GenRoom AI

Software Requirements Specification(SRS)

과목명: 졸업 프로젝트
팀원 : 안재민, 김인교, 노을영
담당교수: 유준범
제출일: 2026.05.12

Content

1. Introduction

1.1 Purpose

본 문서는 3D 공간 자동 생성 시스템 GenRoom AI의 소프트웨어 요구사항 명세(Software Requirements Specification, SRS)이다.

본 문서는 다음 목적을 위해 작성되었다.

1. GenRoom AI의 기능·비기능 요구사항을 정의한다.
2. 모듈 간 인터페이스(JSON 스키마)를 명시하여, 모듈 독립성과 확장성을 확보한다.
3. 개발·테스트·인수 과정에서 요구사항 추적(Traceability)의 근거 문서로 활용한다.
4. 유지보수 및 기능 확장 시 영향 범위 분석의 기준을 제공한다

대상 독자

독자	활용 목적
개발팀	모듈 설계·구현·연동
QA / 테스트	테스트 케이스·인수 기준 작성
시스템 유지보수 담당	기능 변경·확장 시 영향 분석

1.2 Scope

GenRoom AI는 사용자의 자연어 입력을 받아, 공간 내 가구 배치 좌표(JSON)와 3D Mesh(GLB)를 생성하고, Unity를 통해 완성된 3D 방을 제공하는 시스템이다.

시스템은 다음 4개의 독립 모듈로 구성된다.

모듈	역할
Input Module (FR-1)	자연어 입력 수집, 프롬프트 표준화, 1차 메타데이터 JSON 생성
Positioning Module (FR-2)	가구 위치·회전·스케일 생성, 좌표 검증·보정
3D Modeling Module (FR-3)	가구별 3D Mesh 생성, 스케일·방향 검증
Generation Module (FR-4)	JSON·Mesh를 Unity 환경에 배치·렌더링

각 모듈은 상호 간섭을 최소화하며, 정형화된 JSON 포맷의 데이터를 통해서만 연동한다. 모듈 간 데이터 전달은 로컬 JSON 파일을 기본 매개체로 한다.

시스템 산출물
- 가구 배치 정보: JSON
- 3D 모델: GLB
- 최종 출력: Unity 기반 3D 방 환경

Input Module은 User Interface(3.1.1)와 Functional Requirement(3.2.1)로, 나머지 모듈은 Functional Requirement(3.2.2~3.2.4)로 기술한다.

1.3 Definitions, acronyms and abbreviations

용어	정의
GenRoom AI	본 프로젝트의 시스템 명칭. 자연어 기반 3D 실내 공간 자동 생성 시스템
Input Module	사용자 자연어 입력을 받아 표준화된 프롬프트·메타데이터 JSON을 생성하는 모듈. UI를 포함한다(FR-1)
Positioning Module	Input Module 출력을 바탕으로 가구의 위치(position), 회전(rotation), 스케일(scale)을 생성·검증·보정하는 모듈(FR-2)
3D Modeling Module	Positioning Module 출력을 바탕으로 가구별 3D Mesh 파일을 생성·검증하는 모듈(FR-3)
Generation Module	Positioning·3D Modeling 결과(JSON + Mesh)를 Unity에 배치·렌더링하는 모듈(FR-4)
Layout JSON	방·가구 배치 정보를 담은 JSON.room_type, room_size, atmosphere, furniture_number, objects 등을 포함
room_type	방 유형(예: BedRoom, LivingRoom)
atmosphere	방 분위기·테마(예: Horror, Modern)
furniture_number	배치 대상 가구 수(정수)
Module Interface JSON	모듈 간 주고받는 정형화된 JSON. SI-1~SI-3에 정의
Bounding Box (BB)	가구가 차지하는 직육면체 영역. 배치 검증·Collision 판정·Mesh 스케일 맞춤에 사용
Mesh / 3D Model	가구의 3D 기하 데이터
GLB	Binary glTF 형식의 3D 모델 파일
Collision	두 가구 BB가 부피적으로 겹치는 상태. Edge Contact(면 접촉)는 허용
Wall Attachment	가구 한 면이 방 벽에 밀착·정착된 배치 상태

1.4 References

REF-1. IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications
REF-2. Google, Gemini API — Generative Language
REF-3. DiffuScene 등, Scene Layout Generation 연구·구현
REF-4. Unity Technologies, Unity Manual — GLB/FBX Import, Runtime Scene Loading
REF-5. GenRoom AI, JSON Interface Specification (본 문서 3.1.2 Software Interfaces)

1.5 Overview

제 1장: 시스템 개요 및 문서 목적 설명
제 2장: 시스템 주요 기능 및 프로젝트 전제 환경 설명
제 3장: 각 컴포넌트별 기능 요구사항 세부 설명. Module를 기준으로 세분화하여 제시한다.

2. Overall description

2.1 Product perspective

GenRoom AI는 사용자의 자연어 입력을 통해 실내 공간의 가구 배치 정보(JSON)와*3D Mesh(GLB/FBX)를 자동 생성하고, Unity 환경에서 완성된 3D 방을 제공하는 시스템이다.

