

AR 기술을 활용한 소셜 네트워킹 서비스 진행 현황

meARy

201924447 김진영

201924557 임석윤

201924613 허취원

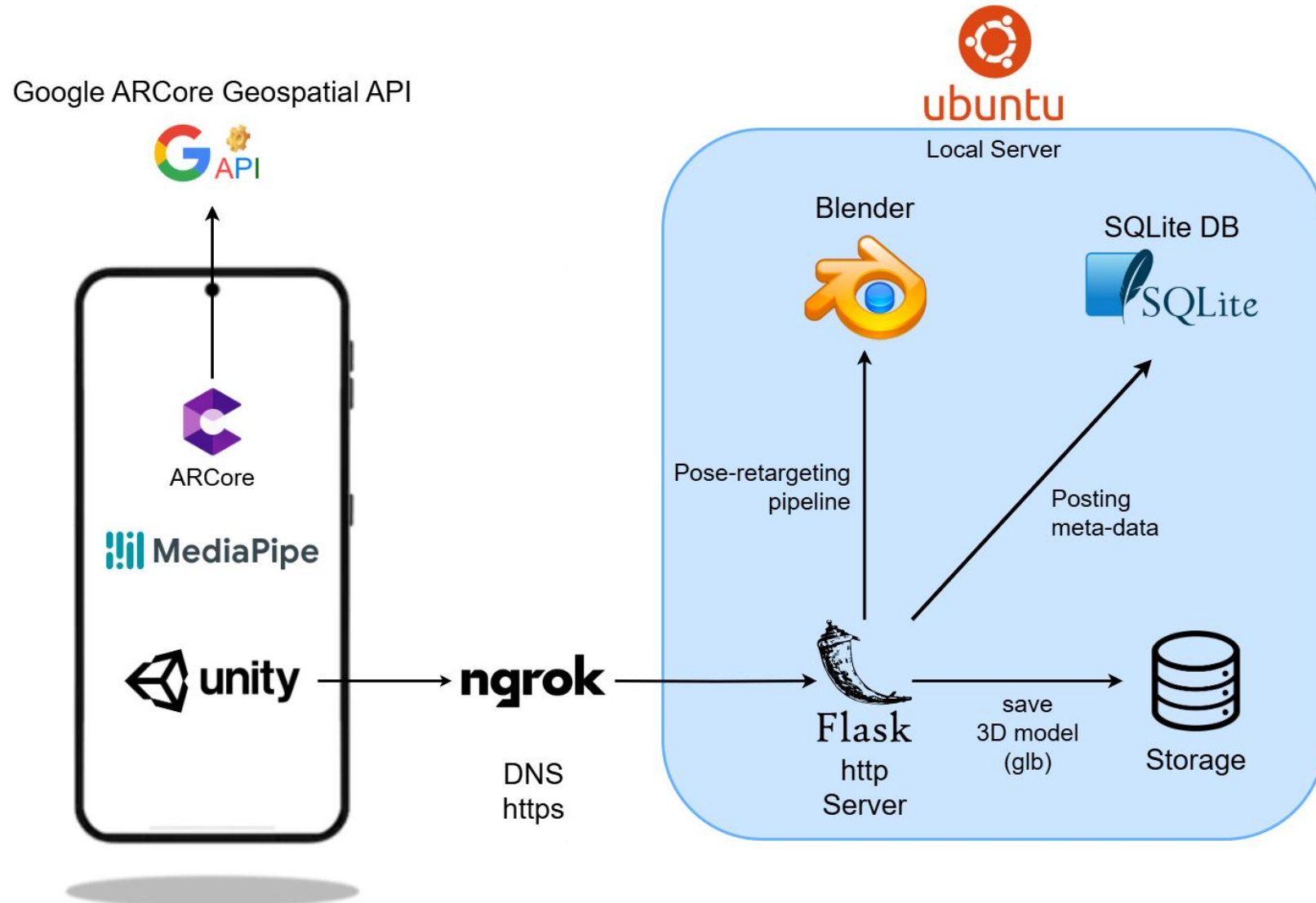
1. 프로젝트 목표

- AR 기능을 활용한 SNS 서비스를 통해 사용자가 현실 공간 위에 디지털 콘텐츠를 올리고, 이를 다른 사용자와 공유 및 체험하면서 새로운 방식의 SNS 서비스를 제공.

