



SI-Follow

자체 sLLM 개발 통한 기업 업무 활용 생성형 AI 플랫폼

TEAM 3조 원형과형들

한병찬, 이재호, 임영훈, 정원형, 허우영

[멘토] 최근호

목 차

- 01 프로젝트 개요
- 02 프로젝트 팀 구성 및 역할
- 03 프로젝트 수행 절차 및 방법
- 04 프로젝트 수행 경과
- 05 시연
- 06 추후 계획
- 07 Q&A

01 프로젝트 개요

1

프로젝트 주제
및 선정 배경,
기획의도

2

프로젝트
내용

3

프로젝트
구조
및
기술스택

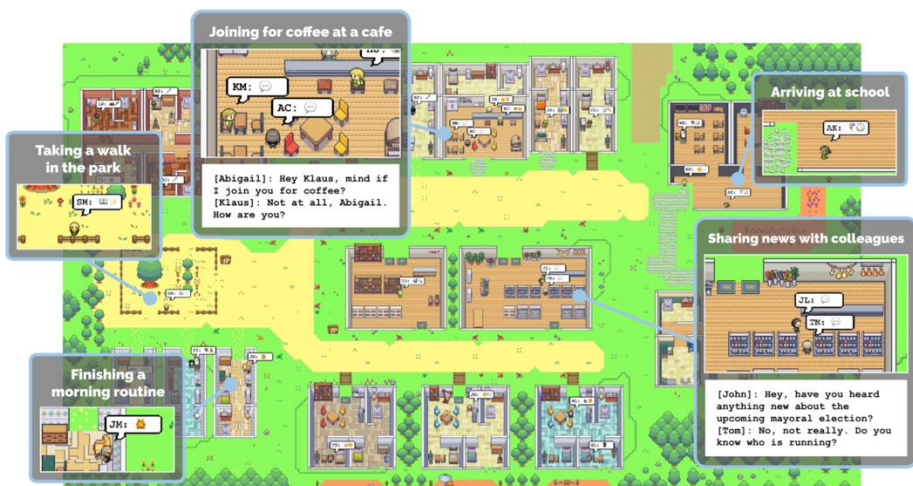
4

활용방안 및
기대 효과

01

프로젝트 개요(프로젝트 주제 및 선정 배경)

Generative Agents



LLM agent들에게 persona를 부여하여 서로 상호 작용하게 만들었더니 인간 행동을 유사하게 재현



SI 기업 실무자의 역할들을 LLM agent들에게 주 입하면 코딩을 더욱 더 잘할 수 있지 않을까?

01

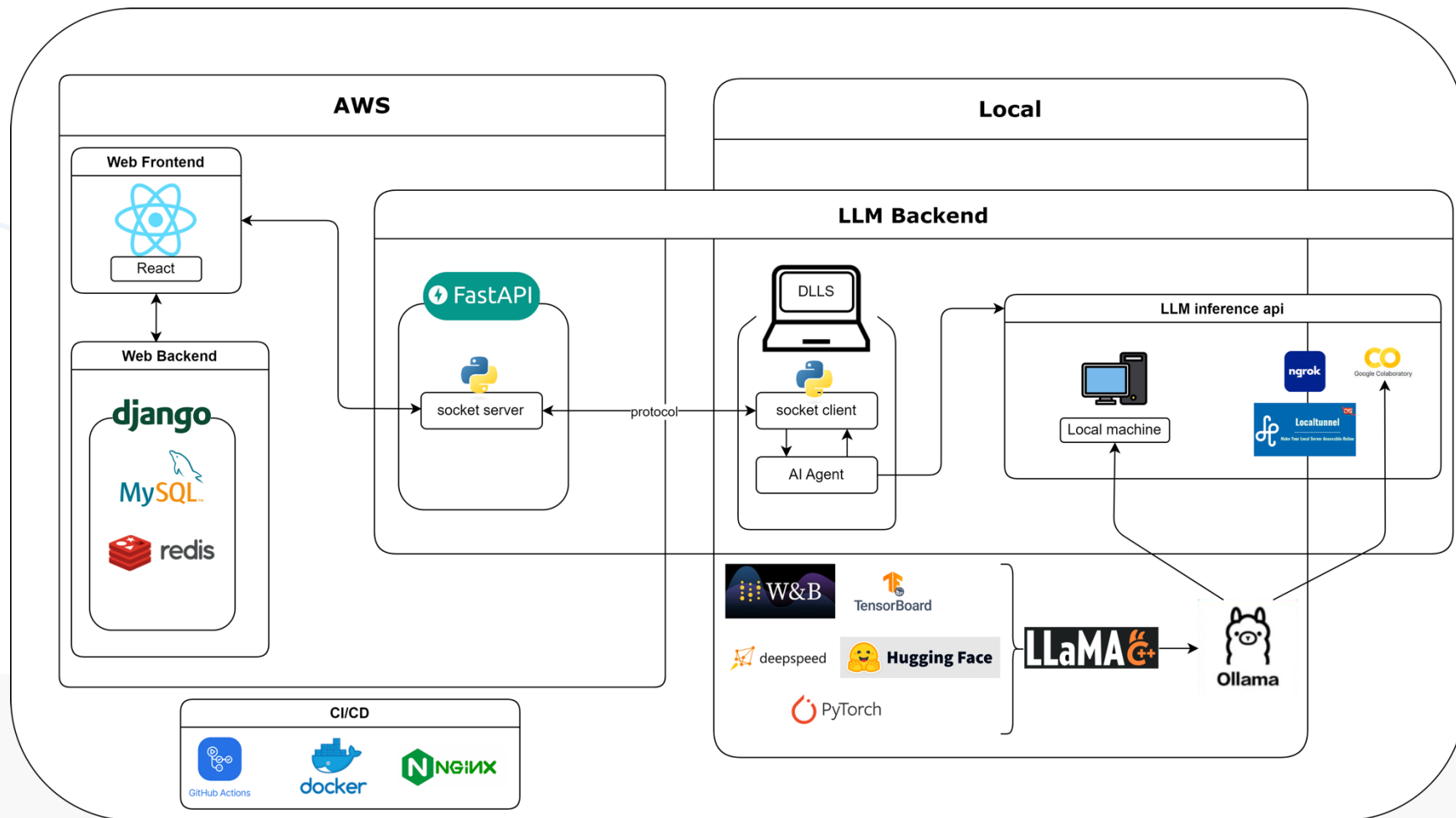
프로젝트 개요(프로젝트 내용)