기존 절차적(Procedural) 방 생성 도구는 정해진 규칙에 따라 반복적인 결과만 만들 수 있고, 수동 Unity 배치는 시간·전문성이 많이 든다. GenRoom AI는 생성형 AI의 창의성과 모듈별 검증·보정 파이프라인을 결합하여, 비전문가도 짧은 시간에 플레이·개발에 활용 가능한 실내 공간을 얻을 수 있도록 한다.

이 시스템은 4개의 주요 Module로 구성되며, 각 Module는 정형화된 데이터를 Output으로 산출하여 다음 Module의 Input으로 사용하게 된다. 기존에 구현되어있는 LLM, 방 배치 AI(Respace 등), 3D Mesh 생성 AI, Unity Engine을 각각 사용하는 4개의 Module로 구성되어 있으며, GenRoom AI는 AI 출력을 그대로 최종 결과로 사용하지 않는다.

단계	검증·보정 내용
Input Module	JSON 형식·데이터 타입·양수 room_size 등 입력 정합성 검증 (FR-1.6)
Positioning Module	벽·바닥 관통, BB Collision, Wall Attachment, 가구 수·방 크기 공간 정합성 보정 (FR-2.2~2.6)
3D Modeling Module	Mesh 스케일·방향이 Positioning BB와 기하 정합성 일치 (FR-3.3~3.4)
Generation Module	JSON 좌표와 Mesh 1:1 매칭 배치, 최종 간섭 없음 (FR-4.2, FR-4.6)

2.2 Product functions

본 절은 사용자 관점의 제품 기능을 정의한다. 세부 요구사항은 제3장 Functional Requirement를 따른다.

ID	사용자 기능	담당 모듈	관련 요구사항	우선순위
PF-1	자연어로 방·가구·테마·크기 요구 입력	Input	FR-1.1, FR-1.2, UI-1	Must
PF-2	입력을 분석해 room_type, room_size, atmosphere, furniture_number 등 JSON 생성	Input	FR-1.3~FR-1.7, SI-1	Must
PF-3	가구 배치 좌표(위치·회전·스케일)	Positioning	FR-2.1, SI-2	Must

	자동 생성			
PF-4	배치 좌표 검증·보정(벽 밀착, Collision 해소, 가구 수·방 크기 정합)	Positioning	FR-2.2~FR-2.6	Must
PF-5	가구별 3D Mesh(GLB/FBX) 자동 생성	3D Modeling	FR-3.1~FR-3.2, SI-3	Must
PF-6	생성 Mesh의 스케일·방향을 Positioning BB에 맞게 보정	3D Modeling	FR-3.3~FR-3.4	Must
PF-7	Mesh 생성 실패 시 Default 가구 모델로 대체	3D Modeling	FR-3.5, QA-3	Must
PF-8	JSON·Mesh를 Unity에 배치해 완성된 3D 방 출력	Generation	FR-4.1~FR-4.4	Must
PF-9	AI 생성 과정의 진행 상황·안내 메시지 표시	Input (UI)	UI-3, QA-5	Must
PF-10	모듈 간 JSON 파일 저장·전달	전 모듈	DC-2, FR-1.7, FR-2.7, FR-3.6	Must
PF-11	잘못된 입력·API 오류 시 시스템 중단 없이 오류 안내	Input, 전 모듈	FR-1.2, FR-1.5, QA-8	Must
PF-12	최종 3D 방의 가구 간 물리적 간섭 없음 보장	Generation	FR-4.6, QA-6	Must

2.3 User characteristics

- GenRoom AI의 주요 사용자는 월드·레벨 제작에 많은 시간을 투자하기 어려운 인디 게임 개발자이다.

항목	특성
목표	짧은 시간에 플레이 가능한 실내 공간·레벨 제작
Unity 숙련도	중급 — 씬 편집·에셋 import 가능, 고급 스크립팅은 필수 아님
3D 모델링	직접 제작 역량 제한적 → AI 생성 Mesh 활용
입력 선호	자연어 입력 선호, JSON 직접 편집은 고급 사용자만
기대 산출물	Unity 씬에서 즉시 활용 가능한 GLB/FBX + 배치된 3D 방
Pain Point	반복적 방 배치, 에셋 수집·배치, 3D 모델 제작 시간

2.4 Constraints

2.4.1 Technical and Architecture Constraints

ID	제약	관련 **DC/FR**
TC-1	모듈 간 데이터 전달은 JSON 파일만 사용한다	DC-2
TC-2	모듈 간 JSON은 UTF-8 인코딩을 따른다	CI-1
TC-3	좌표 데이터는 JSON, 3D 모델은 GLB·FBX 형식을 준수한다	DC-5
TC-4	Positioning Module 및 3D Modeling Module은 Python으로 구현한다	DC-7
TC-5	Input Module UI 및 Generation Module은 C# (Unity) 로 구현한다	DC-4, DC-7
TC-6	Generation Module의 출력 엔진은 Unity를 사용한다	DC-3
TC-7	한 모듈의 오류가 다른 모듈에 직접 전파되지 않도록 독립 예외 처리를 갖춘다	DC-6, QA-8
TC-8	API Key·인증 토큰은 소스 코드에 노출하지 않고 환경 변수 등으로 관리한다	QA-1