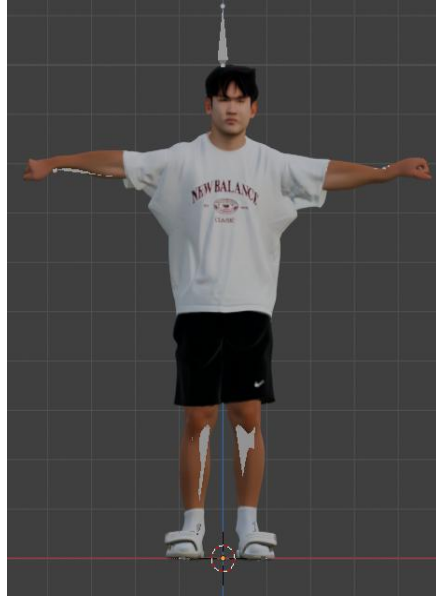
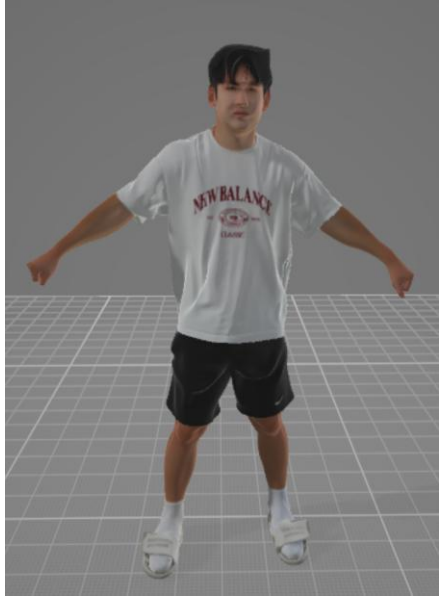
2. 기능별 구현 분류

- ① 사용자의 2D 사진을 사용한 3D 모델링 및 Rigging.
- ② Mediapipe 기술을 활용한 사진 기반 자세 추정 및 적용.
- ③ Unity ARFoundation, Google ARCore 기술을 활용한 AR 어플리케이션.
- ④ Flask, SQLite, Blender 를 활용한 서버.

3. 서비스 아키텍처



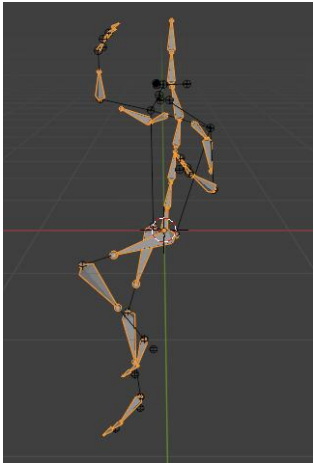
• 진행 상황



3D 모델링 파트

- ① 사람의 정면모습 촬영
- ② Tripo3D를 이용하여 fbx 형식으로 3D 모델링
- ③ Mixamo를 이용하여 auto-rigging
- ④ 위 과정들은 현재로서는 자동화가 불가능하여 초기에 한번만 진행

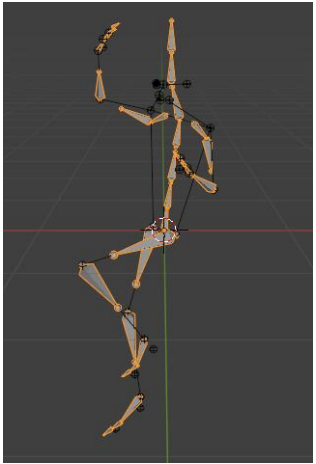
• 진행 상황



Blender Scene 구성

- ① MediaPipe를 이용한 사진 속 인물의 pose_landmarks.json 추출
- ② MediaPipe -> BlazePose (33개의 랜드마크만 제공)
- ③ Mixamo -> HumanIK(Inverse Kinematics) 좌표, 회전, 계층 기반
- ④ 또한 MediaPipe와 Blender의 서로 다른 좌표계를 맞추고, 사진의 크기에 맞게 3D 모델을 auto-scaling 합니다

