**AI(ChatGPT) 기반 브레인스토밍 웹서비스 개발**

**프로젝트 목표**

* **AWS Lambda와 API Gateway를 활용하여 서버리스 환경에서 ChatGPT 기반 브레인스토밍 웹서비스를 개발.**
* **HTML/CSS/JavaScript로 간단하고 직관적인 사용자 인터페이스(UI)를 구현.**
* **Chain of Thought (COT) 기법을 활용하여 AI 응답의 논리성과 창의성 향상.**
* **팀원들의 다양한 역할(프론트엔드, 백엔드, 클라우드 시스템, 프롬프트 엔지니어링)을 통해 협업과 기술 개발 역량 강화.**

**1주차: 기초 학습과 환경 설정**

**첫 번째 미팅**

* **목표:**
  + **프로젝트 소개 및 역할 분담.**
  + **Python, HTML/CSS/JavaScript 기초 학습 시작.**
* **계획:**
  + **공통:**
    - **프로젝트 목표와 기대 결과물 공유.**
    - **AWS 계정 생성 및 기본 서비스 소개.**
  + **프론트엔드 개발자: HTML/CSS 구조 이해 및 간단한 웹 페이지 설계 실습.**
  + **백엔드 개발자: Python의 변수, 조건문, 반복문 학습. 간단한 입력/출력 프로그램 작성.**
  + **클라우드 시스템 담당자: AWS Lambda와 API Gateway 기본 사용법 학습.**
  + **프롬프트 엔지니어: ChatGPT API 개념 이해 및 간단한 프롬프트 실습.**

**두 번째 미팅**

* **목표:**
  + **Python과 HTML/CSS/JavaScript를 활용해 간단한 프로그램 작성.**
  + **팀 역할별 학습 결과 공유.**
* **계획:**
  + **공통:**
    - **첫 미팅 학습 결과 공유.**
    - **간단한 사용자 입력/출력 프로그램 연습.**
  + **프론트엔드 개발자: HTML 폼 구성 및 버튼 이벤트 처리(JavaScript 활용).**
  + **백엔드 개발자: Python으로 "Hello World" Lambda 함수 작성 및 테스트.**
  + **클라우드 시스템 담당자: Lambda와 API Gateway 연동하여 간단한 API 배포.**
  + **프롬프트 엔지니어: 다양한 입력값에 따른 ChatGPT 응답 실험.**

**2주차: 기본 기능 구현**

**첫 번째 미팅**

* **목표:**
  + **사용자 입력값 처리와 Lambda 연동.**
  + **API 설계 및 기본 흐름 구현.**
* **계획:**
  + **공통:**
    - **사용자 입력값 → Lambda 처리 → 응답 반환 구조 논의.**
  + **프론트엔드 개발자:**
    - **사용자 입력 필드와 API 호출(fetch) 구현.**
  + **백엔드 개발자:**
    - **Lambda에서 사용자 입력값을 처리하고 기본 응답 생성.**
  + **클라우드 시스템 담당자:**
    - **API Gateway에서 POST 요청 처리 설정.**
  + **프롬프트 엔지니어:**
    - **브레인스토밍 주제를 기반으로 한 초기 프롬프트 설계.**

**두 번째 미팅**

* **목표:**
  + **API 호출 테스트 및 기본 흐름 점검.**
* **계획:**
  + **공통:**
    - **프론트엔드에서 API 호출 테스트.**
    - **통합 작업에서 발생한 문제 논의 및 디버깅.**
  + **프론트엔드 개발자:**
    - **API 응답 데이터를 화면에 표시.**
  + **백엔드 개발자:**
    - **ChatGPT API 호출을 Lambda에 통합.**
  + **클라우드 시스템 담당자:**
    - **API Gateway 배포 및 엔드포인트 안정화.**
  + **프롬프트 엔지니어:**
    - **ChatGPT 응답 품질 테스트 및 피드백 수집.**