2.4.2 Performance Constraints

ID	제약	상세
PC-1	3D Mesh 1개 생성 시간 ≤ 3분	3.3 NFR 참조
PC-2	방 배치 좌표 1회 생성 시간 ≤ 2분	3.3 NFR 참조
PC-3	JSON·Mesh 종합 후 최종 방 출력 ≤ 1분	3.3 NFR 참조

2.5 Assumptions and dependencies

2.5.1 Assumptions

ID	**Assumptions**
A-1	사용자(또는 배포 환경)는 안정적인 인터넷 연결을 갖춘다
A-2	외부 API Key·인증 토큰은 사전에 설정되어 있다
A-3	Unity 프로젝트는 GLB·FBX import가 가능한 상태이다
A-4	Input Module의 LLM은 JSON 추출·파싱 가능한 형태로 응답한다
A-5	Positioning·3D Modeling AI는 유사 API로 대체 가능하며, 대체 시 JSON 인터페이스(SI)는 유지된다 (QA-7)
A-6	모듈 간 JSON 파일 저장 경로에 쓰기 권한이 있다
A-7	사용자는 1.6 Usage Scenarios(SC-1~4)와 유사한 정상적 자연어 입력을 제공한다

2.5.2 Dependencies

ID	의존 대상	사용 모듈	용도	대체 가능
D-1	Gemini API (또는 동등 LLM API)	Input	자연어 분석·프롬프트 표준화	가능 (QA-7)
D-2	DiffuScene 등 장면 배치 AI	Positioning	가구 좌표 생성	가능 (QA-7)
D-3	3D Mesh AI	3D Modeling	Text-to-3D Mesh 생성	가능 (QA-7)
D-4	Unity Engine	Generation	3D 방 렌더링·배치	불가 (DC-3)
D-5	Python 3.x Runtime	Positioning, 3D Modeling	AI 연동·검증·보정 로직	버전 범위 내 대체

3. Specific requirements

3.1 External interface requirements

본 절은 GenRoom AI가 외부(사용자·하드웨어·타 소프트웨어·통신) 와 주고받는 인터페이스를 정의한다. Input Module의 사용자 인터페이스, 모듈 간 JSON 데이터 명세(SI), 통신 규약(CI), 하드웨어 요구(HI)를 포함한다.

3.1.1. User Interface

Input Module의 사용자 인터페이스(UI)는 웹 기반 단일 페이지 UI로 제공한다. 검색창 형태의 자연어 입력을 중심으로, 입력 단계(Input Stage) 와 생성 단계(Generation Stage) 두 화면으로 구성한다.

UI-S1. Input Stage (입력·레이아웃 확인 화면**)**

(UI-1.1,1.2,1.3 예시)

UI-1.1 서비스 타이틀

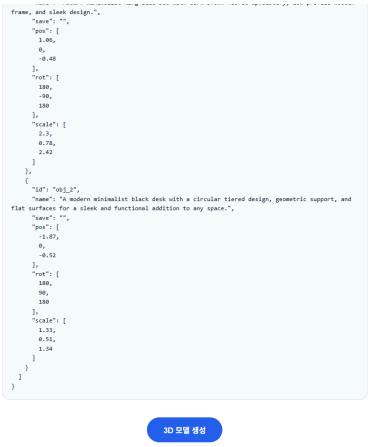
항목	요구사항
표시 내용	GenRoom (또는 GenRoom AI)
위치	화면 상단 중앙
우선순위	Must

UI-1.2 자연어 입력창

항목	요구사항
형태	단일 행 텍스트 입력 필드 (검색창 스타일, 둥근 모서리)
Placeholder	방 설명을 입력하세요...
입력 방식	키보드 입력, Enter 또는 제출 동작으로 레이아웃 생성 요청
자동 포커스	화면 진입 시 입력창에 포커스
우선순위	Must

UI-1.3 안내

항목	요구사항
기본 문구	Enter로 공간 좌표를 생성합니다 / 공간 좌표 생성 중입니다.. 잠시만 기다려주세요
표시 조건	입력 대기 상태에서만 표시 / Positioning Module 처리 중에서만 표시
우선순위	Must



(UI-1.4,1.5 예시)

UI-1.4 레이아웃 JSON 표시

항목	요구사항
형태	읽기 전용 <pre> 영역 (고정폭 글꼴, 스크롤 가능)
표시 내용	Positioning Module 결과 JSON (SI-2 형식, pretty-print)
표시 조건	레이아웃 생성 성공 후에만 표시
우선순위	Must

UI-1.5 3D 모델 생성 버튼

항목	요구사항
라벨	3D 모델 생성
형태	Primary 버튼 (강조 색상, pill 형태)
활성 조건	레이아웃 JSON 표시가 완료된 후
동작	클릭 시 Generation Stage(UI-S2)로 전환, 3D Modeling Module 처리 시작
우선순위	Must