• 진행 상황

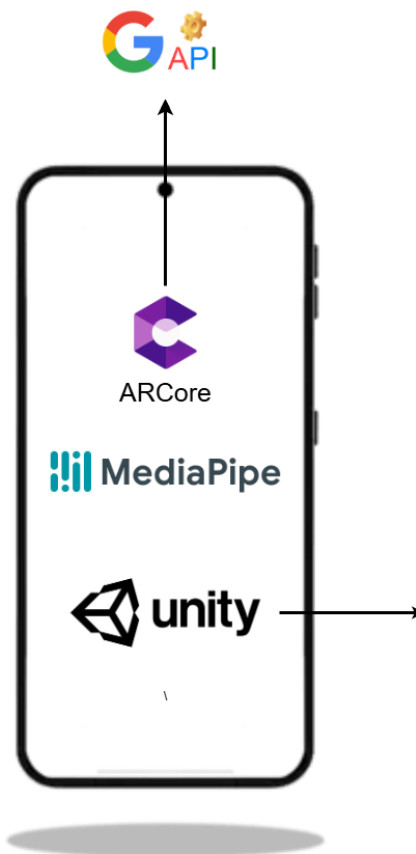


Pose-retargeting 파트

- ① Build_targets 함수를 통해 MediaPipe 랜드마크를 기반으로 뼈대의 목표 지점생성
- ② 이후 apply_constraints 함수로 bone 이동후 pose를 bake한다
- ③ Unity에서 모델 사용을 편리하게 하기 위해서 원점을 두발의 중심으로 이동한다.
- ④ 마지막으로 fbx였던 파일 형식을 armature와 mesh가 포함된 glb(glTF 2.0) 형식으로 변환하여 저장한다.
- ⑤ 모든 과정을 자동화하였으며 오류가 없을시 3-5초 내에 리타게팅 가능

• 진행 상황

Google ARCore Geospatial API

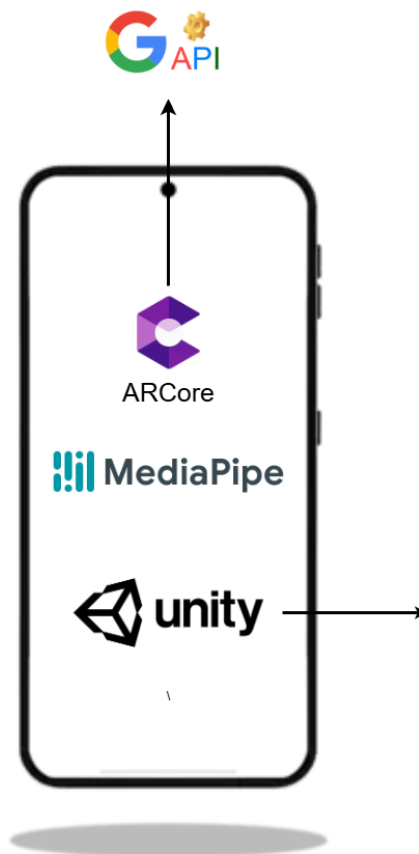


기능 1 구현 흐름 (SNS 포스팅 생성 및 서버 전송)

- ① 사진 촬영.
- ② Mediapipe(C# plugin)을 사용하여 pose detection.
- ③ 이전 결과를 토대로 화면상의 사람의 발 위치 얻음 (화면 픽셀 위치).
- ④ 화면 픽셀 기반으로 ARRaycastManager를 활용하여 AR world position 생성.
- ⑤ AREarthManager를 활용하여 Geospatial position으로 변환.
- ⑥ UnityWebRequest를 활용하여 Flask 서버로 사진(Texture2D), Geospatial position 전송

