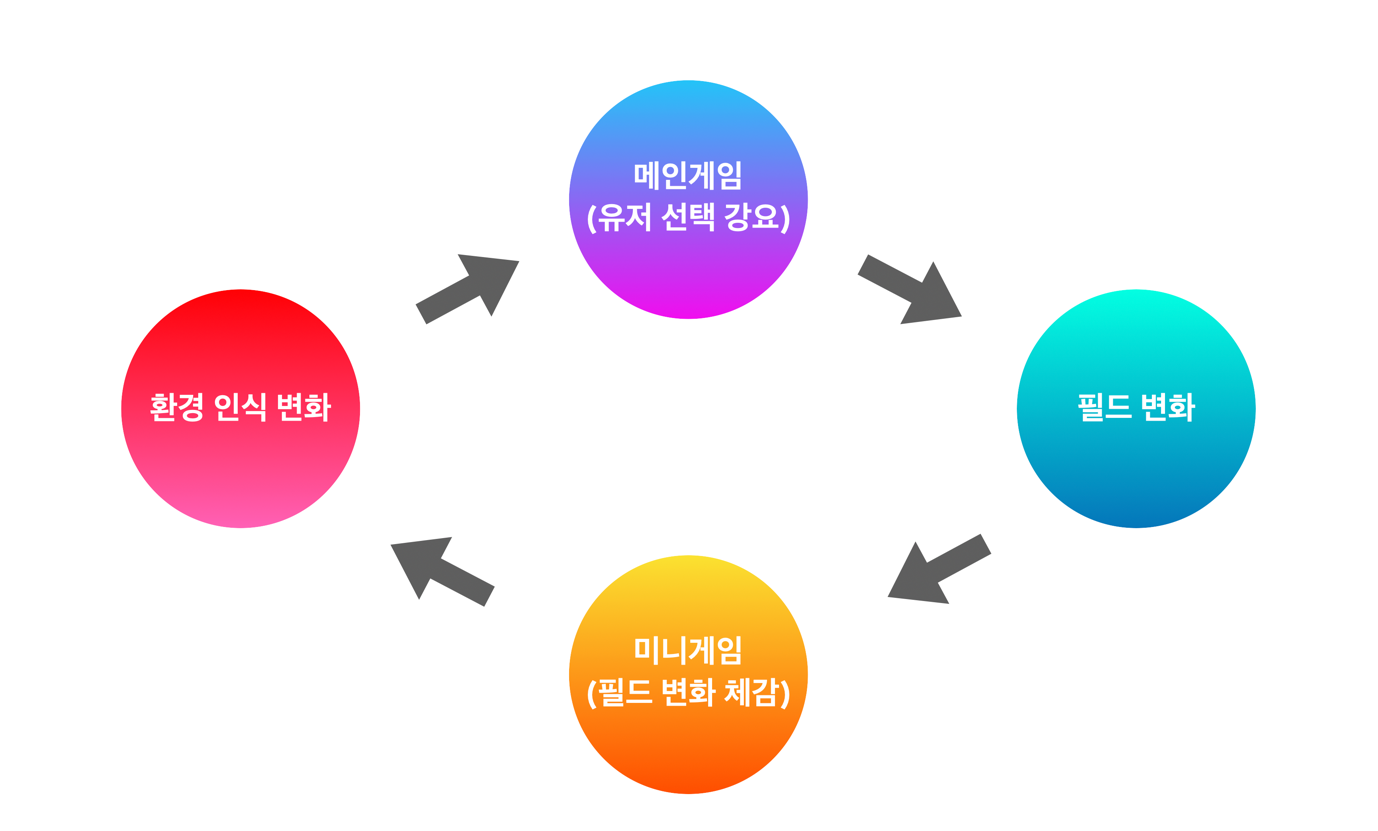
**4.1 메인게임 매뉴얼**

**4.1.1 게임 오브젝트 및 실행화면**

|  |  |  |
| --- | --- | --- |
| **오브젝트** | | |
|  | | |
| A 오브젝트 | 캐릭터 (플레이어) | 터치로 움직이며, 사물들과 상호작용을 하는 역할 |
| B 오브젝트 | 현관문 | 외부활동 통로.  상호작용 시 외부활동을 선택. |
| C 오브젝트 | 부엌 | 캐릭터의 허기를 채울 수 있는 역할. |
| D 오브젝트 | 워치 | 캐릭터의 전반적 상태, 환경 변동 추이, 게임 별 점수를 확인. |
| E 오브젝트 | TV | 미니게임 입장 통로. |
| F 오브젝트 | 침대 | 캐릭터의 피로도를 회복할 수 있는 역할. |
| G 오브젝트 | 샤워실 | 캐릭터의 기분과 피로도를 회복할 수 있는 역할. |

|  |  |
| --- | --- |
| **오브젝트** | |
|  |  |
| 로그인 화면 | 로딩 화면 |
|  |  |
| 샤워실 상호작용 | 침대 상호작용 |
|  |  |
| 현관문 상호작용 | 현관문 상호작용 시 외부활동 선택지 제시 |
|  |  |
| 마트 선택지 선택 | 쇼핑 중 환경 선택지 제시 |
|  |  |
| 부엌 상호작용 | 부엌 상호작용 시 요리 선택지 제시 |
|  |  |
| 요리 후 허기 회복 | 샤워실 상호작용 |
|  |  |
| 샤워실 상호작용 시 환경 선택지 제시 | 샤워 후 피로도 및 기분 회복 |
|  |  |
| TV 상호작용 | TV와 상호작용 시 미니게임 선택지 제시 |
|  |  |
| 좌측 하다 워치 클릭 시 캐릭터 상태확인 | 전날 대비 환경 변화량 확인 |
|  |  |
