조장: 유양우(201636010)

조원: 김민지(201835625)

전재경(201835732)

팀 과제

**<다빈치코드게임시스템>**



* 서론

1. 제안서 작성
   1. 팀 소개 및 역할 분담

● 팀 소개

-조장: 유양우(201636010)

-조원: 김민지(201835625), 전재경(201835732)

● 역할 분담

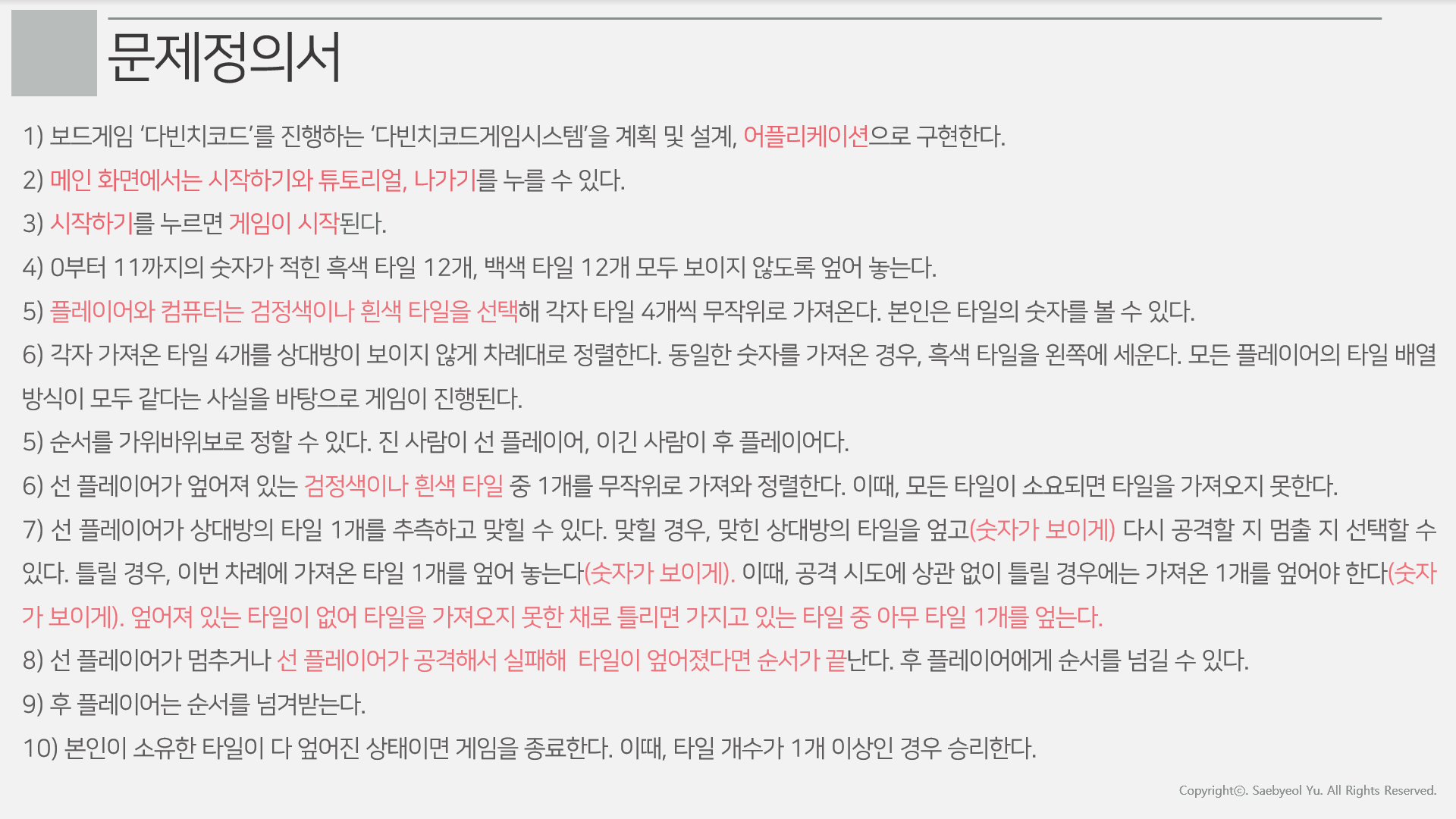
계획, 요구사항 명세서, 설계, 구현, 테스트 및 유지보수 단계로 진행한다.

-전재경: 계획 및 설계

-김민지: 설계 및 구현

-유양우: 총괄 및 구현

* 1. 문제정의서 및 알고리즘 구현방법
* 문제정의서



1) 보드게임 ‘다빈치코드’를 진행하는 ‘다빈치코드게임시스템’을 계획 및 설계, 어플리케이션으로 구현한다.

2) 메인화면에서는 시작하기와 튜토리얼 나가기를 누를 수 있다.

3) 시작하기를 누르면 게임이 시작된다.

4) 0부터 11까지의 숫자가 적힌 흑색 타일 12개, 백색 타일 12개 모두 보이지 않도록 엎어 놓는다.

5) 플레이어와 컴퓨터는 검정색이나 흰색 타일을 선택해 각자 타일 4개씩 무작위로 가져온다. 본인은 타일의 숫자를 볼 수 있다.

6) 각자 가져온 타일 4개를 상대방이 보이지 않게 차례대로 정렬한다. 동일한 숫자를 가져온 경우, 흑색 타일을 왼쪽에 세운다. 모든 플레이어의 타일 배열 방식이 모두 같다는 사실을 바탕으로 게임이 진행된다.

5) 순서를 가위바위보로 정할 수 있다. 진 사람이 선 플레이어, 이긴 사람이 후 플레이어다.

6) 선 플레이어가 엎어져 있는 검정색이나 흰색 타일 중 1개를 무작위로 가져와 정렬한다. 이때, 모든 타일이 소요되면 타일을 가져오지 못한다.

7) 선 플레이어가 상대방의 타일 1개를 추측하고 맞힐 수 있다. 맞힐 경우, 맞힌 상대방의 타일을 엎고(숫자가 보이게) 다시 공격할 지 멈출 지 선택할 수 있다. 틀릴 경우, 이번 차례에 가져온 타일 1개를 엎어 놓는다(숫자가 보이게). 이때, 공격 시도에 상관 없이 틀릴 경우에는 가져온 1개를 엎어야 한다(숫자가 보이게). 엎어져 있는 타일이 없어 타일을 가져오지 못한 채로 틀리면 가지고 있는 타일 중 아무 타일 1개를 엎는다.

8) 선 플레이어가 멈추거나 선 플레이어가 공격해서 실패해 타일이 엎어졌다면 순서가 끝난다. 후 플레이어에게 순서를 넘길 수 있다.

9) 후 플레이어는 순서를 넘겨받는다.

10) 본인이 소유한 타일이 다 엎어진 상태이면 게임을 종료한다. 이때, 타일 개수가 1개 이상인 경우 승리한다.