UI-S2. Generation Stage (3D 생성 진행 화면)

(UI-2.1,2.2 예시)

UI-2.1 진행 제목

항목	요구사항
처리 중	3D 모델 생성 중입니다
완료 후	3D 모델 생성이 완료되었습니다
우선순위	Must

UI-2.2 전체 진행률

항목	요구사항
형태	수평 progress bar (0~100%) / “2 / 5 모델 생성 완료” Text
갱신 기준	완료된 가구 수 / 전체 가구 수, 현재 가구의 세부 progress 반영
우선순위	Must

3.1.2 Software interfaces

모듈 간 데이터 교환은 JSON 파일 3단계(SI-1 → SI-2 → SI-3) 로 정의한다. 각 파일은 UTF-8 인코딩, 단일 JSON 객체 형식을 따른다.

항목	규칙
파일명	genroom_si1.json, genroom_si2.json, genroom_si3.json
저장 위치	{WORKSPACE}/data/ (환경 변수 GENROOM_DATA_DIR로 설정, 기본값 ./data)
덮어쓰기	각 Module은 이전 단계 파일을 읽은 뒤 동일 경로에 전체 JSON을 덮어쓰기
단위	길이: meter (m), 각도: degree (°)
좌표계	Unity 연동 왼손 좌표계, Y-up. 방 바닥 중심을 원점 (0, 0, 0) 으로 한다
room_size 축	x = width(가로), y = height(높이), z = depth(세로)

SI-1. Input Module → Positioning Module

```
{
"room_type": "BedRoom",
"room_size": { "x": 6, "y": 5, "z": 3 },
"atmosphere": "Horror",
“furniture number”: 7,
}
```

SI-2. Positioning Module → 3D Modeling Module

```
{
  "room_type": "BedRoom",
  "room_size": {
    "x": 6.0,
    "y": 3.0,
    "z": 5.0
  },
  "atmosphere": "Horror",
  "furniture_number": 7,
  "objects": [
    {
      "id": "obj_1",
      "name": "stone_fireplace",
      "category": "fireplace",
      "position": { "x": -4.15, "y": 0.0, "z": -2.10 },
      "rotation": { "x": 0.0, "y": 90.0, "z": 0.0 },
      "scale": { "x": 1.75, "y": 2.70, "z": 0.95 }
    }
  ]
}
```

SI-3. 3D Modeling Module → Generation Module 명세서

```
{
  "room_type": "BedRoom",
  "room_size": { "x": 6.0, "y": 3.0, "z": 5.0 },
  "atmosphere": "Horror",
  "furniture_number": 7,
  "objects": [
    {
      "id": "obj_1",
      "name": "stone_fireplace",
      "category": "fireplace",
      "position": { "x": -4.15, "y": 0.0, "z": -2.10 },
      "rotation": { "x": 0.0, "y": 90.0, "z": 0.0 },
      "scale": { "x": 1.75, "y": 2.70, "z": 0.95 },
      "mesh_file": {
        "path": "meshes/obj_1_stone_fireplace.glb",
        "format": "glb",
        "is_default": false
      }
    }
```

```
  ]
}
```

3.1.3 Communication interfaces

CI-1. 모듈 간 **JSON** 파일 통신
CI-1.1 모듈 간 데이터는 SI-1~SI-3 JSON 파일로만 전달한다 (DC-2)
CI-1.2 인코딩: UTF-8
CI-1.3 JSON은 schema_version 필드를 포함해야 한다
CI-1.4 Module은 입력 파일 존재·파싱·필수 필드 검증 후 처리를 시작한다
CI-1.5 출력 파일은 처리 성공 시에만 덮어쓴다. 실패 시 이전 파일 유지

CI-2. 외부 **AI API** 통신
CI-2.1 Input Module LLM (Gemini API 등) 대상으로 한 통신으로는 HTTPS / REST api를 활용한다.
CI-2.2 Positioning AI (DiffuScene 등) 대상으로 한 통신으로는 HTTPS / REST api를 활용한다.
CI-2.3 3D Mesh AI 대상으로 한 통신으로는 HTTPS / REST api를 활용한다.
CI-2.4 요청·응답 Content-Type: application/json (API 제공자 스펙 따름)
CI-2.5 API Key·토큰은 환경 변수로 전달, 소스 코드 비노출 (QA-1)
CI-2.6 네트워크 타임아웃·Quota 초과 시 FR-1.5, FR-3.5 예외 처리 적용
CI-2.7 API 교체 시 SI-1~SI-3 스키마는 유지 (QA-7)

3.2 Functional Requirement

3.2.1 FR-1 Input Module

*Input Module*은 웹 *UI(3.1.1 UI-S1)* 를 통해 자연어를 수집하고*, LLM*을 이용해 *SI-1 JSON*을 생성·저장한다.

FR-1.1 사용자 자연어 입력 수집
- UI-1.2(Prompt Input)를 통해 사용자로부터 자연어 형태의 방 생성 요청을 수집한다.
- Enter 또는 동등한 제출 동작으로 레이아웃 생성을 요청한다. 제출 시 UI-1.4(레이아웃 생성 중)를 표시하고 입력창을 비활성화한다.