| 각 게임 별 점수 랭킹 확인 | 로그아웃 기능 |
|  |  |
| 침대 상호작용 | 취침 로딩 |
|  |  |
| 취침 시 다음 날 활동 가능 | 출근 선택 시 환경 선택지 제시 |
|  |  |
| 근무 로딩 | 점심시간 로딩 |
|  |  |
| 점심식사 관련 환경 선택지 제시 | |

**4.1.2 게임 진행 설명**

****

[그림] Iceberg Content Wheel

플레이어는 Iceberg를 플레이하며, 메인게임 → 필드 변화 → 미니게임 → 환경 인식 변화로 구성되는 컨텐츠 구성을 반복하여 경험하게 됩니다.

우선 인생 시뮬레이션을 기반으로 한 메인게임의 경우, 플레이어는 게임 속 캐릭터가 되어 살아가게 됩니다. 플레이어는 출근으로 하루를 시작하여 취침으로 하루를 마무리하게 됩니다. 이 과정에서 플레이어는 수많은 환경 선택지를 필수적으로 접하게 됩니다. 출근 방법(자가용 / 대중교통), 커피 테이크 아웃 방법(일회용 컵 / 텀블러), 쇼핑 방법(비닐봉지 / 에코백) 등이 그 예입니다. 저희는 플레이어들이 해당 선택지들을 의도적으로 회피할 수 없도록 필수적으로 선택하기 위한 다양한 장치들을 마련했습니다. 예를 들어 플레이어는 원활한 게임 플레이를 위해 캐릭터 상태 3요소(기분, 허기, 피로도)를 신경 써야 합니다. 상태 3요소가 부족하게 되면 플레이어는 게임 플레이에 불이익을 받게 되며, 해당 요소들을 회복하기 위해서는 환경 선택이 강요되는 방식입니다.

플레이어들이 선택한 선택지들은 모두 Firebase에서 취합하여 현재 게임 내의 환경 상태를 갱신하게 됩니다. Firebase는 전체 선택지 대비 환경친화적 선택 비율을 Iceberg에게 제공하며, Iceberg는 이를 통해 미니게임의 필드를 변화시킵니다.