- Chief Executive Officer
- Chief Technology Officer
- Programmer
- Code Reviewer
- Software Test Engineer

01

프로젝트 개요(프로젝트 구조 및 기술스택)



01











프로젝트 개요(활용방안 및 기대 효과)

1. 빠른 프로토타입 개발: 새로운 아이디어나 기능을 빠르게 구현하여 데모를 만들 수 있음
2. 자동화된 도구를 통해 초기 설정 작업에 소요되는 인력을 줄일 수 있음
3. 증분형 개발: 추가 기능들을 Agent가 빠르게 작성하고 테스트까지 가능



서비스 기업의 경우 **빠르게** 소비자들의 요구사항을 충족시킬 수 있음

02 프로젝트 팀 구성 및 역할

이름	역할	담당 업무
한병찬	팀장	 Agent tuning  Backend for LLM  LLM modeling  Dataset preprocessing
이재호	팀원	 Agent tuning  Prompt engineering
임영훈	팀원	 Web Frontend
정원형	팀원	 Web Backend  CI/CD for frontend
허우영	팀원	 CI/CD for backend
이상훈	PM	
최근호	멘토	

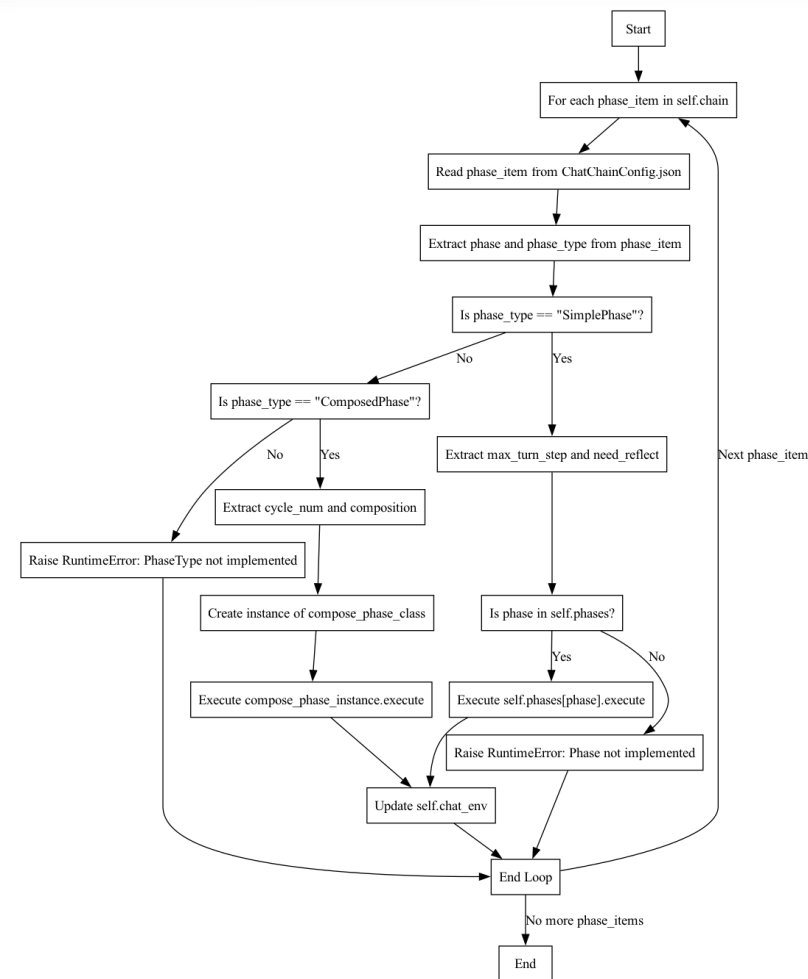
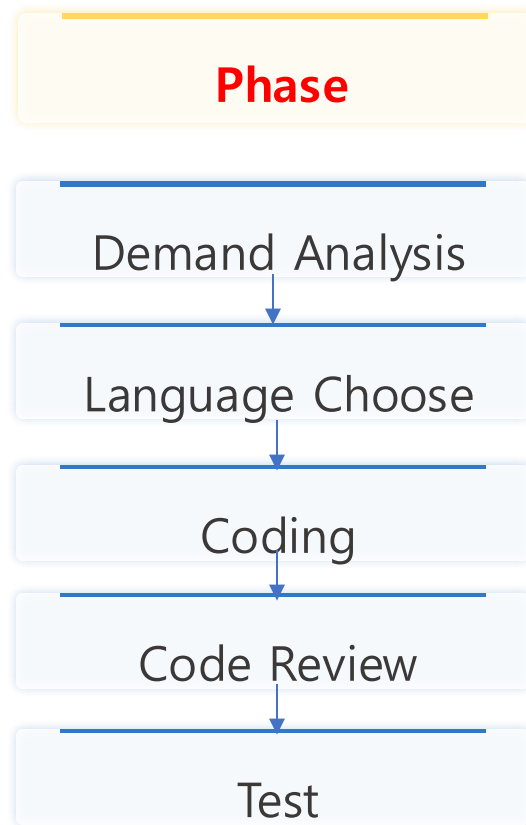
03 프로젝트 수행 절차 및 방법