• 진행 상황

Google ARCore Geospatial API

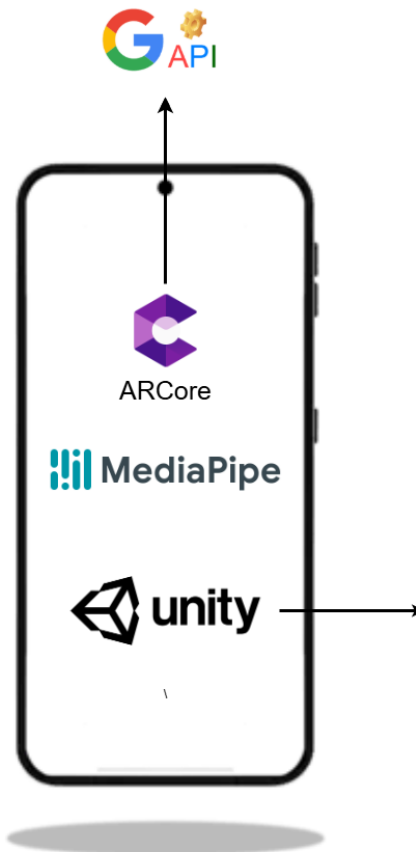


기능 1 구현 디테일(SNS 포스팅 생성 및 서버 전송)

- ① 사진 촬영 이후 사진의 pose detection 수행에 앞서 ARRaycastManager의 Raycast 기능이 불안정.
✓ ARPlaneDetector를 통해 사진 촬영 이전에, 사용자가 팝업창으로 Raycast 기능을 확인.
- ② Google API의 불안정성으로 인한 AR world position -> Geospatial position 변환 실패.
- ③ 네트워크 연결 이슈로 인한 서버로의 포스팅 정보 전송 실패.
- ④ AR 공간 식별 실패로 인한 Raycast를 사용한 AR word position 획득 실패.
- ⑤ 사진에 사람이 없어서 발생하는 pose detection 실패.
✓ 위와 같은 런타임에 발생할 수 있는 여러 예외를 UIManager를 통해 팝업창으로 확인 가능.

• 진행 상황

Google ARCore Geospatial API

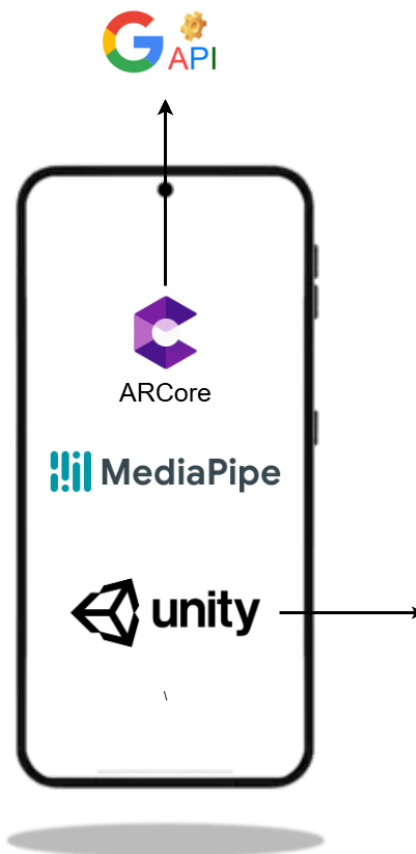


기능 2 구현 흐름 (위치 기반 AR 포스팅 열람)

- ① AREarthManager를 통해 사용자의 Geospatial position 생성.
- ② Geospatial position, UnityWebRequest를 사용하여 Flask 서버로 주변 포스팅 정보 요청.
- ③ 서버로부터 받은 Response (posting id, Geospatial position)을 바탕으로 3D 모델 파일 (glb) 요청.
- ④ glb 파일을 토대로 GameObject 프리팹 생성
- ⑤ GameObject 프리팹, Geospatial position을 사용하여 ARAnchor 생성 및 3D 모델 AR world에 표현.