플레이어는 게임 내 취침 전 캐릭터의 기분을 높이기 위해서 미니게임을 필수적으로 실행하여야 합니다. Penguin Run, Water Doggy, Shot Bear 하는 세 가지 미니게임 선택지가 존재하며, 플레이어의 취향에 따라 원하는 게임을 플레이하게 됩니다. 해당 게임은 환경 변수에 따른 각각의 필드 변화가 존재합니다.

|  |  |
| --- | --- |
| - Penguin Run | 장애물 및 크랙의 양이 증가. |
| - Water Doggy | 빙하 파괴율 증가 및 감점 요소 생성 비율 증가. |
| - ShotBear | 적의 체력, 쓰레기 아이템 데미지, 쓰레기 아이템 출현 빈도 증가, 체력 아이템의 회복 양, 플레이어의 체력, 눈덩이의 데미지 감소. |

다음은 유저의 선택에 따라 환경 오염이 심각해졌을 경우 각각의 미니게임의 필드가 변화하는 방식입니다.  
이와 같이 환경 변수에 따라서 미니게임의 필드가 플레이어에게 유리하게 또는 불리하게 변화합니다.

미니게임의 플레이가 끝나는 시점에서 플레이어는 메인게임에서의 선택과 미니게임 필드 변화 사이의 연관성을 깨닫게 될 것입니다. 메인게임에서 자신이 한 선택이 미니게임 내의 동물들에 어떠한 미칠 수 있는지 자연스럽게 알게 되는 것이죠. 이에 따라 플레이어는 조금이나마 환경에 대한 올바른 인식을 가지게 될 것입니다. 만약 지금 당장 느끼지 못했더라도 Iceberg를 지속적으로 이용한다면 언젠가는 환경 인식이 바뀌게 된다는 것은 분명합니다.

다시 플레이어는 메인게임으로 돌아가 위에 기술한 게임 진행방식을 반복하게 됩니다. 다른 유저와의 랭킹 경쟁, 날마다 변화하는 환경 등 지루함을 방지하기 위한 장치를 마련했기 때문에 유저는 오랫동안 Iceberg를 플레이 할 수 있습니다.

**4.2 미니게임(Penguin Run) 매뉴얼**

**4.2.1 게임 오브젝트 및 실행화면**

|  |  |  |
| --- | --- | --- |
| **오브젝트** | | |
|  | 펭귄 (플레이어) | 플레이어는 조작 버튼을 이용해 펭귄을 좌우로 조종. |
|  | 크랙 장애물 | 환경변수의 영향으로 출현빈도가 정해짐.  펭귄으로 크랙을 밟을 시 라이프가 감소. |
|  | 흰 상자 장애물 | 환경변수의 영향으로 출현빈도가 정해짐.  펭귄이 흰 상자와 충돌 시 라이프와 상관없이 바로 게임오버. |

|  |  |
| --- | --- |
| **실행화면** | |
|  |  |
| 미니게임(펭귄 런) 로비화면 | 튜토리얼 |
|  |  |
| 시작화면 | 바닥크랙 장애물과 충돌 |
|  |  |
| 장애물을 회피 | 장애물의 수 증가 |
|  |  |
| 게임오버 |

**4.2.2 게임 진행 설명**

0. 흰 상자 장애물의 출현 빈도, 바닥 크랙 장애물의 출현 빈도, 펭귄(플레이어)의 속도는 메인게임에서 받아온 환경변수에 의해 조절되어 초기화.

1. 로비화면에서 튜토리얼 선택 시 간략한 게임 방법 설명

2. 스타트 선택 시 바로 게임 시작.

3. 플레이어는 화면의 좌, 우 화살표 버튼으로 펭귄을 조작.

4. 플레이어는 펭귄을 조작하여 장애물을 피하면서 앞으로 진행.

5. 펭귄이 이동한 거리에 따라서 비례하게 점수를 획득.

