# GRID SAGA기술 문서

# 개요

## 1.1 작성 목적

본 문서는2025/08/08부터 진행되는 Grid Saga의 개발을 위한 기술 지침 문서이다.

## 1.2 프로젝트 요약

Grid Saga의 명칭은 유니티 2D Grid상에서 이루어지는 서사시, 무용담이라는 의미에서 Grid에 Saga를 합성하여 Grid Saga로 명명하였다. 해당 게임의 컨셉은 실시간 턴제 방식으로 두 진영이 한 필드에서 전투를 치르는 방식이며, 타이머를 기반으로 전투 타이머 내에서 상대 진영의 hp를 모두 제거한 쪽이 우승하는 방식을 따른다. 타이머를 넘기거나 아군 진영 hp가 모두 제거될 경우 패배 처리한다.

# 개발 환경

|  |  |
| --- | --- |
| 개발 환경 | 용도 |
| Django (Python) | 백엔드 서버 |
| SQLite | DBMS |
| Unity (C#) | 클라이언트 엔진 |

# 시스템 아키텍처

텍스트, 스크린샷, 원, 폰트이(가) 표시된 사진

AI 생성 콘텐츠는 정확하지 않을 수 있습니다.

Grid Saga 시스템 아키텍처 구조는 Django 공식 MVT 구조를 따른다. MVT 구조는 Model – View – Template형식의 시스템 아키텍처를 말한다. MVT 구조에서 Template는 클라이언트를 담당하며, View는 클라이언트 요청에 따른 로직을 수행하고 이를 반환한다. Model은 DB 스키마 정의 및 ORM을 통한 DB 관리를 담당한다.

본 프로젝트의 시스템 아키텍처 구조는 해당 구조에 따라 Template은 Unity, View와 model은 Django로 한다. 이로써 렌더링, 게임 진행 코드는 유니티에 두고 데이터 송수신 및 관리는 Django에서 수행하도록 분리한다. Unity – Django 간 통신은 DRF를 통해 REST를 이용한다.

# 게임 설계

## 4.1. 게임 흐름 설계

### 4.1.1. 용어 정의

* 파티: 1 ~ 6캐릭터를 1 파티로 규정, 적의 경우도 동일
* 스테이지: 일반 몬스터 또는 보스가 존재하는 한 번의 전투가 벌어지는 공간
* 계층: 첫 일반 몬스터 ~ 보스까지 스테이지 묶음을 한 계층으로 정의
* 계층 타이머: 1계층 당 존재하는 시간 제한 타이머
* 피로도: 던전 진행에는 계정 별로 일일 횟수 제한을 둔다, 해당 제한 자원은 피로도로 명명

### 4.1.2. 게임 진행 방식

게임 진행은 pve 던전 공략 방식을 따른다, 다만 pvp 확장 가능성을 염두하고 설계한다.

* 던전입장: 플레이어 선택에 의해 던전에 입장, 피로도를 차감하여 DB에 반영한다
* 스테이지 진행: 스테이지를 순서대로 진행하며 계층 타이머가 모두 소진되기 전에 스테이지의 적을 전멸시키면 다음 스테이지로 이동한다, 다만 이동 과정서는 계층 타이머를 차감하지 않으며 스테이지에서 전투 중일 경우만 타이머를 차감한다
* 계층 클리어: 계층의 보스를 쓰러뜨리면 다음 계층으로 진행이 가능하다
* 타이머 규칙: 1 계층 1 계층 타이머 방식을 따른다, 계층에 플레이어가 진입하면 작동을 시작하고, 다음 계층으로 이동하면 해당 계층의 타이머를 새로 시작한다, 계층 클리어 이전에 타이머 시간이 되면 패배 처리한다
* 던전 나가기 조건: 파티 전멸, 타이머 아웃, 유저 의지에 따른 던전 나가기 처리
* 플레이어 선택: 던전 진행 중 방식을 수동 입력과 자동 진행으로 선택 가능

### 4.1.3. 던전 설계 방식

* 계층: 던전은 첫 몬스터 ~ 보스 몬스터의 묶음인 계층으로 나뉘어진다, 계층을 클리어하면 보상을 지급한 후 다음 계층으로 진행할지 선택 가능하게 한다
* 보상: 플레이어가 보스 스테이지를 넘겨 계층을 클리어하면 아이템 제공
* 난이도: 플레이어가 계층을 클리어하면 다음 계층 몬스터를 변경 및 강화를 병행하여 레벨 스케일링, 스탯 전반에 대한 퍼센트에 따른 레벨 스케일링 예정이나 정확한 퍼센트 수치 값은 미정 상태
* 캐릭터 사망: 계층 진행 중 전원이 사망하면 나가기, 한 캐릭터가 남아서라도 계층을 클리어 하면 생존 캐릭터는 100% hp로 회복, 사망한 파티 캐릭터는 hp를 50%로 부활
* 추가보상: 플레이어가 계층 진행 중 자동 진행을 한 번도 사용하지 않고 계층을 진행하면 추가 보상 제공, 이 외에는 특정 제한 시간 이하에 계층 클리어(ex: 타이머가 10분인 계층을 3분 내 클리어) 또는 파티 인원 전원 생존 상태에서 계층 클리어 등 고려
* UI 고려 사항: 현재 몇 계층인지, 남은 계층 타이머가 몇인지, 캐릭터 스킬 사용 가능 여부, 오토의 경우 스킬 사용 순서 큐, 파티 총합hp와 상대의 총합 hp

## 4.2. 캐릭터 설계

### 4.2.1. 캐릭터 기초 설계

캐릭터는 플레이어가 던전 공략에 사용할 수 있는 개체들이다.