● 알고리즘 구현방법

1) 플레이어 : 타일 더미에서 각각 4개씩 고르기

2) 플레이어가 4개 타일 고른 후, 봇(상대방)이 고르기

3) 봇이 다 고르면 두 플레이어가 타일 정렬

4) 플레이어 턴으로 시작, 타일 더미 중 1개 선택

5) 플레이어 타일 정렬 후, 봇의 정렬된 타일 중 1개 선택

6) 플레이어가 선택한 봇의 타일에 적힌 숫자 추측, textfield에 숫자 입력하고 누르기

7) 정답 검사

- 맞으면 공격할 지, 멈출 지 선택

- 공격 : 4번으로 돌아감

- 멈추기 : 8번(봇 턴)으로 감

- 틀리면 : 8번(봇 턴)으로 감

8) 봇 턴 : 타일 더미 중 1개 선택 후 정렬

9) 플레이어의 정렬된 타일 중 1개 선택, 숫자 추측

10) 정답인지 검사

- 맞으면 공격할 지, 멈출 지 랜덤으로 선택

- 공격할 지 : 8번으로 감

- 멈출 지 : 4번(플레이어 턴)으로 돌아감

- 틀리면 : 4번(플레이어 턴)으로 돌아감

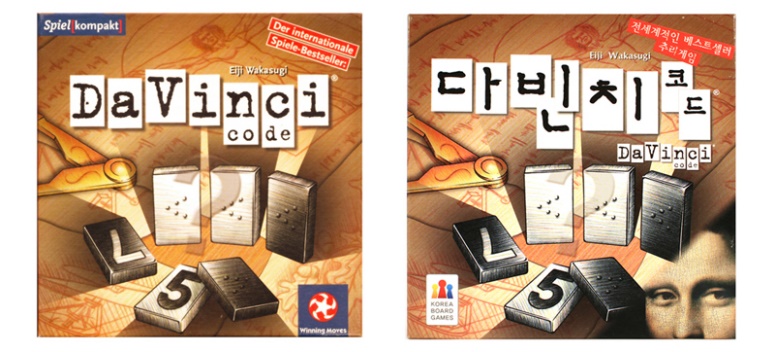
11). 계속 4~10번 반복 작업.

각 패널에 놓인 타일이 다 보여지거나, 13개의 타일이 다 들어오게 되면 종료

* 1. 기술 동향

2002년 일본의 갓켄에서 개발된 보드게임이며 본래 이름은 '아르고(Algo)'이다. 2003년에는 독일 위닝무브즈(Winning Moves Deutschland GmbH)에서 '코다'라는 이름으로 출시됐다. 위닝 무브즈사는 2004년에 게임의 이름을 'davinci code'로 바꿔 출시했 한국에서는 코리아보드게임즈에서 ‘다빈치 코드’로 정식 발매한 뒤, 이 이름으로 유명해졌다.

한국어판만의 다른 점은 바로 박스의 왼쪽 하단에 모나리자 이미지를 추가한 것이다. 모나리자는 다빈치의 가장 유명한 작품인 만큼, 국내 소비자들에게 흥미를 일으킬 수 있을 것으로 예상됐다.



논리적인 추리, 심리전, 운 등 3가지 재미있는 게임 요소를 적절하게 버무려지는 과정에서 약간의 운이 재미를 좌우해 어린이 ~ 20대 초 중반 정도 연령대에서 수요가 증가하는 추세이다.

기존에 출시된 앱들은 공격이 틀려 번호를 공개할 때(가지고 있는 타일의 숫자가 공개 될 때) 제대로 이게 엎어진 건지 아닌지 UI를 해놓았지만 우리 팀은 확실히 숫자가 공개됐는지 안됐는지 UI를 할 예정이다.

또 기존 다빈치코드앱들은 개발을 더 하지않고 멈춰 있다. 그로 인해 매칭도 잘 되지 않고 서버도 불안정하다. DB로 랭킹이랑 서버를 넣어 매칭하는 것은 시간 문제상 바로 되진 않겠지만 차차 추가 할 예정이다.

* 1. 프로젝트 일정

● ~6월 10일: 프로젝트 계획 및 과제 제안서 작성

● 6월 11일 ~ 6월 17일: 문제정의서를 바탕으로 유스케이스 다이어그램 및

유스케이스 기술서, 요구사항 명세서 작성

* 6월 17일 ~ 6월 24일: 설계
* 6월 24일 ~ 7월 10일: 프로젝트 구현 및 테스트, 유지보수
  1. 기대 효과

-보드게임을 계획하고 설계 하는 것으로 소프트웨어 공학의 중요성을 알 수 있다.

-요구사항을 파악, 분석하여 최종적으로 요구사항 명세서를 바탕으로 프로젝트를 진행하며 명세서의 중요성을 알 수 있다.

-객체지향 모델링을 통해 프로그램을 효율적으로 설계할 수 있다.

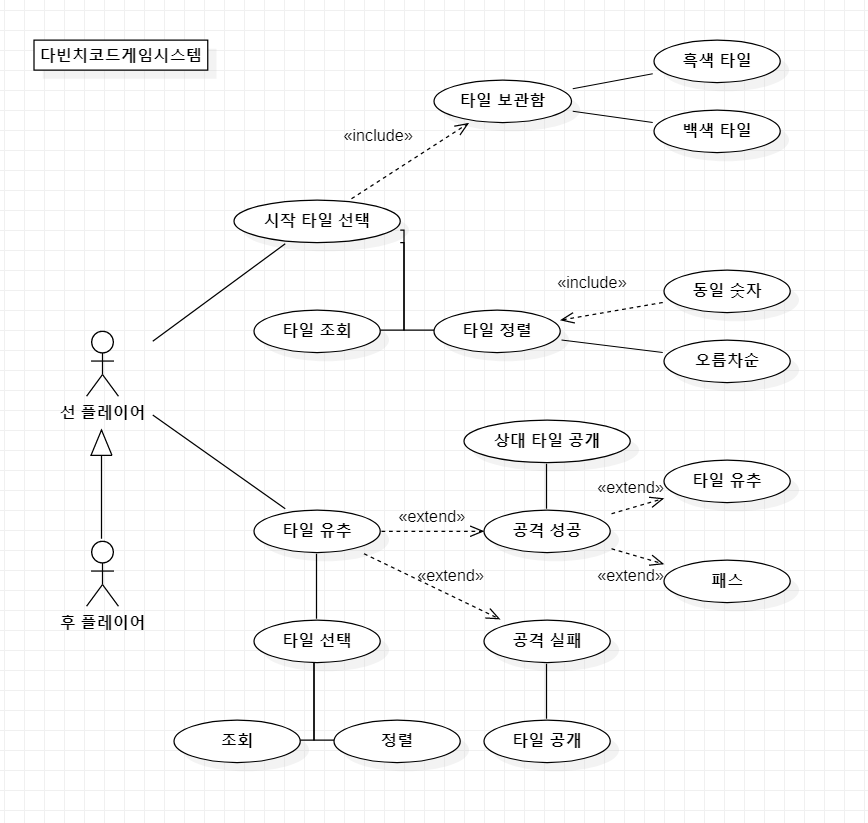
-팀 프로젝트를 통한 협업을 바탕으로 소프트웨어 개발을 하는데 있어 실제 소프트웨어 개발과정을 경험 할 수 있고, 개발 능력을 향상시킬 수 있음.

-구현부터 시작하는 개발보다 체계적인 소프트웨어개발을 할 수 있다.

* 본론

1. 계획 및 설계

* 1. 유스케이스 다이어그램



* 1. 유스케이스 기술서(UseCase Description)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case Description** | | | | | | | |
| Use case 명 | | 선택 | 작성일 | | 2020. 06. 13 | 페이지 | 1/3 |
| Actor명 | | 플레이어 | 작성자 | 다빈치코드  게임시스템 |
| * 목적   다빈치코드에서 선택 작업을 실행한다.   * 설명   플레이어가 타일, 순서 등을 선택한다.   * 선행 조건   플레이어가 게임을 시작한 상태여야 한다.   * 완료 조건   선택 작업 후에 조회와 정렬이 후행 되어야 한다.   * 유형   필수   * 관련 요구사항   플레이어는 Flow에 따라 선택을 진행한다.   * Flow | | | | | | | |
| 번호 | Actor | | | System | | | |
| M01 | 게임을 시작한다. | | | 게임 진행 화면을 요청해 출력한다. | | | |
| M02 | 시작 타일을 선택한다. | | | 시작 타일 선택을 요청한다. 선택() | | | |
| M03 | 순서를 정한다. [A01] | | | 게임 경로 추가화면을 출력한다. | | | |
| M04 | 타일을 선택한다. | | | 타일 선택을 요청한다. 선택() | | | |
| M05 | 상대 플레이어의 타일 뒷면을 본다. [B01],[B02],[B03] | | | 게임 경로 추가화면을 출력한다. | | | |
| A01 | 가위, 바위, 보를 선택한다. | | | 게임 경로 추가 요청을 한다. 게임경로 추가()  게임 진행 화면을 출력한다. | | | |
| B01 | 상대 플레이어 타일을 선택한다. | | | 게임 경로 추가 요청을 한다. 게임경로 추가()  게임 진행 화면을 출력한다. | | | |
| B02 | 타일의 숫자를 맞힌다. | | | 게임 경로 추가 요청을 한다. 게임경로 추가()  게임 진행 화면을 출력한다. | | | |
| B03 | 공격 시도 여부를 선택한다.  (타일 유추와 연관) | | | 게임 경로 추가 요청을 한다. 게임경로 추가()  게임 진행 화면을 출력한다. | | | |