구분	기간	활동	비고
기획	9/5(목) ~ 9/13(금)	 프로젝트 기획 및 주제 선정  관련논문 research	
Agent 코드 분석	9/14(토) ~ 9/26(목)	 Agent 코드분석  Agent 튜닝 방향 설정	
Agent 튜닝	9/27(금) ~	 기획을 바탕으로 튜닝  Unittest phase 추가	
LLM Backend 구축	9/30(월) ~	 AWS 서버 구축  Local PC 서버 구축  LLM API 서버 구성	
Frontend 구축	9/25(수) ~	 로그인 구성  Agent 구동 시각화  Agent 결과 시각화	
Backend 구축	9/25(수) ~	 로그인 구성	
CI/CD	10/1(화) ~	 Frontend CI/CD 구성  Backend CI/CD 구성	
LLM tuning	10/9(수) ~	 Dataset 전처리  SFT  Quantization	

04 프로젝트 수행 경과

▶ Agent 코드 분석

Agent baseline



04 프로젝트 수행 경과

▶ Agent 코드 튜닝 방향

기존 방식은 생성된 코드를 main.py를 실행하여 에러 파악

Unittest Phase

- 1.코드의 품질을 보장
- 2.버그를 조기에 발견할 수 있도록 수정

LLM과 사용자가 상호작용 하는 부분이 없어 요구사항을 정확히 파악하지 못하는 부분 존재

HumanInteraction Phase

- 1.코드 생성 의뢰 단계에서 agent가 사용자와의 대화를 통해 요구사항을 확실히 전달(RAG 사용)
- 2.Agent에게 추가적인 요구사항을 개발 진행 도중 전달 가능