FR-1.2 입력 유효성 검증
- 유효하지 않은 입력을 검증하며, 발견될 경우 입력창을 재활성화하고 재입력을 유도한다.
- 유효하지 않은 입력은 다음과 같다.
    1. 빈 문자열 또는 공백만으로 구성된 입력
    2. LLM 처리 후에도 SI-1 필수 필드(room_type, room_size, atmosphere)를 유효한 JSON 값으로 정의할 수 없는 입력

FR-1.3 LLM 기반 메타데이터 추출
- 사용자로부터 입력받은 일반 텍스트를 LLM으로 보내 SI-1에 해당하는 Json 데이터로 변환한다.

FR-1.4 LLM API 통신 및 오류 처리
- LLM API(Gemini API 등)를 호출하고 오류를 처리한다.
- 네트워크 타임아웃, Quota 초과, 응답 파싱 실패 시 UI-1.7에 오류 메시지를 출력한다.
- 오류 발생 시 오류 메세지를 통해 재입력을 유도한다.

FR-1.5 SI-1 데이터 정합성 검증
- LLM 출력을 통해 구성된 SI-1의 타입·값을 검증하며, 검증 실패시 FR-1.2와 동일하게 오류 처리를 진행한다.
- 검증 기준은 아래와 같다
    1. room_type = 비어 있지 않은 string
    2. room_size = number(>0)
    3. atmosphere = 비어 있지 않은 string

FR-1.6 JSON 파일 저장
- 검증된 SI-1을 로컬 JSON 파일로 저장한다.
- 저장에 성공 하면 Positioning Module 처리를 트리거한다.

FR-1.7 Positioning 및 **Modeling** 과정 **UI** 표시
- Positioning Module 및 Modeling Module의 진행과정을 UI-1.4, UI-1.5, UI-1.6, SI-2에 웹 UI에 표시한다.

FR-1.8 3D 생성 단계 전환
- UI-1.6에 따라 사용자가 「3D 모델 생성」을 클릭하면 Generation Stage로 전환한다.

3.2.2 FR-2 Positioning Module

*Positioning Module*은 *SI-1*을 읽어 가구 배치 *SI-2*를 생성한다. 현 프로젝트에 사용하는 *Positioning* 엔진은 *ReSpace*이며, 엔진 교체 가능하도록 *JSON* 입·출력 양식을 고정한다.

FR-2.1 배치 **JSON** 생성
- SI-1과 사용자 프롬프트를 바탕으로 가구 objects[] 초안 JSON을 생성한다.
- 1차적인 구현은 ReSpace 엔진을 통해 제작한다.
- Positioning 엔진을 JSON 생성 도구 바라보고, SI-2 스키마를 만족하는 한 교체 가능하도록 제작한다.
- 외부에서 생성된 좌표 JSON을 입력으로 받아도, FR-2.2~2.6 보정 파이프라인을 동일하게 적용할 수 있어야 한다.

FR-2.2 벽면 밀착 **(Wall Attachment)**
- 모든 가구의 BB 한 면을 방 벽에 밀착시킨다. 로직은 아래와 같이 설계한다.
    1. 각 object BB에 대해 X·Y·Z 축 방향 4개 벽면까지의 거리를 계산한다.
    2. 가장 가까운 벽면을 선택한다.
    3. BB 해당 면이 선택된 벽면과 접촉하도록 position을 조정한다.
    4. 모든 object에 적용한다.

FR-2.3 Collision 검출 및 위치 재조정
- 두 가구 BB의 부피적 겹침을 해소한다. 부피적 겹침에 대한 정의는 아래와 같다.
    1. Collision: 두 BB AABB의 내부 부피 overlap > 0
    2. Edge Contact: 면만 접촉(overlap = 0) → 허용
- Collision 발생 시 위치 재조정을 진행해야 한다.
- 재조정 후에도 FR-2.2(벽면 밀착)를 유지해야 한다.

FR-2.4 Collision 반복 제한
- FR-2.3에 따른 Collision 해소 시 발생할 수 있는 무한 루프를 방지한다.
- object당 또는 전역 Collision 해소 시도를 최대 5회로 제한하며, 5회 초과 시 배치 가능한 object만 SI-2에 포함하고, 미배치 항목에 대한 오류 메시지를 출력한다.

FR-2.5 방 크기 대비 과대 가구 처리
- 주어진 방의 크기가 너무 작아 가구를 배치할 수 없는 경우를 처리한다.
- object BB가 room_size 내에 들어오지 않으면 FR-2.3 재배치 로직을 수행하며 재배치 5회 후에도 해당 object를 배치할 수 없으면 오류 메시지 출력 후 프로세스 진행을 멈추고 Input Module로 돌아가 다시 사용자 입력을 받을 수 있도록 한다.

FR-2.6 가구 수 확정 및 미배치 로그
- Positioning Module이 최종 furniture_number와 objects[]를 확정한다.
- furniture_number = Positioning 목표 가구 수 이며, objects.length = 실제 가구 수 이다.
- furniture_number - objects.length > 0 이면 미배치 로그 출력를 출력한다.