* 선택

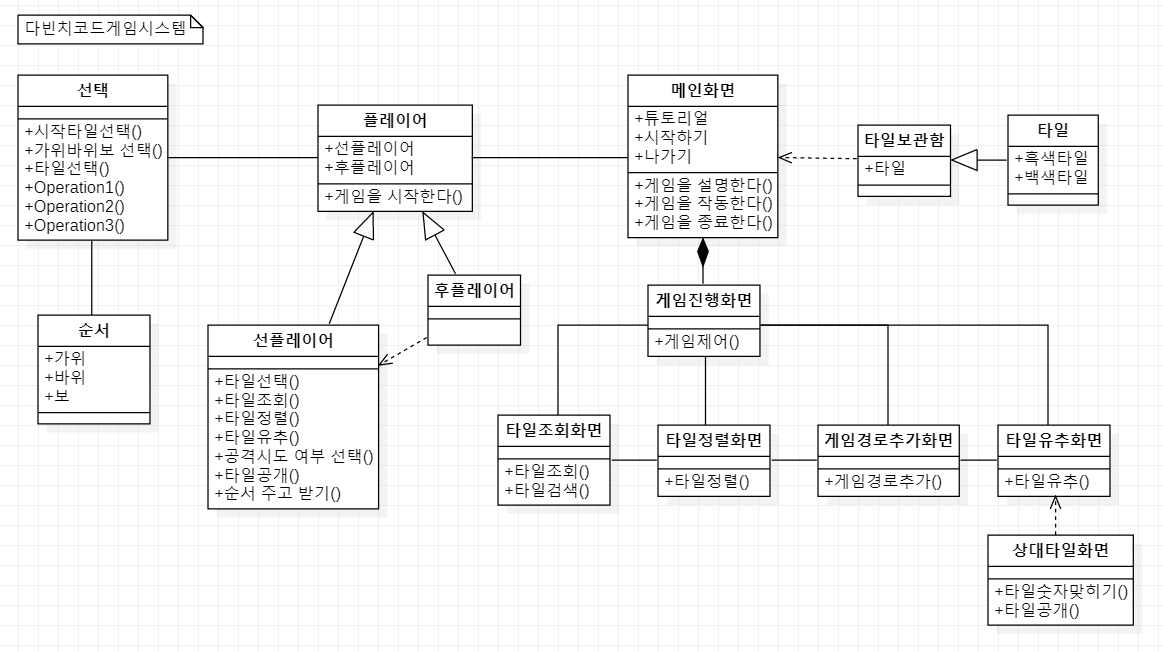
● 조회, 정렬

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case Description** | | | | | | | |
| Use case 명 | | 조회, 정렬 | 작성일 | | 2020. 06. 13 | 페이지 | 2/3 |
| Actor명 | | 플레이어 | 작성자 | 다빈치코드  게임시스템 |
| * 목적   다빈치코드의 타일을 조회하고 정렬한다.   * 설명   플레이어가 타일을 선택 후, 조회와 정렬 작업을 진행한다.   * 선행 조건   플레이어가 게임을 시작하고, 타일을 선택한 상태여야 한다.   * 완료 조건   타일을 조회하고 타일을 오름차순으로 정렬해야 한다.   * 유형   필수   * 관련 요구사항   플레이어는 Flow에 따라 조회, 정렬을 진행한다.   * Flow | | | | | | | |
| 번호 | Actor | | | System | | | |
| M01 | 게임 진행 화면에서 타일을 조회한다. | | | 타일 조회 화면을 요청하고 출력한다. | | | |
| M02 | 게임 진행 화면에서 타일을 정렬한다. [A01] | | | 타일 정렬 화면을 요청하고 출력한다. | | | |
| M03 | 타일 정렬을 요청한다. | | | 타일은 모두 같은 배열 방식 즉, 오름차순으로 정렬한다. | | | |
| M04 | 상대방의 타일 뒷면을 조회한다. [B01],[B02] | | | 상대 타일 화면을 요청하고 출력한다. | | | |
| A01 | 동일한 숫자이면서 다른 색 타일을 뽑는다. | | | 흑색 타일을 왼쪽에 세우는 방식으로 정렬한다. | | | |
| B01 | 공격 성공한 상대 타일을 조회한다. | | | 타일 조회 화면을 출력한다. | | | |
| B02 | 공격 실패 시 이번 차례에 가져온 타일 1개를 공개한다. [C01] | | | 타일 조회 화면을 출력한다. | | | |
| C01 | 타일을 가져오지 못했다면 본인의 정렬된 타일을 조회한다. | | | 타일 조회 화면을 출력한다. | | | |

● 유추

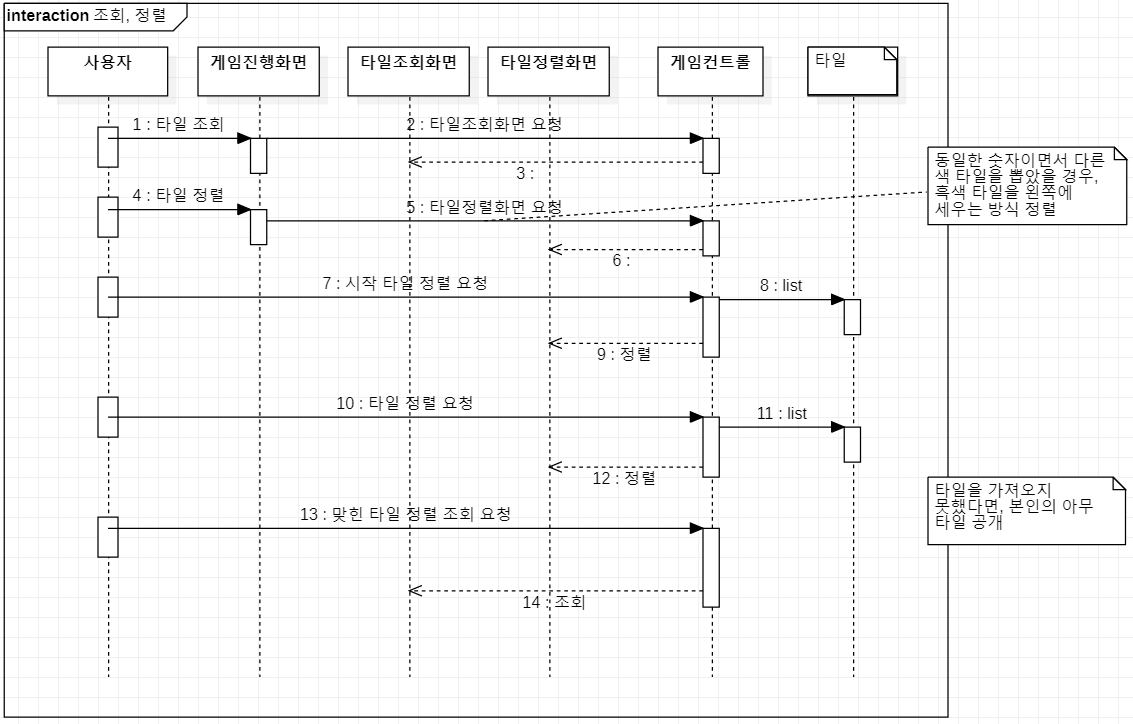
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Use Case Description** | | | | | | | |
| Use case 명 | | 유추 | 작성일 | | 2020. 06. 13 | 페이지 | 3/3 |
| Actor명 | | 플레이어 | 작성자 | 다빈치코드  게임시스템 |
| * 목적   다빈치코드의 타일을 유추한다.   * 설명   상대 플레이어의 타일에 적혀 있는 숫자를 맞힌다.   * 선행 조건   게임 시작, 조회, 정렬, 선택   * 완료 조건   두 플레이어 중 한 명의 타일 한 개라도 공개되어야 한다.   * 유형   필수   * 관련 요구사항   플레이어는 Flow에 따라 유추를 진행한다.   * Flow | | | | | | | |
| 번호 | Actor | | | System | | | |
| M01 | 상대 플레이어의 타일을 유추한다. | | | 상대 타일 화면을 요청하고 출력한다. | | | |
| M02 | 상대 플레이어의 타일을 유추한다. | | | 타일 유추 화면을 요청하고 출력한다. | | | |
| M03 | 타일의 예상 숫자를 입력한다. [T01] | | | 타일 유추 화면을 출력한다. | | | |
| T01 | 공격 성공 [S01] | | | 상대 타일을 공개한다. | | | |
| T02 | 공격 실패 | | | 본인 타일을 공개한다. | | | |
| S01 | 공격시도 여부를 결정한다. [D01], [D02] | | | 게임 진행 화면을 출력한다. | | | |
| D01 | 타일을 유추한다. | | | 타일 유추 화면을 요청하고 출력한다. | | | |
| D02 | 패스한다. | | | 상대 플레이어의 순서를 시작한다. | | | |
|  |  | | |  | | | |
|  |  | | |  | | | |