04 프로젝트 수행 경과

▶ Agent 코드 튜닝 방향

개발해야 하는 기능을 놓치는 경우 발생



Backlog Phase

1. 개발해야 하는 기능들을 먼저 정의하여 빼먹는 기능이 없도록 함
2. 어떤 기능을 구현했는지 사용자가 파악하기 쉬움

작성 완료된 코드들을 다시 확인하기 불편함

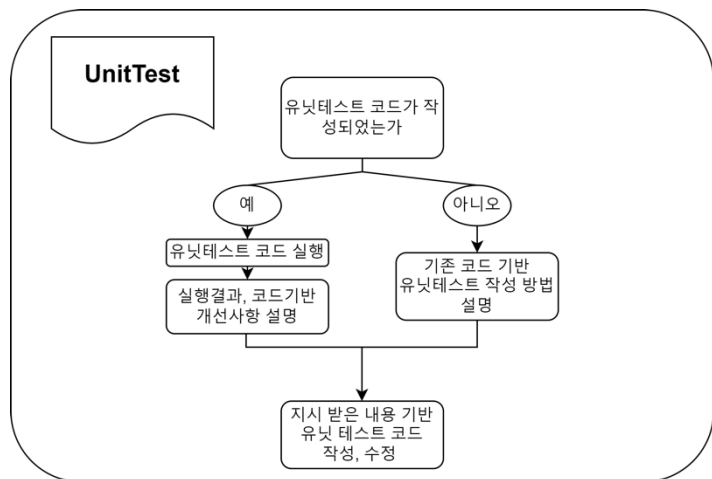


Github Phase

Github에 작성 완료된 코드들을 커밋하여 사용자가 언제든지 코드 확인 및 수정에 용이하도록 함

04 프로젝트 수행 경과

▶ Unittest phase

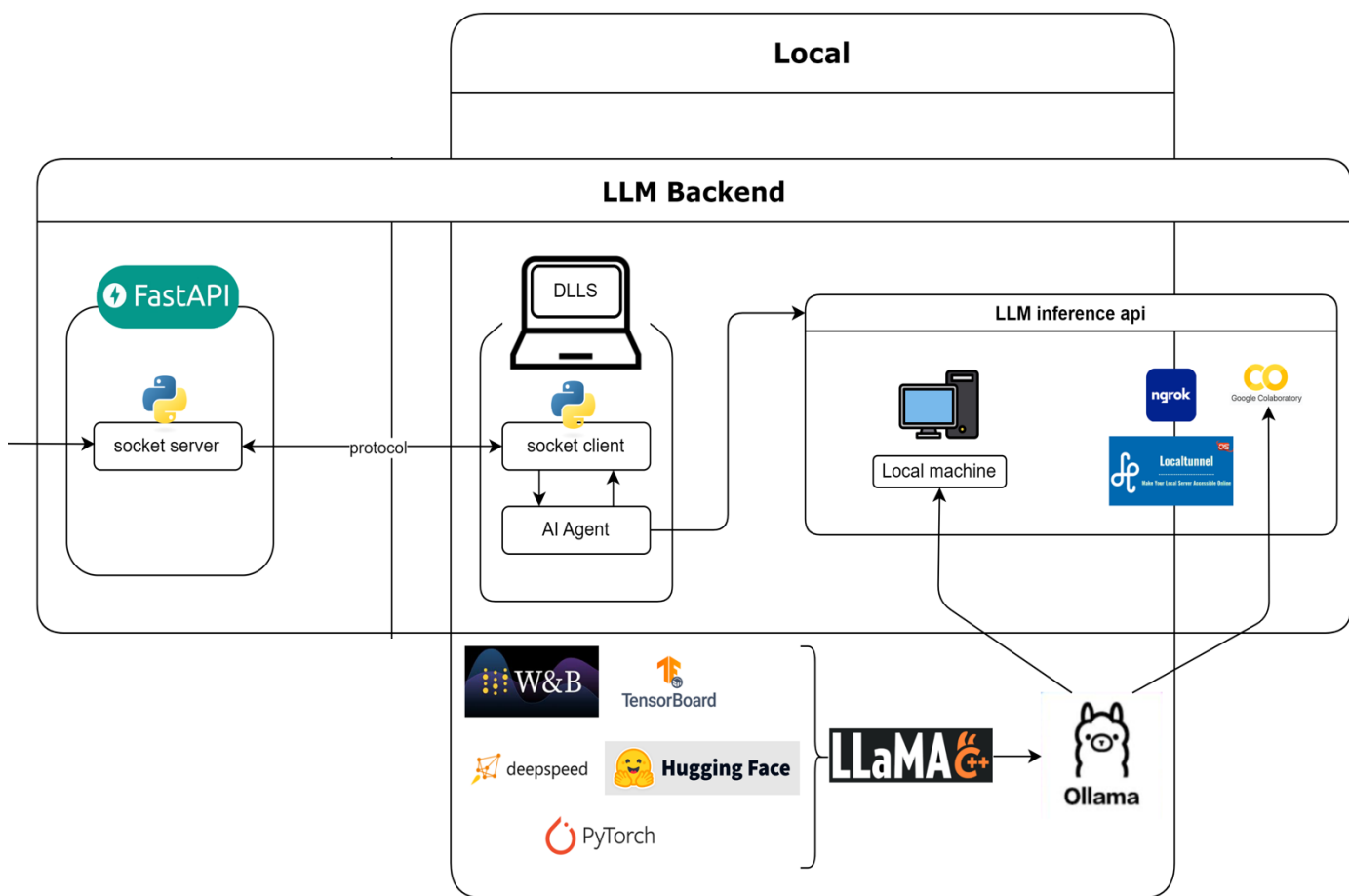


"If no Unit test Codes and Unit test reports are provided, analyze the given source code to identify all public functions and classes. Provide a d
"If Unit test Codes are provided, review the reports and identify potential issues or missing coverage areas. Suggest modifications to improve the
"When suggesting improvements or modifications for Unit test code, follow these rules:",
"1. Each Unit test file name should ****start with 'unittest_**** (e.g., 'unittest_example.py'). Additionally, provide a suggested file name for the u
"2. Base your suggestions **on** the provided source code. Ensure that each public function and class is adequately tested.",
"3. Use meaningful and consistent naming conventions to make it easy to understand what functionality is being tested."