• 진행 상황

Google ARCore Geospatial API



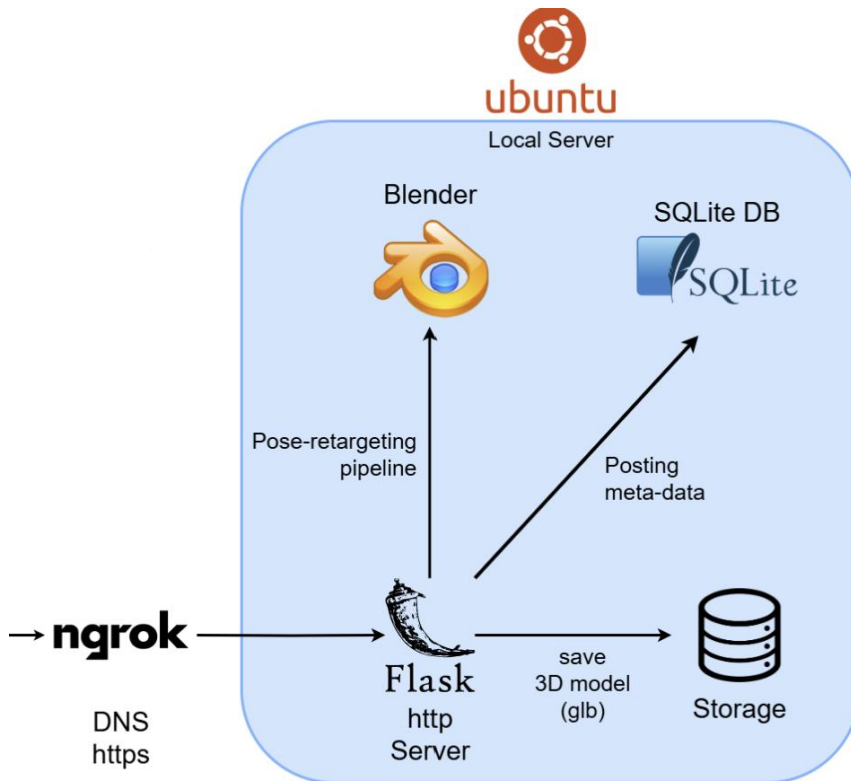
기능 2 구현 디테일 (위치 기반 AR 포스팅 열람)

- ① Geospatial position, glb 파일을 활용해 3D 모델을 AR world에 표현할 때, altitude의 부정확성으로 인해 생성된 모델이 공중에 떠있을 수 있음.
 - ✓ Terrain을 활용하여 위도 경도 좌표를 가지고 정확한 고도 정보 추론 및 Anchor 생성
 - ✓ Terrain 기능이 여러 이슈 (실내, 고도 정보 부족)로 인해 고도 정보를 추론하지 못할 경우 Plane detection을 통해 고도 정보를 추론 및 Anchor 생성