* 1. 클래스 다이어그램

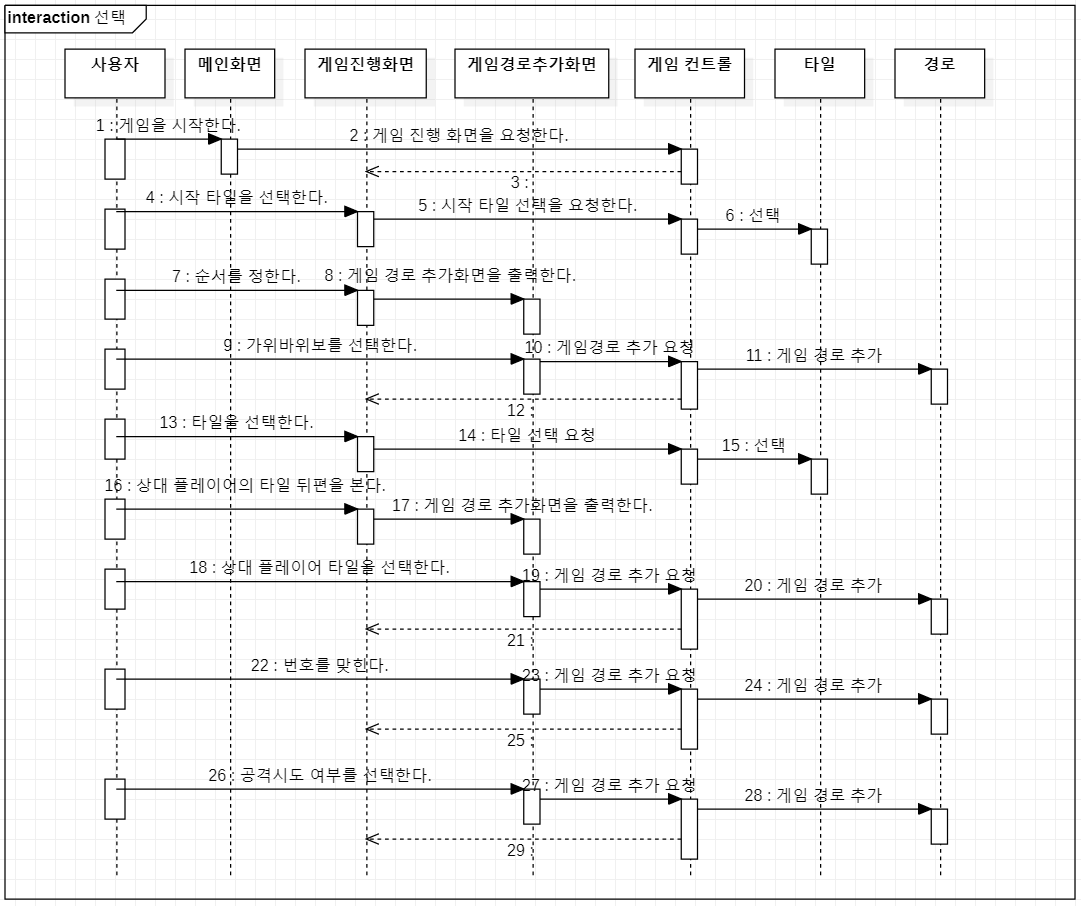


* 1. 시퀀스 다이어그램

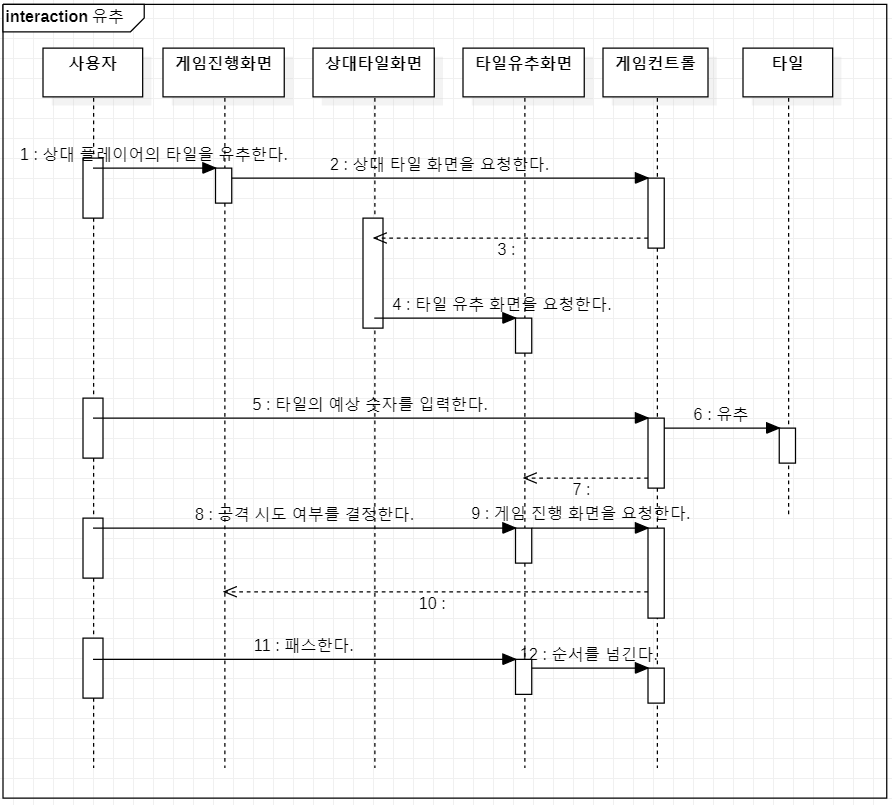
● 조회, 정렬



● 선택



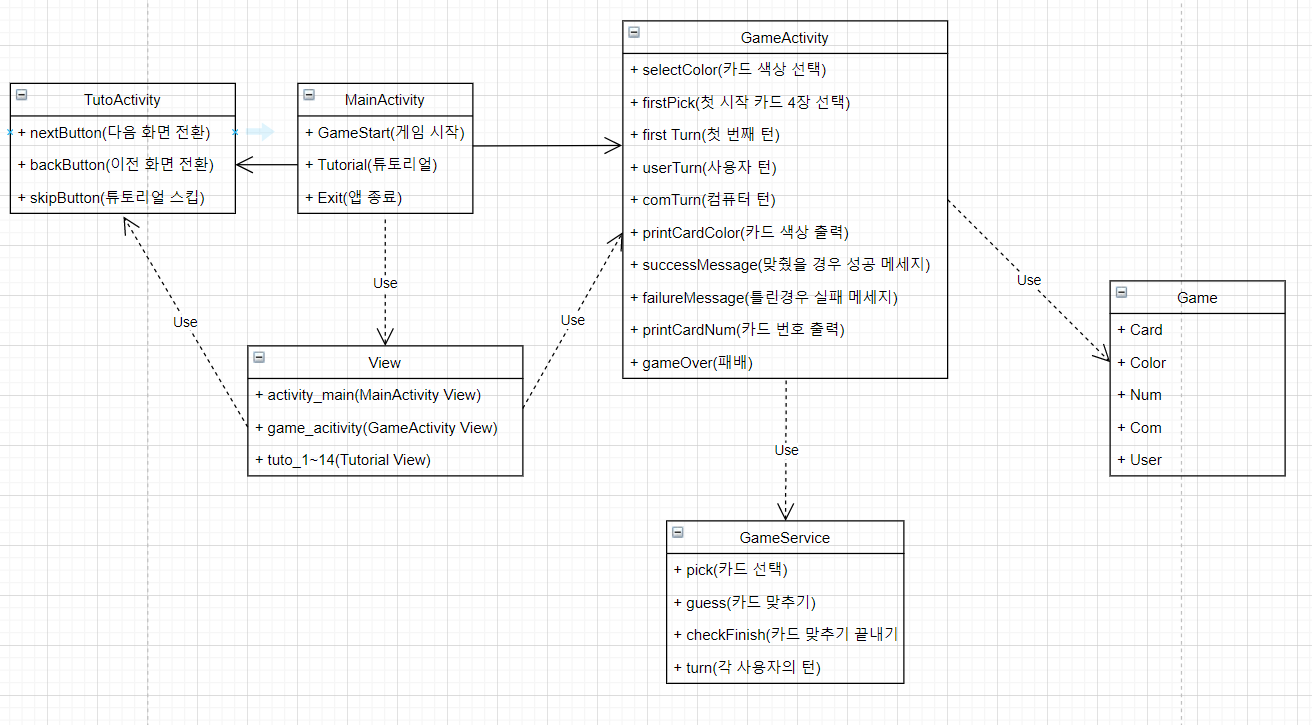
* 유추

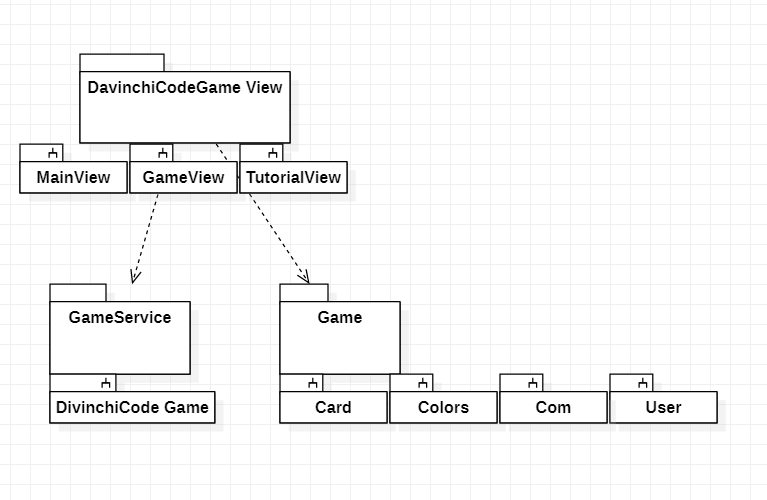


* 1. 요구사항 명세서

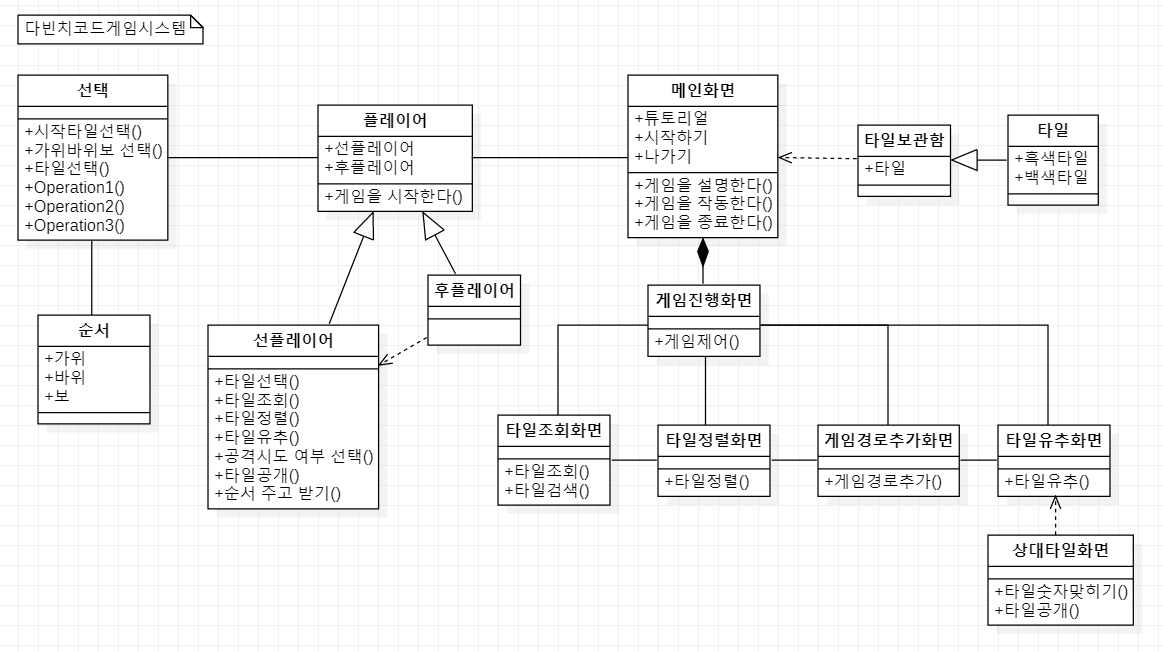
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **요구사항 명세서** | | | | | | |  |  |  |
|  |  |  |
| **작성처/작성자** | | | | | 다빈치코드게임시스템/  유양우, 김민지, 전재경 | |  |  |  |
| **작성일/버전** | | | | | 2020-06-15 / v1.0 | |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
| 비즈니스 요구사항 | | | | | 프로젝트 배경 | 논리적인 추리, 심리전, 운 등 3가지 재미있는 게임 요소를 적절하게  버무린 보드게임을 남녀노소 손 쉽게 해보자는 취지에서 출발했다. | |  |  |
| 프로젝트 목적 | 보드게임 ‘다빈치코드’를 진행하는 ‘다빈치코드게임시스템’을  계획 및 설계, 어플리케이션으로 구현한다. | |  |  |
| 프로젝트 목표 | 기존 출시 앱과 차별화: 확실히 숫자가 공개됐는지 안됐는지 UI를 구현 할 예정이다. | |  |  |
| 기존 출시 앱의 한계점: 개발을 더 하지않고 멈춰 있어 매칭과 서버 불안정  -> DB로 랭킹이랑 서버를 넣어 매칭하는 것 차차 추가 | |  |  |
| 성공기준 | 계획 및 설계를 통한 어플리케이션의 완벽한 구현 | |  |  |
| 비즈니스 규칙 | | | | | 사용사례 기술서, 다이어그램, 그리고 요구사항 명세서에 따른다. | | |  |  |
| 안드로이드 스튜디오 AVD를 통해 구현 UI를 실시간 확인할 수 있게 한다. | | |  |  |
| 시스템  연동사항 | | | | | 매칭과 서버는 DB서버와의 안정된 연결이 요구된다. | | |  |  |
|  |  |  |  |  |  |  |  |  |  |
| **ID** | | | | **구분** | **서비스(메뉴)** | **기능** | **요구사항** | **제약조건** | **진행사항** |
|  |