* 보유 규칙: 플레이어는 여러 캐릭터를 소유 가능하나, 동일 캐릭터의 중복 소유는 불가하다
* 스테이터스: 기본적으로 Hp, Defense, Attack, Speed, Crit\_rate 가 존재한다. 플레이어가 소유한 캐릭터의 경우 해당 스테이터스 외에도 Level, Exp가 존재한다.

### 4.2.2. 스테이터스 별 역할

* Hp: 계층 진행 중 캐릭터 개인의 hp가 다 달면 해당 캐릭터는 해당 계층 내에서 사용 불능이 된다. 만약 캐릭터 모두의 총합 hp가 없어지면 던전 나가기 처리가 된다, 만약 계층을 모두 클리어 할 경우 사망한 캐릭터는 50%의 hp를 가지고 부활한다
* Defense: 들어오는 데미지를 감소시키기 위한 로직
* Attack: 상대에게 얼마나 데미지를 입힐지 결정, 일반 공격은 attack 만큼 스킬은 attack + (attack \* 스킬 계수)
* Speed: 공격 큐 상에서의 순서를 결정, 한 번의 공격 후 다음 공격까지 얼마나 걸릴지에 대한 쿨 다운에도 관여
* Crit\_rate: 크리티컬 확률, 발생시 1.5배의 데미지 적용
* Level: 특정 캐릭터의 Level을 결정한다. Level상승에 따라 캐릭터의 스테이터스가 상승한다
* Exp: 캐릭터의 경험치로서 캐릭터 강화 아이템을 통해 상승 가능하다, 해당 수치가 모두 찰 경우 Level을 상승시키고 Exp는 0으로 초기화한다. 만약 이전 레벨업에서 잔여 Exp가 있으면 해당 레벨의 새 Exp 바에 반영한다.
* 데미지 식: ATK = (Attack \* buff/debufff) \* if crit( \* 2)
* 방어식: damage = ATK \* (1 - DEF / (DEF + K)) 정확한 K 값은 테스팅하면서 결정

### 4.2.3. 스킬 유형

캐릭터를 패시브, 액티브 스킬, 아이템 스킬을 가진다

* 패시브 스킬: 4.4.1. 참고
* 액티브 스킬: 4.4.2. 참고
* 아이템 스킬: 해당 캐릭터가 소유한 아이템의 스킬, 만약 아이템을 미장착할 경우 null, 만약 액티브 스킬과 아이템 스킬이 동일할 경우 합연산하도록 한다(ex: 동일하게 된 스킬의 효과가 공격력 10% 증강이고 공격력이 100일 경우 100 \* 0.1 + 100 \* 0.1하여 120이 되도록)

## 4.3. 아이템 설계

아이템은 캐릭터 강화 아이템과 장착 아이템으로 나뉜다.

### 4.3.1. 캐릭터 강화 아이템

캐릭터 강화 아이템은 캐릭터에게 exp를 제공하는 1회성 재화이다. ItemType은 Consume으로 정의한다.

* 효과: 특정 캐릭터에게 소비할 경우 exp를 증가

### 4.3.2. 장착 아이템

장착 아이템은 캐릭터에게 장착되는 아이템으로 한 번 획득하면 플레이어 소유는 영구적이다. ItemType은 Equip으로 정의한다.

* 효과: 장착 아이템을 소지한 캐릭터가 가진 스테이터스 중 하나를 퍼센트 또는 값만큼 증가
* 아이템 스킬: 특정 액티브 스킬을 캐릭터에게 적용
* 중복: 이미 플레이어가 소유한 아이템을 획득하는 경우 캐릭터 강화 아이템으로 대신 지급한다.
* 제한: 캐릭터가 아닌 장비가 진행 주체가 되는 현상을 막기 위해 패시브 스킬을 아이템 스킬에 배치는 금함

## 4.4. 스킬 설계

스킬은 패시브 스킬, 액티브 스킬로 나뉜다. 두 스킬은은 모두 트리거를 기반으로 작동하나 트리거의 조건에서 차이가 발생한다.

### 4.4.1. 패시브 스킬

패시브 스킬은 특정 조건을 트리거로 받아 자동 발동되는 스킬이다. 주로 캐릭터 자신에 대한 버프를 효과로 한다.

* 트리거: 캐릭터 체력 상황, 조우한 적 타입
* 효과: 패시브 스킬을 소유한 캐릭터의 스테이터스를 증가

### 4.4.2. 액티브 스킬

액티브 스킬은 수동 모드에서 플레이어의 직접 입력, 자동 모드에서 큐 스택 차례를 트리거로 받아 발동되는 스킬이다. 한 번 사용 후 다음 사용까지 쿨다운이 필요하며 상대에게 직접 피해, 디버프, 아군에게 일시적인 강한 버프 등의 영향을 끼치는 스킬을 구조로 한다.