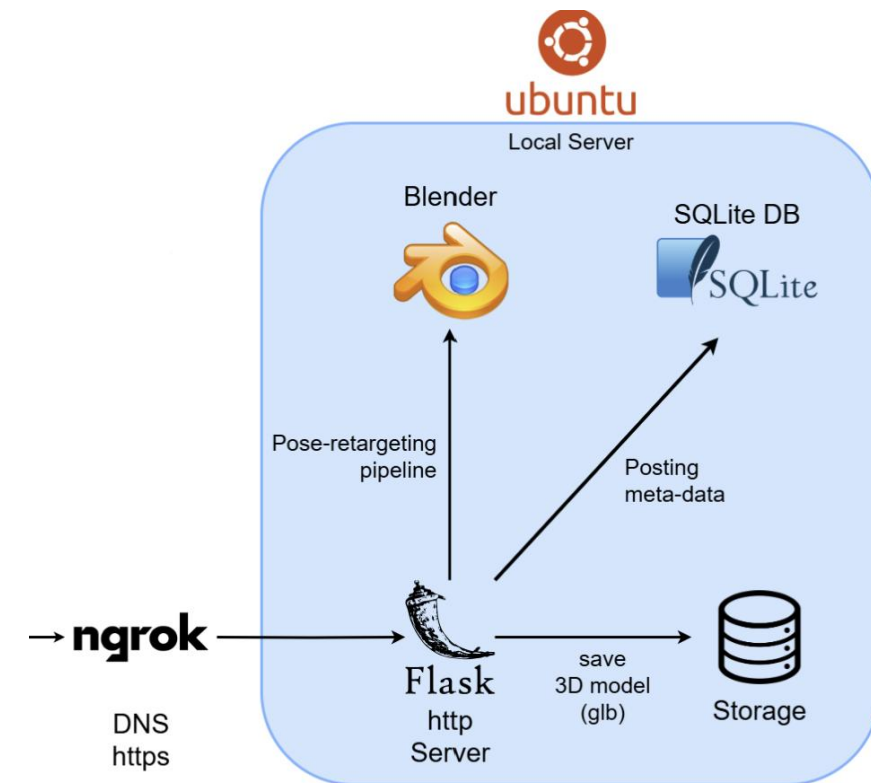
• 진행 상황

서버 아키텍처

- ngrok HTTPS static domain
- Ubuntu 22.04 Local Server
 - Python Flask REST API Gateway
 - Blender Pose-Retargeting pipeline
 - SQLite DB
 - Local Storage



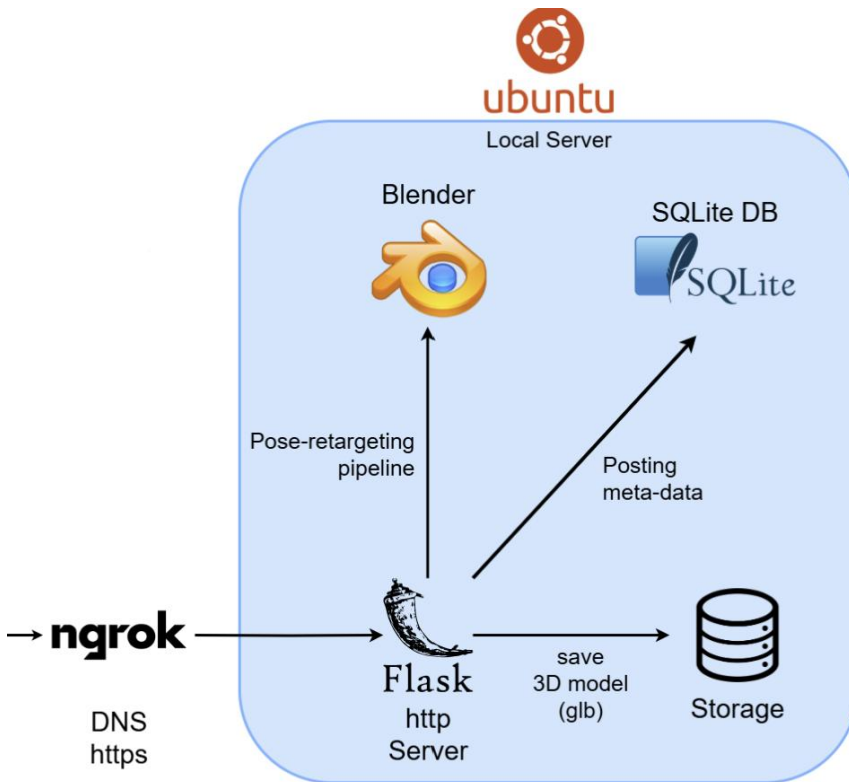
• 진행 상황



서버 기능 (포스팅 업로드 request)

- ① 클라이언트가 이미지와 지리 정보 업로드
- ② 이미지 속 인물 자세 자동 인식
- ③ Blender 파이프라인으로 3D 모델 리타게팅
- ④ .glb 파일 생성
- ⑤ 이미지 & 3D 모델 스토리지 저장
- ⑥ 지리 정보 & 파일 메타데이터 DB 저장

• 진행 상황



서버 기능 (포스팅 다운로드 request)

- ① 클라이언트 요청 (지리 정보 기준)
- ② DB에서 반경 100m 내 포스팅 조회
- ③ 위치 정보, 이미지, 3D 모델 파일 반환
- ④ 클라이언트에서 위치 기반 AR 경험 제공