| 0 | 0 | 0 | 1 | 사용자 | 메인 화면 | 튜토리얼 | 다빈치코드의 기본적인 룰을 그림과 글로 설명한다. | 플레이어 | 반영 |  |
| 0 | 0 | 0 | 2 |  | 시작하기 | 게임을 시작한다. | 플레이어 | 반영 |  |
| 0 | 0 | 0 | 3 |  | 나가기 | 게임을 종료한다. | 플레이어 | 반영 |  |
| 0 | 1 | 0 | 1 | 게임 진행  화면 | 타일 조회 | 타일 조회 화면을 요청한다. | 플레이어 | 반영예정 |  |
| 0 | 1 | 0 | 2 |  | 타일 정렬 | 타일 정렬 화면을 요청한다. | 플레이어 | 반영예정 |  |
| 0 | 1 | 0 | 2 |  | 시작 타일 선택 | 시작 타일 선택을 요청한다. 타일 보관함에서 타일 4개를 선택한다. | 플레이어 | 반영예정 |  |
| 0 | 1 | 0 | 3 |  | 순서 정하기 | 게임 경로 추가화면을 출력한다. 가위, 바위, 보를 선택한다. | 플레이어 | 반영예정 |  |
| 0 | 1 | 0 | 4 |  | 타일 선택 | 타일 선택을 요청한다. 타일 보관함에서 타일 1개를 선택한다. | 플레이어 | 반영예정 |  |
| 0 | 1 | 0 | 5 |  | 상대 플레이어  타일 뒷면 보기 | 게임 경로 추가화면을 요청한다. | 플레이어 | 반영예정 |  |
| 0 | 1 | 0 | 6 |  | 상대 플레이어  타일 선택 | 게임 경로 추가를 요청한다. | 플레이어 | 반영예정 |  |
| 0 | 1 | 0 | 7 |  | 번호 맞히기 | 게임 경로 추가를 요청한다. | 플레이어 | 반영예정 |  |
| 0 | 1 | 0 | 8 |  | 공격시도 여부  선택 | 게임 경로 추가를 요청한다. | 플레이어 | 반영예정 |  |
| 0 | 1 | 0 | 9 |  | 상대 플레이어  타일 유추 | 상대 플레이어의 타일을 유추한다. | 플레이어 | 반영예정 |  |
| 0 | 1 | 1 | 0 |  | 공격시도 여부  선택 | 게임 진행 화면을 요청한다. | 플레이어 | 반영예정 |  |
| 0 | 2 | 0 | 1 | 타일 조회 화면 | 타일 조회 | 타일을 조회한다. | 플레이어 | 반영예정 |  |
| 0 | 2 | 0 | 2 |  | 맞힌 타일 정렬  조회 요청 | 맞힌 타일을 조회한다. | 플레이어 | 반영예정 |  |
| 0 | 3 | 0 | 1 | 타일 정렬 화면 | 타일 정렬화면  출력 | 타일 정렬을 출력한다. | 플레이어 | 반영예정 |  |
| 0 | 3 | 0 | 2 |  | 시작 타일 정렬  요청 | 시작 타일을 정렬한다. | 플레이어 | 반영예정 |  |