6. 펭귄의 속도는 거리에 따라서 비례하게 증가.

7. 흰 상자 장애물에 부딪히면 바로 게임오버.

8. 환경이 극심하게 변화되면 크랙이라는 장애물이 생기는데 밟으면 라이프가 1개 닳고, 3개 라이프가 모두 닳게 되면 게임오버.

**4.3 미니게임(Water Doggy) 매뉴얼**

**4.3.1 게임 오브젝트 및 실행화면**

|  |  |  |
| --- | --- | --- |
| **오브젝트** | | |
|  | 물개 (플레이어) | 플레이어는 가상 조이스틱을 이용해 물개를 상하좌우로 조종. |
|  | 시계 아이템 | 시계 아이템 획득 시 플레이할 수 있는 시간이 연장. |
|  | 산소 아이템 | 산소 아이템 획득 시 물개의 산소량을 회복. |
|  | 물고기 아이템 | 물고기 아이템 획득 시 물개의 에너지 5점 회복.  점수 10점 회득. |
|  | 쓰레기 장애물 | 환경변수의 영향으로 출현빈도가 정해짐.  쓰레기 장애물에 충돌 시  에너지 5점 감소 |
|  | 해파리 장애물 | 환경변수의 영향으로 출현빈도가 정해짐.  해파리 장애물에 충돌 시  에너지 10점 감소 |

|  |  |
| --- | --- |
| **실행화면** | |
|  |  |
| 미니게임(워터 도기) 로비화면 | 튜토리얼1 |
|  |  |
| 튜토리얼2 | 튜토리얼3 |
|  |  |
| 시작화면1 | 시작화면2 |
|  |  |
| 물고기 아이템 습득 | 시계 아이템 습득 |
|  |  |
| 물 위 빙하에서 산소 충전 | 해파리 장애물과 충돌 |
|  |  |
| 일정확률로 사라져가는 빙하 | 게임오버 |

**4.3.2 게임 진행 설명**

0. 빙하 파괴 확률, 쓰레기 장애물 출현 빈도는 메인게임에서 받아온 환경변수에 의해 조절되어 초기화.

1. 로비화면에서 튜토리얼 선택 시 간략한 게임 방법 설명

2. 스타트 선택 시 바로 게임 시작.

3. 좌측 하단 가상 조이스틱으로 물개를 상/하/좌/우로 이동.

4. 우측 하단 대쉬 버튼으로 물개를 빠르게 가속 이동.

5. 매초 일정량의 에너지와 산소가 자동으로 감소.

6. 맵을 떠다니는 물고기 아이템을 획득해 점수와 에너지 증가.

7. 맵을 떠다니는 해파리 및 쓰레기 장애물과 충돌 시 점수 감소.

8. 산소가 부족할 경우 에너지가 급격히 감소.

9. 수면 위 빙하로 올라갈 경우 산소량 증가.

10. 일정 확률로 빙하가 파괴.

11. 물개 (플레이어)의 에너지가 0이 되면 게임 종료.

**4.4 미니게임(Shot Bear) 매뉴얼**

**4.4.1 게임 오브젝트 및 실행화면**

|  |  |  |
| --- | --- | --- |
| **오브젝트** | | |
|  | 북극곰 (플레이어) | 플레이어는 가상 조이스틱을 이용해 앞/뒤/좌/우 로 이동 가능.  공격 버튼 터치 시 눈덩이를 던져 공격 가능. |
|  | 인간 (적) | 맵을 무작위로 돌아다니며, 쓰레기 장애물을 투척.  플레이어가 던진 눈덩이와 충돌 시 체력이 감소.  체력이 0이 될 시 회복 아이템을 생성.  환경변수의 영향으로 속도와 개체 수가 정해짐. |
|  | 눈덩이 | 인간 (적)에게 데미지를 준다.  환경변수의 영향으로 데미지의 양이 정해짐. |
|  | 쓰레기 장애물 | 북극곰 (플레이어)에게 데미지를 준다.  환경변수의 영향으로 출현빈도와 데미지의 양이 정해짐. |
|  | 회복 아이템 | 북극곰 (플레이어)의 체력을 회복시킨다.  환경변수의 영향으로 회복량이 정해짐. |