FR-2.7 SI-2 스키마 필드 생성
- SI-2 필수 필드를 채운다. 규칙은 아래와 같다.

필드	규칙
objects[].id	obj_1, obj_2, …
objects[].name	3D Modeling 프롬프트용 설명명
objects[].category	가구 유형 (FR-3.4 Trimesh 검증용)
position	BB 중심 (E-2), Unity 좌표계 (A-4)
rotation	Euler degree, Y-up
scale	BB 크기 (m)
bounding_box.size	scale과 동일

FR-2.8 SI-2 JSON 파일 저장
- SI-1 필드를 유지한 채 FR-2.7에 따라 생성된 스키마 필드 들을 추가·갱신하여 저장한다.

3.2.3 FR-3 3D Modeling Module

*3D Modeling Module*은 *SI-2*를 읽어 *Mesh*를 생성하고 *SI-3*를 저장한다*.1*차 엔진은 *Tripo API*이며*, FR-2*와 같이 교체 가능하도록 설계한다.

FR-3.1 3D Mesh 생성 프롬프트 작성
- SI-2에 적혀 있는 atmosphere와 objects.name, objects.category를 바탕으로 LLM으로 object별 Text-to-3D 프롬프트를 작성한다.
- 작성된 프롬프트는 모든 가구들에 대해서 atmosphere에 따라 일관성을 보장 돼야 함을 명시한다.

FR-3.2 Mesh AI 호출 및 병렬 생성
- 3D Mesh AI API를 통해 object별 Mesh를 생성한다.
- 1차적인 구현은 Tripo API를 통해 진행하며, SI-3 계약 유지 시 교체 가능하다.
- object별 요청을 순차가 아닌 병렬로 제출하여 총 소요 시간을 줄인다.
- 출력 형식: GLB 우선 (Must). FBX 경로·import 설계상 지원 (D-2, Should).
- 저장: {GENROOM_DATA_DIR}/meshes/{id}_{name}.glb

FR-3.3 Mesh 생성 실패 시 재요청
- Mesh 생성 실패 시 1회 재요청 후 재실패시에는 placeholder(단순 큐브 Mesh)로 대체한다.
- 재실패의 경우에는 사용자에게 오류메시지를 통해 placeholder로 대체되었음을 알려야한다.

FR-3.4 Mesh 방향 검증
- 생성된 3D 모델의 방향이 카테고리별로 정의된 방향과 일치하도록 모델의 방향 값 수정 기능을 수행해야 한다.
- python Trimesh 라이브러리를 통해 구현하며, category별 자세 정의는 아래와 같다.

category	올바른 자세 **(rotation=0** 기준**)**
L자형 가구(의자, 소파 등)	좌판이 +Y(위)를 향함
상판형 가구(책상, 테이블, 침대)	매트리스 상면이 +Y를 향함
박스형 (옷장, 수납장 등)	밀도가 가장 높은 평면을 찾아 그걸 바닥과 뒷면으로 확정
generic	Mesh up-axis가 +Y 와 ±15° 이내

FR-3.5 SI-3 JSON 파일 저장
- SI-2를 유지하고 mesh_file 필드를 추가하여 저장한다.

FR-3.6 3D 생성 진행 **UI** 갱신
- Generation Stage에서 진행 상황을 표시한다.

3.2.4 FR-4 Generation Module

*Generation Module*은 *SI-3*와 *Mesh* 파일을 읽어 *Unity*에 방을 구성한다. *Modeling* 완료 시 *Unity*를 자동 실행하고 씬에 결과를 표시한다.

FR-4.1 Modeling 완료 후 **Unity** 자동 실행

- 3D Modeling Module 완료 시 Unity 실행 및 씬 로드를 시작하며 구체적인 순서는 아래와 같다.
    1. 웹 UI(UI-2.5)에「Modeling module에서 생성이 완료되면 씬을 실행합니다」(또는 동등 문구) 출력.
    2. Unity Engine을 자동 실행한다.
    3. {GENROOM_DATA_DIR}/genroom_si3.json 및 meshes/ 를 읽어 FR-4.1~4.4 수행.

FR-4.2 SI-3 로드 및 씬 구성
- SI-3를 파싱하고 Unity 씬 구성을 시작한다.

FR-4.3 방 셸 생성
- room_size 기준으로 방 구조물을 생성한다. 바닥 + 4면 벽 생성하며, 천장은 생성하지 않는다

FR-4.4 가구 **Mesh** 배치
- SI-3 각 object의 Mesh를 씬에 배치한다.

FR-4.5 Mesh import 재시도 및 **placeholder**
- 최종 배치 단계에서 시스템적으로 오류가 발생할 경우 재시도해야 한다.
- import 실패 시 최대 3회 재시도하며, 3회 실패 시 cube placeholder로 대체한다.

FR-4.6 최종 **3D** 방 표시
- 사용자에게 완성된 3D 방을 표시한다.
- Unity Scene View 또는 Game View에서 방 셸·가구가 즉시 확인 가능해야 한다.
- 카메라·조명은 기본 씬 설정을 사용한다.