| 0 | 3 | 0 | 3 |  | 타일 정렬 요청 | 타일을 정렬한다. | 플레이어 | 반영예정 |  |
| 0 | 4 | 0 | 1 | 게임 경로 추가화면 | 경로 추가 | 게임 경로 추가 요청을 받으면 게임 경로 추가 화면을 출력한다. | 플레이어 | 반영예정 |  |
| 0 | 5 | 0 | 1 | 상대 타일 화면 | 상대 플레이어의 타일 유추 | 상대 타일 화면을 요청한다. | 플레이어 | 반영예정 |  |
| 0 | 5 | 0 | 2 |  | 타일 유추 화면  요청 | 타일 유추 화면을 요청해 출력한다. | 플레이어 | 반영예정 |  |
| 0 | 6 | 0 | 1 | 타일 유추 화면 | 타일 유추 | 타일의 예상 숫자를 입력한다. | 플레이어 | 반영예정 |  |
| 0 | 6 | 0 | 2 |  | 공격시도 여부  선택 | 게임 진행 화면을 요청한다. | 플레이어 | 반영예정 |  |
| 0 | 6 | 0 | 3 |  |  | 패스 | 순서를 넘긴다. | 플레이어 | 반영예정 |  |
| 1 | 0 | 0 | 0 | 관리자 | 게임컨트롤 | 게임 시스템 관리 | 게임 시스템을 관리한다. | 개발자 | 반영 |  |
| 1 | 0 | 0 | 1 |  | 매칭 | DB 서버 구축 | DB로 랭킹과 서버를 넣어 매칭하는 것을 관리한다. | 개발자 | 반영예정 |  |
|  |  |  |  | 공통 | 디자인 | 고객의 요구사항에 맞는 UI를 구현한다. | | | |  |
|  |  |  |  |  | 시스템 | 매칭과 서버는 DB서버와의 안정된 연결이 요구된다. | | | |  |
|  |  |  |  |  | 요구사항 분석 | 고객과 개발자의 정기적인 회의가 필요하다. | | | |  |
|  |  |  |  |  | 유지보수 | 미결된 사항들을 보완한다. | | | |  |
|  |  |  |  |  |  |  | | | |  |

1. 설계 및 구현
   1. 구조(아키텍처) 설계





* 1. 데이터 설계

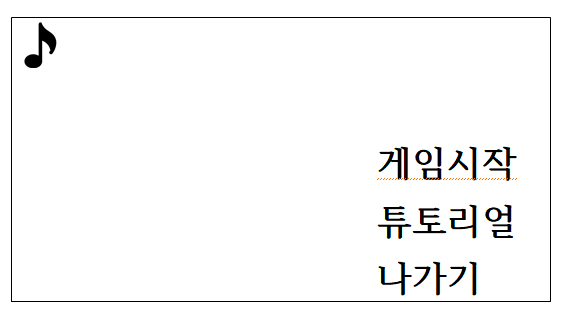


다빈치 코드라는 프로젝트 특성 상 이긴 횟수를 기록하여 점수로 측정한 다음 랭킹을 매기기 위하여 데이터베이스에 저장하는 것이 아니라면 데이터베이스가 필요하지 않기에 데이터베이스를 사용하지 않았습니다.

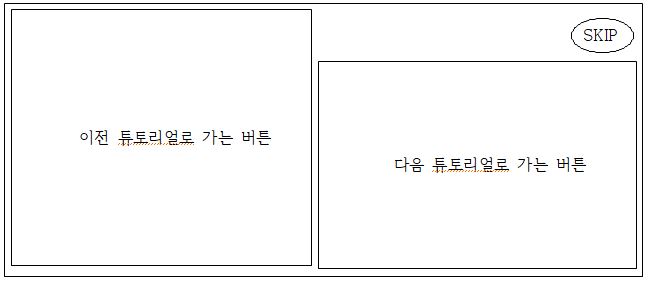
프로젝트 시스템을 좀 더 자세히 보여 드리기 위해 데이터베이스 설계도 부분은 클래스 다이어그램으로 대체합니다.