* 트리거: 플레이어의 직접 입력, 자동 모드에서 캐릭터의 큐 스택 차례가 올 경우
* 효과: 상대 및 아군에게 직접 피해, 일시적인 강한 버프, 디버프 영향
* 조건: 강한 성능을 가지지만 쿨다운이 필요

# 5. UX/UI

## 5.1. 로그인 UI

로그인 UI는 플레이어가 게임 실행 직후 마주치는 UI로서 유저의 로그인 가능 여부를 다룬다

* 인증: 아이디, 패스워드를 입력 받아 인증하기 위한 UI
* 새 계정 생성: 새로운 유저가 새 계정을 생성하기 위한 UI

## 5.2. 로비 UI

로비 UI는 로그인 UI에서 계정 정보를 인증한 유저가 마주치는 UI로서 유저의 정보, 게임 플레이 진입 여부, 캐릭터와 아이템을 비롯한 재화에 대한 정보를 제공한다

### 5.2.1 메인 로비

* 유저 정보: 메인 로비의 유저 UI는 간략한 유저 명칭 정도만 제공한다, 향후 던전 공략도, 랭킹 등 추가 세부 요소가 있을 경우 유저 UI를 클릭하여 자세한 정보를 확인하는 시스템을 채택한다
* 재화: 캐릭터, 아이템 재화는 파티 관리 세부 정보에서 확인 가능하게 한다
* 던전 진입: 클릭하면 확인용 UI를 띄워주고 확인 시 피로도를 차감하고 던전에 진입한다

### 5.2.2 파티 관리

파티 관리는 아이템 관리, 캐릭터 관리 둘로 나눈다.

* 아이템 관리: 아이템의 소지 여부, 개수 확인이 가능하다. 사용 아이템과 장착 아이템 관리 칸은 별개로 한다
* 캐릭터 관리: 캐릭터의 소지 여부, 레벨 확인이 가능하다 세부 스테이터스, 캐릭터 경험치 현황, 장착 아이템 등은 캐릭터를 클릭하면 세부 정보로 볼 수 있게 한다

## 5.3. 던전 UI

던전 UI는 던전에 진입한 후 확인하게 되는 UI로서 주로 전투 관련한 정보를 제공한다.

### 5.3.1 전투 중 UI

* 필드: UI 중간에 위치하며 필드에는 적과 아군이 위치한다
* 스테이지 정보: 현재 몇 계층인지와 몇 스테이지인지를 계층 – 스테이지 방식으로 제공한다
* 계층 타이머: 현재 계층 타이머가 몇 분 남았는지 (분 : 초)로서 제공한다
* 액티브 스킬 사용 가능 여부: 캐릭터의 액티브 스킬 사용 가능 여부를 표시한다, 쿨타임이라면 짙은 색으로 만든 후 radial로 원래 색깔로 돌아오게 한다
* 패시브 스킬 발동: 버프라면 녹색 화살표를 캐릭터 주변에 표시한다, 디버프라면 빨간색 화살표를 아래로 가게 하여 캐릭터 주변에 표시한다
* 총합 hp: 파티원의 총합 hp를 표시한다
* 개벌 hp: 파티원의 개별 hp를 표기한다, 정확한 위치는 미정 상태
* 적 총합 hp: 적의 총합 hp를 표시한다

### 5.3.2 전투 후

* 일반 스테이지의 경우: 다음 스테이지로 진행한다
* 보스 스테이지의 경우: 수령한 보상 정보를 제공한 후 다음 계층으로 이동할지 나갈지 선택하게 한다
* 나가기(패배, 타임 아웃, 사용자 선택 등): 지금까지 획득한 자원을 보여준 후 로비로 돌려보낸다

# 6. 구현 세부 계획

## 6.1. 던전 구현

* 해당 단계에서는 던전에서의 진행 로직을 우선적으로 구현한다
* 던전 내부에서 사용되기 위한 전투 로직, 매니저, 캐릭터, 아이템을 구현한다
* 이 부분에서 구현할 것은 어떠한 캐릭터들이, 어떠한 아이템을 소지하였을 경우, 스킬의 발동은 어떻게 이루어 지는지, 진행은 어떻게 되는지, 게임오버는 어떻게 처리되는지 매니저는 무엇이 필요한지에 집중한다

## 6.2. 로비 구현

* 이 부분은 로비를 구현한다
* 이 부분은 플레이어의 정보를 해당 정보가 구현된 던전에서 실사용 가능하게 한다
* 이 단계에서의 데이터 관리는 로컬로 이루어 진다

## 6.2. 서버 구현

* 실제 서버를 구현한다
* 플레이어의 정보와 데이터들의 정보를 DB 서버에 저장하는 로직을 만든다
* ID/PW를 통한 인증도 해당 단계에서 구현한다
* 현재 REST, JWT을 사용하여 연결 및 인증하는 것이 계획이 된다