3.2.5 FR Traceability Summary

PF (2장)	주요 **FR**	**SI**
PF-1 자연어 입력	FR-1.1, 1.2	—
PF-2 JSON 생성	FR-1.3~1.6	SI-1
PF-3 좌표 생성	FR-2.1, 2.7~2.8	SI-2
PF-4 배치 보정	FR-2.2~2.6	SI-2
PF-5 Mesh 생성	FR-3.1~3.2	SI-3
PF-6 방향 검증	FR-3.4	SI-3
PF-7 Default/placeholder	FR-3.3, 4.4	SI-3
PF-8 Unity 출력	FR-4.0~4.6	SI-3
PF-9 진행 UI	FR-1.7, 3.6	UI
PF-11 오류·재입력	FR-1.2, 1.4, 2.4~2.6	UI / log

3.3 Performance requirements

본 절은 측정 가능한 성능 목표를 정의한다.

3.3.1 Module Performance requirements

ID	구간	요구사항	관련 **FR**	측정 시작~종료
PR-1	Input Module (LLM)	SI-1 생성 ≤ 30초	FR-1.3~1.6	프롬프트 제출 → genroom_si1.json 저장
PR-2	Positioning Module	SI-2 생성 ≤ 5분	FR-2.1~2.8	SI-1 로드 → genroom_si2.json 저장 (Collision 반복 FR-2.4 포함)
PR-3	3D Modeling (object당)	Tripo API 1 object ≤ 3분	FR-3.2	API task 생성 → GLB 저장 완료
PR-4	3D Modeling (전체)	병렬 처리 시 전 object set ≤ 5분	FR-3.2	첫 API 병렬 요청 → 마지막 object SI-3 반영
PR-5	Trimesh 방향 검증	object당 ≤ 5초	FR-3.4	GLB 로드 → rotation 보정값 확정
PR-6	Generation Module	Unity 자동 실행 + 씬 배치 ≤ 1분	FR-4.0~4.6	SI-3 로드 → Scene/Game View 표시
PR-7	End-to-End	프롬프트 제출 → Unity 방 표시 ≤ 10분	전 FR	UI 제출 → FR-4.6 완료

3.3.2 UI·통신 성능

ID	요구사항	관련
PR-8	웹 UI 진행률 갱신 주기 ≤ 5초	CI-3.1, FR-3.6
PR-9	웹 UI Input Stage 첫 페인트 ≤ 2초 (로컬 Backend 기준)	UI-S1
PR-10	SI JSON 파일 read/write ≤ 500ms (object 20개 이하)	CI-1

3.4 Design constraints

DC-1 시스템 형태 및 실행 구조

ID	제약
DC-1.1	사용자 입력은 웹 UI(3.1.1 UI-S1/S2) 를 통해 수집한다 (FR-1.1).
DC-1.2	Input·Positioning·3D Modeling 오케스트레이션은 Backend(Python) 에서 수행한다.

DC-1.3	3D Modeling Module 완료 시 Unity Engine을 자동 실행하고 씬에 방을 표시한다 (FR-4.1).
DC-1.4	Backend·웹 UI·Unity 연동을 포함한 통합 실행 환경(런처 또는 동등 메커니즘)을 제공한다.

DC-2 모듈간 **Data** 전달 정의

ID	제약
DC-2.1	Module 간 정식 계약은 SI-1 → SI-2 → SI-3 JSON 파일이다 (3.1.3, CI-1).
DC-2.2	파일 경로: {GENROOM_DATA_DIR}/genroom_si1.json, genroom_si2.json, genroom_si3.json.
DC-2.3	Mesh 파일: {GENROOM_DATA_DIR}/meshes/.
DC-2.4	웹 UI ↔ Backend 내부 API 사용 가능. 단, Module 간 최종 산출물은 SI JSON 파일이 Single Source of Truth (CI-3).

DC-3 출력 엔진 정의

ID	제약
DC-3.1	최종 3D 방 출력은 Unity Engine만 사용한다 (FR-4).

DC-4 UI 환경 정의

ID	제약
DC-4.1	Input Module UI는 HTML 웹 기반 단일 페이지 (3.1.1).
DC-4.2	Unity EditorWindow는 Input UI 대체 수단이 아님. Generation·씬 표시용.
DC-4.3	UI는 한국어 기본 (3.1.1 UI-3.1).

DC-5 산출물 형식 정의

ID	제약
DC-5.1	배치 데이터: JSON (SI-1~3, UTF-8).
DC-5.2	3D Mesh: GLB Must. FBX import·경로 설계 Should (D-2, FR-3.2).
DC-5.3	Placeholder Mesh: GLB cube (FR-3.3, FR-4.4).

DC-6 모듈 책임 및 오류 격리

ID	제약
DC-6.1	Input / Positioning / 3D Modeling / Generation 책임 분리 (1.2 Scope).
DC-6.2	한 Module 오류가 전체 Crash로 이어지지 않도록 Module별 예외 처리 (QA-8).
DC-6.3	Module 실패 시 이전 단계 SI JSON 파일 유지 (CI-1.5).