* 1. 인터페이스 설계

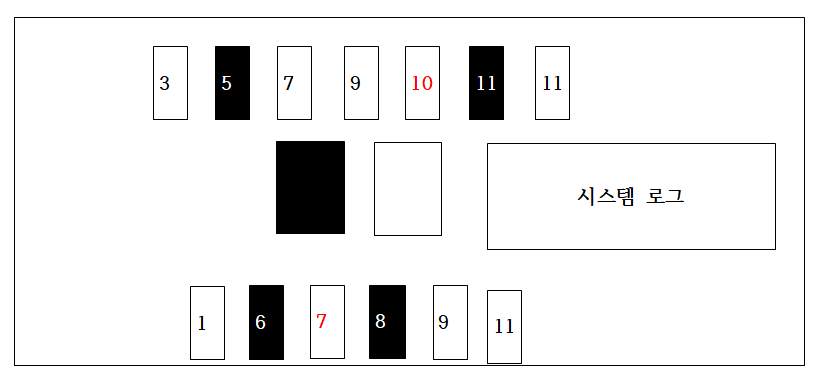
**[메인 화면]**



**[튜토리얼]**

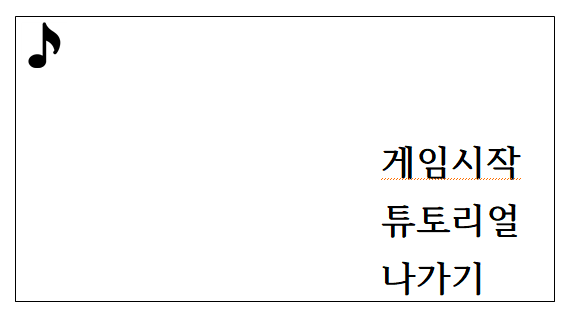
****

**[게임 시작]**



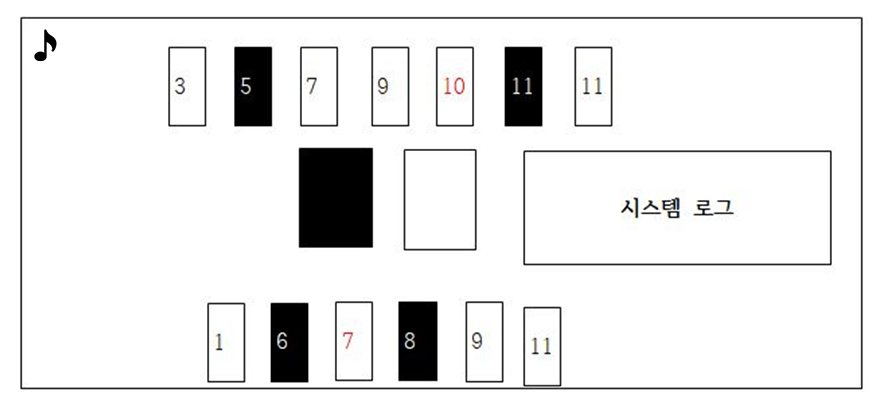
**<메인화면 구조 및 기능>**

**※ 모든 레이아웃은 가로 레이아웃을 기준으로 구현한다.**



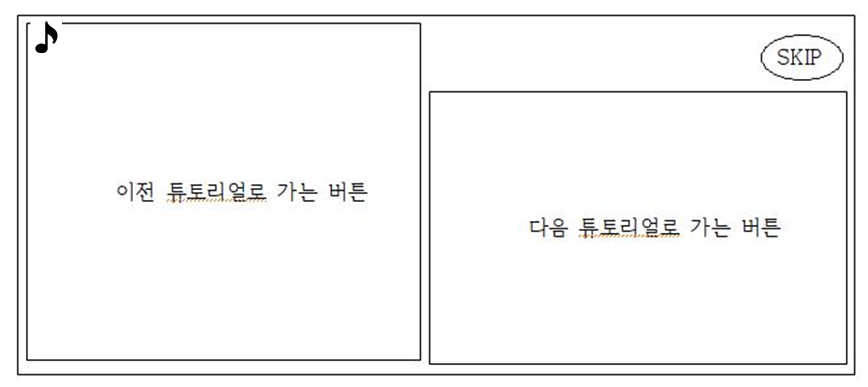
|  |  |  |
| --- | --- | --- |
| 이름 | 컴포넌트 종류 | 기능 |
| 게임시작 | 버튼 | 게임화면으로 들어갈 수 있는 버튼 |
| 튜토리얼 | 버튼 | 튜토리얼 화면으로 들어가 튜토리얼을 볼 수 있는 버튼 |
| 나가기 | 버튼 | 앱을 완전 종료 할 수 있는 버튼 |
| 음표버튼 | 버튼 | 소리를 on/off 할 수 있는 버튼 |

**<게임화면 구조 및 기능>**



|  |  |  |
| --- | --- | --- |
| 이름 | 컴포넌트 종류 | 기능 |
| 시스템 로그 | 레이아웃>텍스트뷰 | 게임의 진행상황을 보여줌으로써 무엇을 해야 하는지  누구의 차례를 표현 한다. |
| 자신의 카드 덱 | 이미지 뷰 | 자신이 가진 카드를 보여주는 이미지  상대가 내 카드를 볼 수 있는 숫자의 표시는  빨강으로 한다. |
| 상대방 카드 덱 | 이미지 뷰 | 상대방의 카드를 보여준다.  카드가 보이는 것은 빨간색 숫자로 표시한다. |
| 중간 덱 | 버튼 | 검정타일이나 흰색타일을 선택해서  자신의 덱으로 가져온다. |

**<튜토리얼화면 구조 및 기능>**



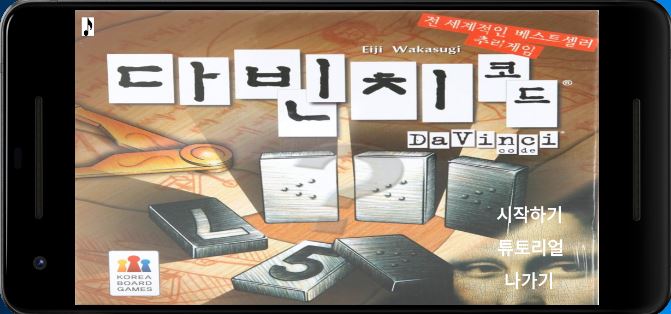
**※이미지를 화면에 나타내고 버튼은 투명으로 만들어 구성한다.**

|  |  |  |
| --- | --- | --- |
| 이름 | 컴포넌트 종류 | 기능 |
| Back 버튼 | 버튼 | 화면의 중간을 기준으로 좌측은 이전 튜토리얼 화면으로  전환 할 수 있다. |
| Front 버튼 | 버튼 | 화면의 중간을 기준으로 우측은 우측 튜토리얼 화면으로  화면을 전환 할 수 있다. |
| SKIP 버튼 | 버튼 | 스킵버튼을 통해 튜토리얼을 스킵 하고 메인화면으로  돌아갈 수 있다. |

* 결론

1. 구현 및 테스트 문서

**<메인화면>**



게임의 메인화면이다. UI그대로 게임 소리를 끄고 킬 수 있는 Button과

시작하기 튜토리얼 나가기라는 옵션으로 구성 되어있다.

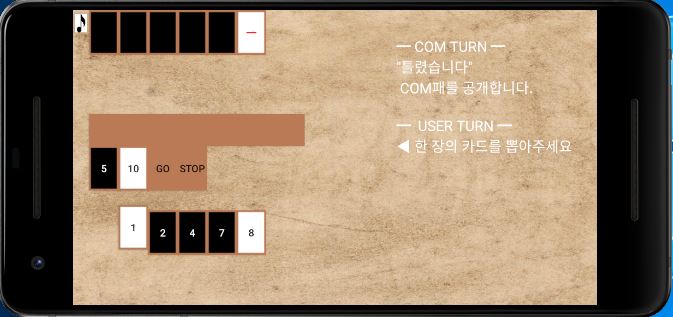
먼저 시작하기를 눌러서 게임시작을 해보겠다.