**3주차: 프론트엔드와 백엔드 연동**

**첫 번째 미팅**

* **목표:**
  + **사용자 입력값과 응답 데이터를 통합.**
* **계획:**
  + **공통:**
    - **데이터 흐름을 연결하고 테스트.**
  + **프론트엔드 개발자:**
    - **UI 개선 및 입력/출력 동작 디버깅.**
  + **백엔드 개발자:**
    - **ChatGPT API 호출 최적화 및 오류 처리 추가.**
  + **클라우드 시스템 담당자:**
    - **API Gateway와 Lambda의 응답 시간 최적화.**
  + **프롬프트 엔지니어:**
    - **응답 데이터 구조 설계 및 개선안 제시.**

**두 번째 미팅**

* **목표:**
  + **전체 데이터 흐름 통합 및 기능 점검.**
* **계획:**
  + **공통:**
    - **데이터 흐름 통합 후 테스트.**
  + **프론트엔드 개발자:**
    - **동적인 UI 렌더링 추가.**
  + **백엔드 개발자:**
    - **Lambda 응답 데이터 구조 정리.**
  + **클라우드 시스템 담당자:**
    - **API 호출 과정에서 발생한 성능 문제 디버깅.**
  + **프롬프트 엔지니어:**
    - **입력값에 따른 다양한 테스트 케이스 실행.**

**4주차: Chain of Thought (COT) 기법 적용**

**첫 번째 미팅**

* **목표:**
  + **COT 기법 이해 및 설계.**
* **계획:**
  + **공통:**
    - **COT 기법 개념 논의 및 적용 방안 설계.**
  + **프론트엔드 개발자:**
    - **COT 기반 응답 데이터를 단계별로 표시.**
  + **백엔드 개발자:**
    - **ChatGPT API 호출에 COT 프롬프트 적용.**
  + **클라우드 시스템 담당자:**
    - **Lambda 함수 성능 테스트 진행.**
  + **프롬프트 엔지니어:**
    - **COT 적용을 위한 프롬프트 최적화.**

**두 번째 미팅**

* **목표:**
  + **COT 결과 점검 및 개선.**
* **계획:**
  + **공통:**
    - **COT 적용 후 전체 응답 데이터 점검.**
  + **프론트엔드 개발자:**
    - **COT 결과를 UI에 직관적으로 표시.**
  + **백엔드 개발자:**
    - **COT 기반 데이터의 구조 개선.**
  + **클라우드 시스템 담당자:**
    - **Lambda와 API Gateway 트래픽 테스트.**
  + **프롬프트 엔지니어:**
    - **COT 품질 평가 및 개선안 도출.**

**5주차: 기능 확장 및 통합 테스트**

**첫 번째 미팅**

* **목표:**
  + **추가 기능 구현(결과 저장/다운로드).**
* **계획:**
  + **프론트엔드 개발자:**
    - **결과 저장 버튼 및 UI 추가.**
  + **백엔드 개발자:**
    - **결과 데이터 저장 API 구현.**
  + **클라우드 시스템 담당자:**
    - **DynamoDB 또는 S3 연동(선택 사항).**
  + **프롬프트 엔지니어:**
    - **기능 확장에 따른 응답 품질 재검토.**

**두 번째 미팅**

* **목표:**
  + **전체 시스템 통합 테스트.**
* **계획:**
  + **공통:**
    - **통합 작업 결과 점검 및 디버깅.**
  + **프론트엔드 개발자:**
    - **UI와 API 동작 최종 점검.**
  + **백엔드 개발자:**
    - **에러 처리 및 API 안정화.**
  + **클라우드 시스템 담당자:**
    - **API 성능 최적화 및 배포 상태 점검.**
  + **프롬프트 엔지니어:**
    - **최종 프롬프트 테스트 및 보고서 작성.**

**6주차: 프로젝트 발표 및 리뷰**

**첫 번째 미팅**

* **목표:**
  + **프로젝트 발표 및 피드백 공유.**
* **계획:**
  + **팀별 발표 및 데모.**
  + **학습 경험과 개선점 논의.**
  + **향후 발전 방안 제시.**

**최종 결과물**

* **AI 브레인스토밍 웹서비스:**
  + **사용자 입력값에 따라 ChatGPT가 창의적이고 논리적인 아이디어를 생성.**
  + **HTML/CSS/JavaScript로 설계된 간단한 UI.**
  + **서버리스 환경에서 AWS Lambda와 API Gateway로 작동.**
* **문서화 및 발표 자료:**
  + **시스템 설계 문서, API 명세서, 사용 가이드, 프로젝트 발표 자료.**