"1. You will start with the \"unittest_main\" file, then go to the \"Source Code\" files that are imported by that file, and so on.",
"2. Please note that the code should be fully functional. Ensure to implement all functions. No placeholders (such as 'pass' in Python).",
"3. When writing or modifying Unit Test code, file names **MUST START WITH \"unittest_\"**.",
"4. ****Important:**** Only one file should contain the \"if __name__ == '__main__':\" block. This block must be included ****only**** in \"unittest_main.py\".",
"5. Do not use 'if __name__ == '__main__':' blocks in files other than unittest_main.py. The file that includes this block is recognized as the unittes
"6. Please refer to the provided source code, Unit Test code and write the unit test code so that there is only one entry point.",
"As the {assistant_role}, to satisfy the new user's demand and make the software execute smoothly and robustly, you should modify the unit test codes

04 프로젝트 수행 경과

▶ LLM Backend 구축



Basemodel
∞ meta-llama/Llama-3.2-3B

04 프로젝트 수행 경과

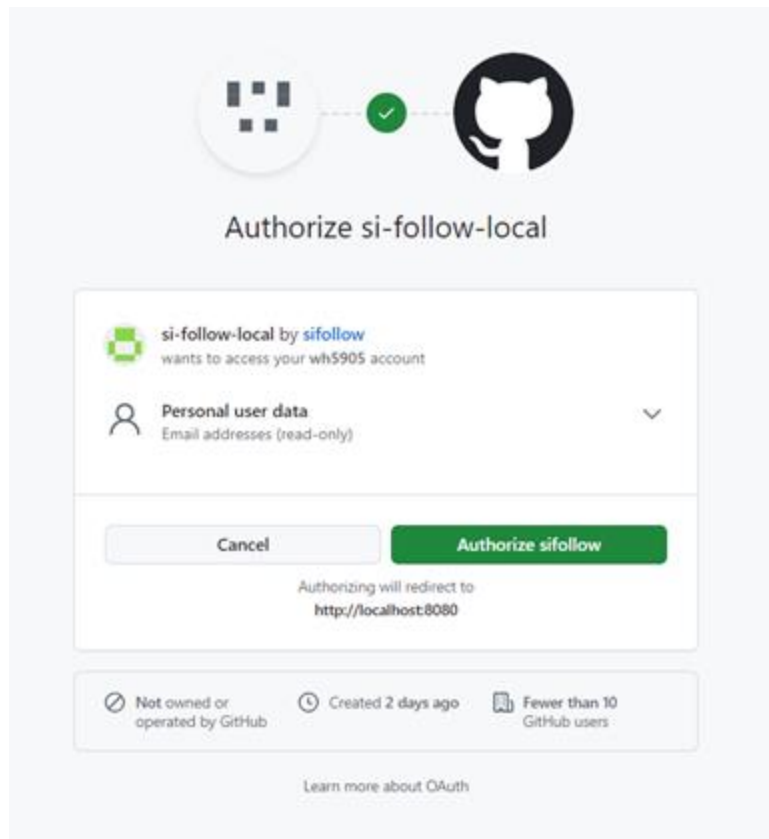
▶ Front, Backend 구축



04 프로젝트 수행 경과

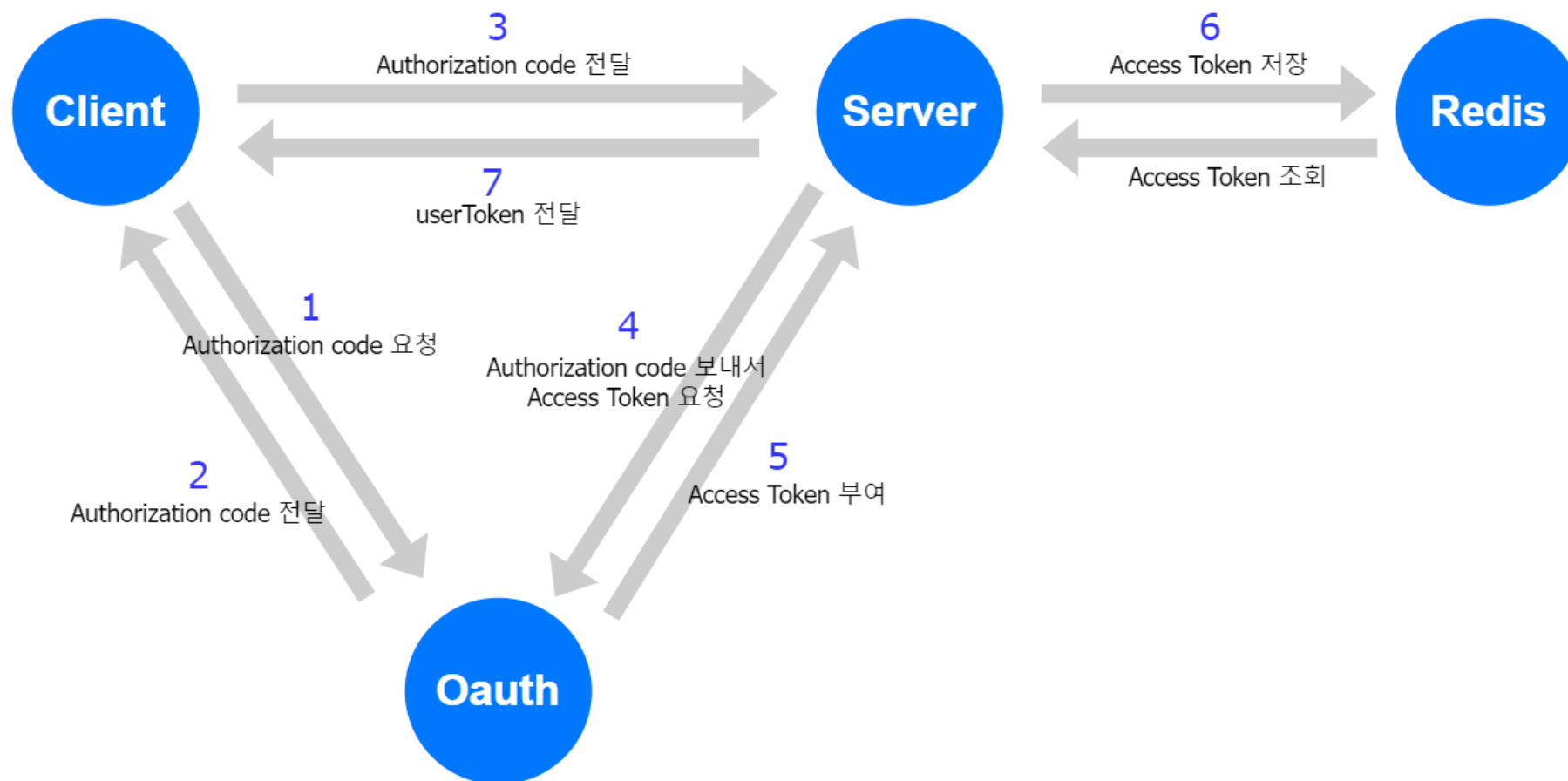
▶ Frontend, Backend 구축

GitHub OAuth와 Redis를 활용한 Oauth 및 토큰 관리 시스템 구축



04 프로젝트 수행 경과

▶ Frontend, Backend 구축



04 프로젝트 수행 경과

▶ Frontend, Backend 구축

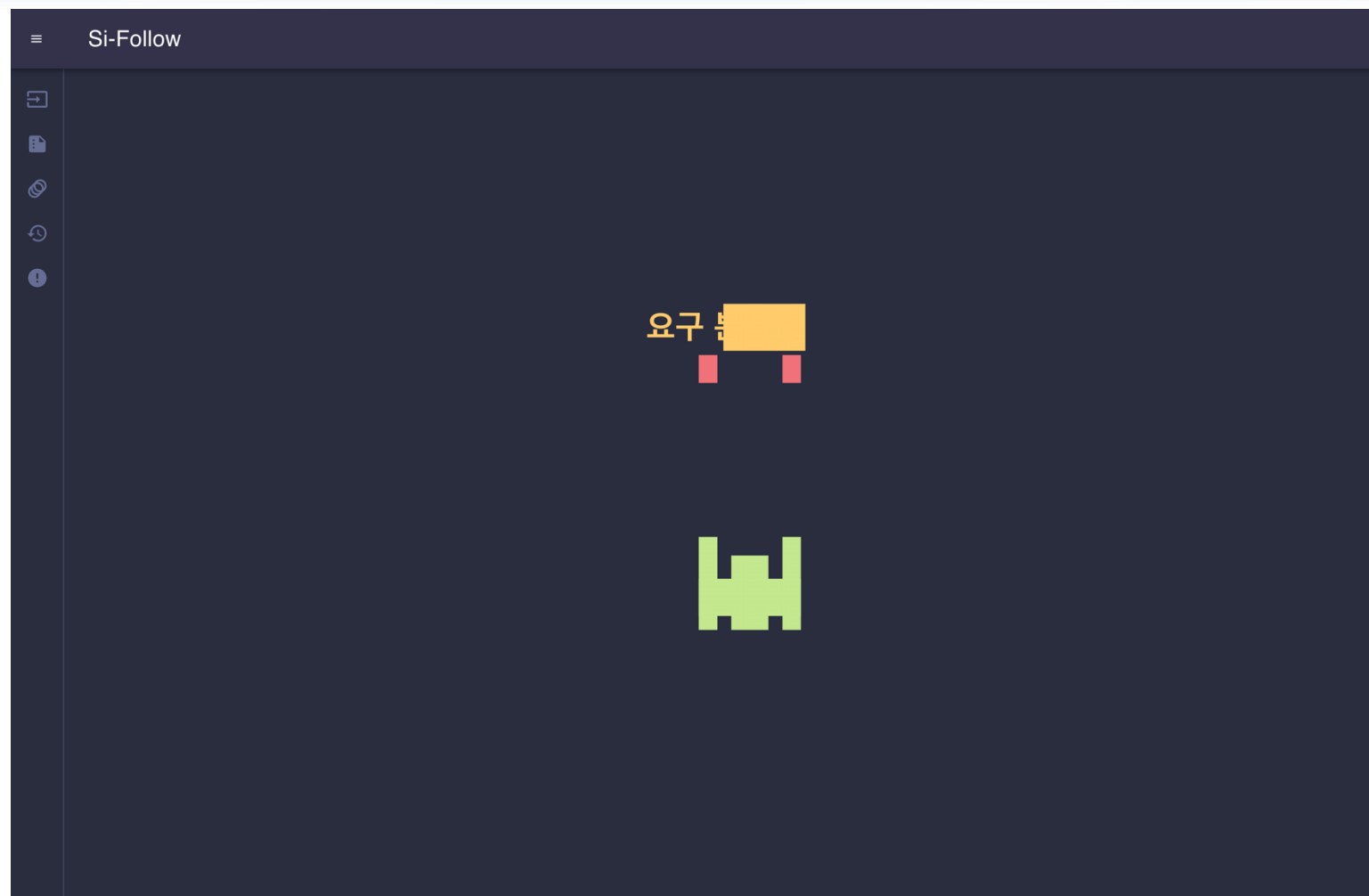
The screenshot shows a web application interface with a dark theme. The title bar at the top says 'Si-Follow'. Below it, there are three tabs: 'Input.html' (active), 'Input.js', and 'Input.css'. The main content area is a form with the following elements:

- Line 1: '유저 토큰 값' (User token value)
- Line 2: Input field containing 'yhoon3002'
- Line 3: '프로젝트 이름' (Project name)
- Line 4: Input field with a placeholder '프로젝트 이름을 입력해주세요 !' (Please enter the project name !)
- Line 5: 'Config 설정' (Config setting)
- Line 6: Two radio buttons, 'Default1' (selected) and 'Default2'.
- Line 7: '요구사항' (Requirement)
- Line 8: A large empty text area for requirements.
- Line 9: A yellow button labeled '보내기' (Send).

At the bottom of the form, there is a footer area with the text '>>> Config 설정: Default'.

04 프로젝트 수행 경과

▶ Frontend, Backend 구축



04 프로젝트 수행 경과

▶ Frontend, Backend 구축

Si-Follow

viewfaq.py

api_views.py

main.py

models.py

faq.py

urls.py

manual.md

meta.txt

e.g

faq.html

settings.py

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

Review 1

Review 1

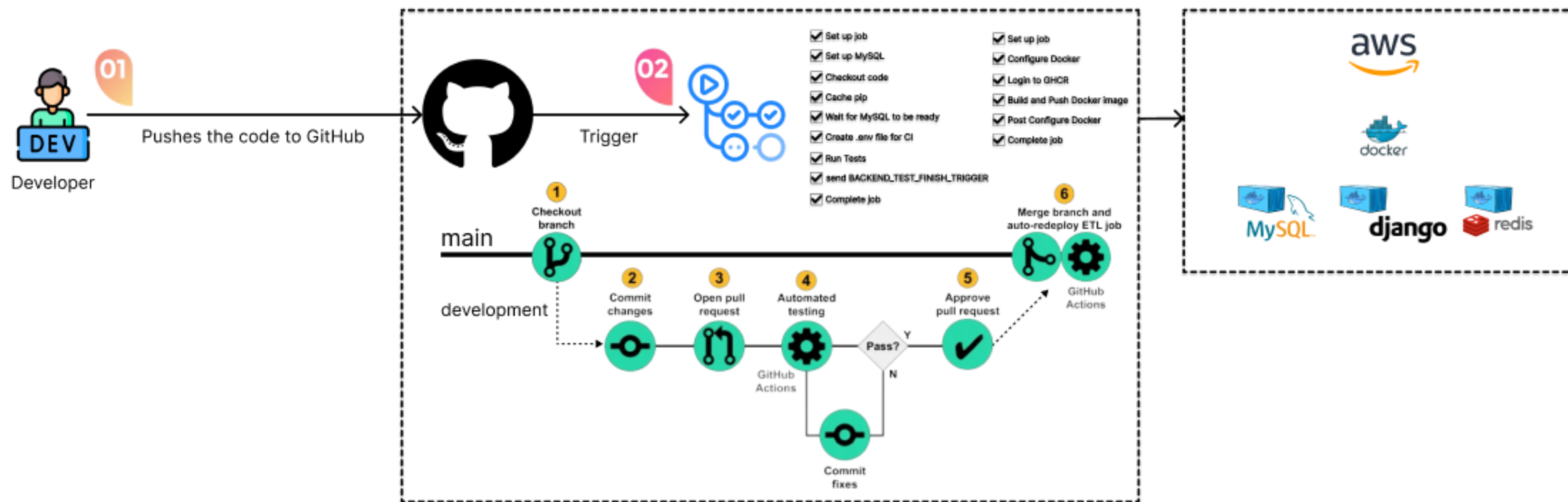
Review 2

» **[Seminar Conclusion]**: After reviewing the provided code, I've identified several areas that require improvement. Here's a detailed analysis of each file: 1. 'main.py': * The code is correct, but it would be better to use a more robust way to run the Django development server, such as using 'django-admin runserver'. 2. 'models.py': * The code seems to be correct, but there's no validation for user input in the 'add_faq' method. This could lead to potential security issues if not handled properly. 3. 'views.py': * The code has several methods that are incomplete or lack comments. For example, the 'handle_request' method only checks if the request method is 'POST' and returns an HttpResponse with a success message. However, it doesn't handle any potential errors that might occur during form submission. * A high-priority comment: + The code does not handle cases where the user submits the form without providing both 'question' and 'answer'. This could lead to unexpected behavior or errors in the database. To fix this, you can add a validation check to ensure that both fields are filled before saving the FAQ. 4. 'faq.html': * The code is correct, but it would be better to use Django's built-in template tags and filters to render the FAQs. This would make the code more reusable and maintainable. 5. 'settings.py': * The code seems to be correct, but you might want to consider using a more secure password hash algorithm for user authentication. High-priority comment: **In views.py, there is no validation check in the handle_request method to ensure both 'question' and 'answer' fields are filled before saving the FAQ. This could lead to unexpected behavior or errors in the database.