**<게임화면>**

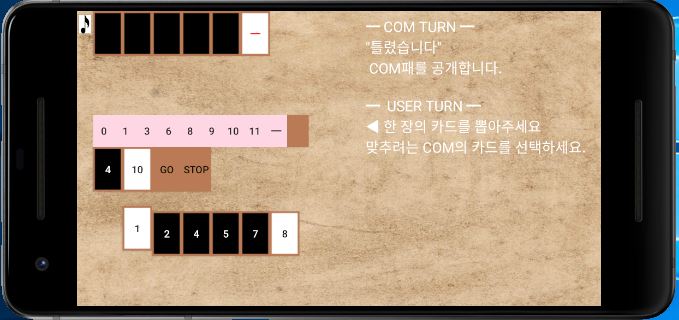


게임시작 첫 화면이다. 왼쪽 위에는 소리를 켜고 끌 수 있는 버튼과 오른쪽 상단에는

시스템 진행창이 턴을 가르쳐주고 진행하라는 대로 하면 된다. 검정패와 흰색 패를 클릭해 나의 덱으로 가져올 수 있다.



카드 5개를 뽑으면 컴퓨터가 먼저 공격을 실시한다. 컴퓨터가 나의 카드의 추측이 틀리면 빨간색으로 방금 뽑은 카드를 뒤집는다. 그리고 나의 공격이 시작된다.



카드를 한 장 뽑고 상대방 카드를 선택하면 아래와 같이 숫자가 나타난다. 현재 내가 가지고 있는 숫자들은 제외시켰다.



11을 맞추면 GO와 STOP중 골라야 한다. GO를 누르면 계속 공격 할 수 있고 STOP을 누르면 턴을 마친다. GO를 눌러보겠다.



GO를 누르고 또 맞춘 상황이다. 다시 GO와 STOP을 고를 수 있다. 다시 한번 GO를 눌러보겠다.



GO를 눌러서 틀리면 나의 카드가 빨간색으로 바뀌는걸 확인 할 수 있다.



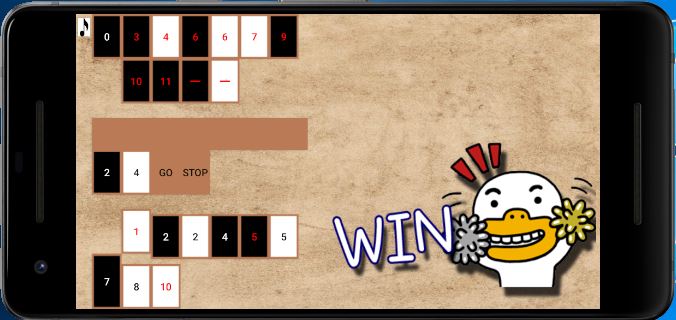
게임 진행 중간쯤 되는 사진이다.



상대방 카드를 맞추고 STOP을 눌렀다. 상태방에게 턴이 넘어간다.



게임이 진행되가며 다 맞춘 상황이다. STOP을 눌른다.

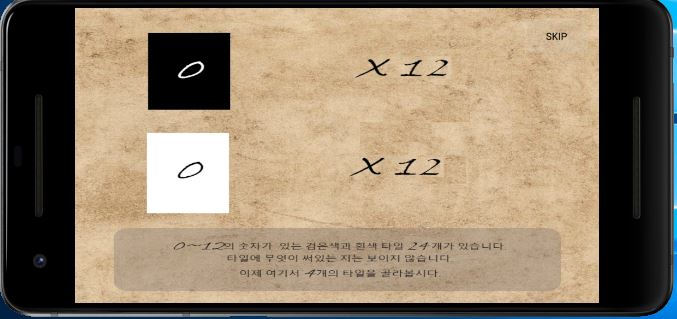


STOP을 누르면 이렇게 이겼다는 오리가 등장을 한다 !

**<튜토리얼>**

****

튜토리얼 첫 화면이다. 왼쪽 상단에 소리 on/off가 있고 우측 상단에는 SKIP버튼을 통해 튜토리얼을 스킵 할 수 있다. 이미지를 하나 씩 때 와서 우측 아무 대나 누르면 튜토리얼이 진행되고 좌측 아무대나를 누르면 튜토리얼 전 부분으로 갈 수 있게끔 만들었다.



우측 버튼 아무거나 누른상태이다. 그 다음 장으로 간 것을 확인 할 수 있다 이렇게 넘기면 

모든 설명을 마친 후 끝이 나게 된다.

* 1. 구현 결과물

코드가 너무 긴 관계로 통파일은 메일로 압축해 보내고 사이버캠퍼스에는 java코드와 xml코드만 첨부하겠습니다.

* 1. 소스코드 및 주석

<https://github.com/yuyw0712/Davinchi>

소스코드가 너무 많고 긴 관계로 깃허브 주소 첨부하겠습니다.

**아쉬운점**

1. 턴이 바뀔 때 마다 소리를 넣어서 하지 못했다.

2. 가끔 카드를 선택 할 때 에러가 나서 팅기는데 시간 부족으로 건들지 못했다.

3. 사람과 1:1 매칭을 구현하지 못한점이 아쉽다.

4. 게임은 애니메이션 효과가 있어야 게임인데 애니메이션까지 추가하지 못한점이 아쉽다.

5. 게임 구현이 어려워 UI에 집중하지 못했다.

6. 게임의 복잡도 때문에 조커카드를 항상 맨 오른쪽에 넣는다는 설정을 한 것이 아쉬웠다.

7. 컴퓨터가 무조건 선 턴으로 고정되는 것이 아쉬웠다.

8. 턴마다 제한시간을 설정했으면 좋았는데 아쉬웠다.

**소감**

이번 소프트웨어 공학 프로젝트를 통하여 '단계별 작업'에 대한 중요성을 알 수 있었다.

수업을 들으며 코딩이 어떻게 기계적인 일이라는 것일까? 라는 생각을 하며 들었는데

이번에 계획-요구사항분석-설계의 단계를 거치고 구현 단계에서 코딩을 해보니 게임 로직의 알고리즘이 어려웠던 것은 제외하고 기본 틀이 갖춰져 있어 다른 요소들은 금방

할 수 있었다. 구현을 하면서 설계 부분은 조금 씩 바뀌는 것 같다.

설계와 구현은 반복적으로 해야 하는 구나라는 것을 생각해 볼 수 있었다.

또 개인 프로젝트는 했었지만 협업의 경험이 별로 없었었다.

팀 프로젝트로 서로가 관리해가고 서로가 커밋한 걸

다음사람이 이어 나가는 협업을 해보는 중요한 시간 이였던 것 같다.

최대한 자바를 사용해 객체지향 적으로 가독성이 좋게끔 코딩 하고 싶었는데

아직은 이정도가 최선인 것 같다. 방학 때 더 기본을 갈고 닦아야 할 거란 걸 느꼈다.

간단한 게임조차도 구현하기 정말 어려운데 애니메이션까지 있고 캐릭터가 움직이는

RPG게임 같은 것은 얼마나 오래 걸리고 어려울지 짐작이 갈 수 있었다.

제한된 시간 때문에 완벽성을 갖추지 못한 것이 아쉽지만, 좋은 경험을 가질 수 있게 한 교수님께 감사를 드린다.

* 참고 문헌

인터넷 링크 All about board game – [www.divedice.com](http://www.divedice.com)

인터넷 링크 IT동아(게임동아) - <https://it.donga.com/21529/>

인터넷 링크 <https://blog.naver.com/kboardgame/80138808964>