|  |  |
| --- | --- |
| **실행화면** | |
|  |  |
| 미니게임(샷 베어) 로비화면 | 튜토리얼 |
|  |  |
| 시작화면 | 적이 도망치면서 쓰레기를 투척 |
|  |  |
| 눈덩이를 던져 공격 | 쓰레기와 충돌 시 체력 감소 |
|  |  |
| 적 처치 시 회복아이템 획득 | 체력이 0되어 쓰러짐 |
|  |  |
| 게임 오버 |

**4.4.2 게임 진행 설명**

0. 인간(적)의 체력, 쓰레기 아이템의 데미지 양, 쓰레기 아이템 출현 빈도, 체력 아이템의 회복 양, 플레이어의 체력, 눈덩이의 데미지 양은 메인게임에서 받아온 환경변수에 의해 조절되어 초기화.

1. 로비화면에서 튜토리얼 선택 시 간략한 게임 방법 설명

2. 스타트 선택 시 바로 게임 시작.

3. 좌측 하단 가상 조이스틱으로 북극곰을 앞/뒤/좌/우로 이동.

4. 우측 하단 어택 버튼으로 눈덩이를 투척.

5. 매초 일정량의 체력이 자동으로 감소.

6. 맵을 무작위로 돌아다니는 인간(적)을 눈덩이로 저격.

7. 인간(적)은 일정 시간마다 쓰레기 장애물을 투척.

8. 인간(적)이 생성한 쓰레기 장애물을 플레이어가 밟을 시 점수와 체력 감소.

9. 인간(적)에게 일정 횟수 이상 눈덩이를 맞출 시 인간(적) 쓰러짐.

10. 인간(적)은 쓰러지면서 회복 아이템 생성.

11. 플레이어는 회복 아이템을 밟을 시 점수와 체력 증가.

12. 플레이어의 체력이 0이 되면 게임 종료.

**5. 결론 및 개선방안**

**5.1 문제점 및 개선방안**

저희가 만든 Iceberg 게임은 현재 메인게임보다는 미니 게임 3종에 컨텐츠가 집중되어 개발되었습니다. 환경변수로 인한 미니게임에 필드 변화를 주기 위한 선택이었지만 메인게임의 컨텐츠 부재는 게임으로서 아쉬운 부분 중 하나입니다. 그렇기 때문에 현재 구현된 상호작용 오브젝트들을 확장하고 선택의 폭을 넓힌다면 현재보다 게임으로서 가치를 높일 수 있다고 생각합니다.

저희 게임은 현재 미니게임이 3종 마련되어 있습니다. 하지만 제한된 수의 컨텐츠는 반복 플레이 방식으로 인해 플레이어의 지루함을 유발할 가능성이 있습니다. 그 때문에 저희는 이러한 문제점을 시즌제 운영으로 하여금 새로운 컨텐츠들과 다양한 미니게임을 추가하는 등 지속해서 업데이트할 예정입니다.

**5.2 결론**

저희 팀 세 명 모두 게임개발의 경험이 거의 없었기 때문에 기술적 문제와 경험의 부재가 프로젝트를 진행하는 데 걸림돌이 되었습니다. 그래서 저희 팀은 지속해서 회의를 하고 서로 정보 공유를 통해서 문제를 해결하고 게임을 구현하였습니다.

10주라는 시간 동안 많은 컨텐츠들을 구현하고 적용하는 것은 넉넉한 시간이었다고 생각하지 않습니다. 하지만 게임을 완성해 나가면서 점점 더 욕심과 자신감이 생겼고, 프로젝트 발표가 끝난 이후에도 지속해서 게임 컨텐츠들은 발전시키고, 추가 개발할 예정입니다.