DC-7 구현 언어 정의

ID	제약
DC-7.1	Input Module (LLM 연동), Positioning Module, 3D Modeling Module (API·Trimesh): Python.
DC-7.2	Generation Module, Unity 연동: C#.
DC-7.3	Input Module 웹 UI: HTML / CSS / JavaScript.

DC-8 교체 가능 설계

ID	제약
DC-8.1	Positioning 엔진(ReSpace 등)은 SI-2 생성 도구. SI-2 스키마 유지 시 교체 가능 (FR-2.1, C-1).
DC-8.2	3D Mesh AI(Tripo API 등)는 SI-3·mesh_file 계약 유지 시 교체 가능 (FR-3.2, D-1).
DC-8.3	Input LLM(Gemini API 등)은 SI-1 스키마 유지 시 교체 가능 (FR-1.3, QA-7).
DC-8.4	엔진 교체 시 Generation Module까지 SI-3 호환 (QA-7).

DC-9 Mesh 생성 및 검증

ID	제약
DC-9.1	Mesh API 요청은 object별 병렬 제출 (FR-3.2).
DC-9.2	스케일 자동 검증·수정 로직 구현 금지 (D-3).
DC-9.3	Mesh 방향 검증은 Python Trimesh + category별 자세 규칙 (FR-3.4).

DC-10 방·배치 규칙

ID	제약
DC-10.1	room_size 미명시 시 6 × 6 × 6 m (FR-1.3, B-2).
DC-10.2	furniture_number는 Positioning Module이 SI-2에 기록 (FR-2.6).
DC-10.3	object position = BB 중심 (FR-2.7, E-2).
DC-10.4	Unity 방 셸: 바닥 + 4벽, 천장 없음 (FR-4.2, E-1).

3.5 Software system attributes

3.5.1 보안성(Security)

QA-1 API 보안

- 외부 API(Gemini API 등) 통신 시 API Key 및 인증 토큰은 안전하게 관리되어야 하며 노출되지 않도록 처리해야 한다.

QA-2 이상 입력 처리

- 시스템이 처리할 수 없는 입력이 들어오거나 개인정보에 접근하는 입력이 들어오면 시스템 내에서 에러로 처리해야 한다.

3.5.2 가용성(Availability)

QA-3 결함 허용**(Fault Tolerance)**
- 3D 모델 생성 실패나 특정 레이어에서 오류가 발생하더라도 전체 시스템이 중단(Crash)되지 않고, 기본(Default) 모델 대체 등의 예외 처리를 통해 서비스를 지속해야 한 기본(Default) 모델 대체 등의 예외 처리를 통해 서비스를 지속해야 한다.

3.5.3 사용성(Usability)

QA-4 직관적 인터페이스
- 단순 자연어 입력창만을 UI로 제작하여 전문적인 지식이 없는 일반 사용자도 방 배치 결과물을 얻을 수 있도록 간결하게 설계해야 한다.

QA-5 피드백 제공
- AI 모델의 생성 과정(이미지 생성, 배치 최적화 등) 동안 사용자가 대기 상태를 인지할 수 있도록 로딩 바나 안내 메시지를 제공해야 한다.

3.5.4 신뢰성(Reliability)

QA-6 출력 데이터 정합성
- UI에서 입력한 카테고리가 최종 3D 씬과 일치해야 하며, Bounding Box와 모델의 스케일이 일관성을 유지해야 한다.

QA-7 API 응답 정합성
- 비슷한 기능을 하는 API에 대해서 바꿔 사용하였을때 출력되는 값은 일정하게 나오도록 한다.

QA-8 예외 처리
- 각 주요 Module는 발생할 수 있는 error들에 대해 독립적인 예외 처리 로직을 갖추어야 한다.

QA-9 외부 **API** 전면 장애 시 동작
- 개별 object 실패(QA-3)와 구분하여, 외부 API 서비스 전체 장애 시 진행 중단 및 명확한 오류를 보장한다.
- 각 단계별로 사용하는 API 및 엔진이 장애 상태일시 파이프라인 진행을 중단하고 오류메시지를 출력해야한다.

3.5.5 유지보수성 (Maintainability)

QA-10 Pluggable Engine 문서화
- Positioning·3D Mesh·Input LLM 교체 시 SI 스키마·환경 변수·API endpoint 변경점 문서화해야한다.

3.5.6 운영성 (Operability)

QA-11 통합 런처 및 **Unity** 자동 실행

- 최종 사용자가 단일 런처 1회 실행으로 전체 파이프라인 환경을 구동할 수 있어야 한다.
- 런처 실행 후 사용자는 브라우저 URL 수동 입력·Backend 수동 실행·venv 수동 활성화 없이 웹 UI에 접속 가능해야 한다.
- Generation 트리거 시 항상 새 Unity 프로세스를 실행한다. (기존 Unity 재사용 금지)
- 런처·Backend·웹 UI 기동 실패 시 Crash 대신 오류 메시지 + 조치 안내를 해야한다. (QA-8)