** To fix this issue, you can add a validation check to ensure that both fields are filled before saving the FAQ: `python def handle_request(self, request): if request.method == 'POST': form_data = request.POST question = form_data.get('question') answer = form_data.get('answer') # Validation check if not question or not answer: return HttpResponse("Please fill both fields") # Save the new FAQ to the model self.faq_model.save() return HttpResponse("FAQ saved successfully")` Additionally, you can consider using Django's built-in form library to handle form validation and submission. This would make your code more reusable and maintainable. Instructions: 1. Update 'views.py' with the revised 'handle_request' method. 2. Review other methods in 'views.py' for completeness and add comments as needed. 3. Consider implementing a more secure password hash algorithm in 'settings.py'. 4. Use Django's built-in template tags and filters to render the FAQs in 'faq.html'. By addressing these issues, you can improve the overall quality and security of your codebase.

» **[Seminar Conclusion]**: The bug in the provided source code is related to how Django is being used. In 'main.py', there's a line where you execute Django using 'execute_from_command_line(['django', 'runserver', '0.0.0.0:8000'])'. However, it looks like your project is not configured correctly for running Django. Here are the issues: 1. You need to specify the 'DJANGO_SETTINGS_MODULE' environment variable or call 'settings.configure()' before accessing settings in 'main.py'. 2. In 'test007/main.py', you're trying to run Django with a non-standard port, which can cause issues depending on how your server is configured. 3. In 'models.py', the class 'Faq' does not inherit from 'models.Model'. To use Django's ORM, you need to define your models as subclasses of 'models.Model'. 4. In 'test007/main.py', after running Django, there is a runtime error when trying to access the FAQs because the Django settings are not properly configured. Here is how you can fix these issues: 1. To specify the 'DJANGO_SETTINGS_MODULE' environment variable, you can use the following command: `bash export DJANGO_SETTINGS_MODULE='your_project.settings'` 2. Alternatively, to call 'settings.configure()' in 'main.py', you would need to know what settings should be configured and how they should be set up for your project. Here's an example of how you can fix the issues with the provided code: 1. Update 'models.py': `python from django.db import models class Faq(models.Model): question = models.CharField(max_length=255) answer = models.TextField()` 2. Create a file named 'settings.py' in your project directory and set up Django's settings: `python import os from pathlib import Path import django # Build paths inside the project like this: BASE_DIR / 'subdir'. BASE_DIR = Path(__file__).resolve() # Quick-start development settings`

04 프로젝트 수행 경과

▶ CI/CD



05 프로젝트 시연

시연링크

06 추후 계획

Agent

1. Backlog phase
2. Github phase
3. HumanInteraction phase
 - ChatWithDoc phase
 - HumanReview phase

LLM

1. 데이터셋 구축
2. Multi-modal 구성을 위한 adapter 모델링

Frontend

1. ChatWithDoc phase UI 구성
2. HumanReview phase UI 구성

Backend

1. 사용기록 저장 DB 구축

Q&A

Thank You