📌 Python 기본 용어

- 클래스(Class): 객체 지향 프로그래밍에서 객체를 생성하기 위한 설계도.
- 객체(Object): 클래스에서 생성된 인스턴스.
- 함수(Function): 특정 작업을 수행하는 코드 블록.
- 메서드(Method): 클래스 내에서 정의된 함수로, 인스턴스가 호출 가능.
- 모듈(Module): Python 코드가 포함된 파일로, 여러 기능을 모아둔 단위.
- 패키지(Package): 여러 모듈을 포함하는 디렉토리 구조.
- 변수(Variable): 값을 저장하는 공간.
- 리스트(List): 순서가 있는 변경 가능한 데이터 구조.
- 튜플(Tuple): 순서가 있지만 변경할 수 없는 데이터 구조.
- 딕셔너리(Dictionary): 키-값 쌍으로 이루어진 데이터 구조.
- 이터레이터(Iterator): next()를 통해 순차적으로 값을 반환하는 객체.
- 제너레이터(Generator): yield 키워드를 사용하여 값을 반환하는 함수.
- 람다 함수(Lambda Function): 익명 함수로, 한 줄로 작성된 함수.
- 데코레이터(Decorator): 함수를 감싸서 기능을 확장하는 Python 기능.

📌 데이터 분석 및 과학 연산

- NumPy: 다차원 배열(ndarray)과 고속 수학 연산을 지원하는 라이브러리.
- Pandas: 테이블 형태의 데이터 처리(DataFrame)를 지원하는 라이브러리.
- SciPy: 과학 연산 및 통계 분석을 위한 라이브러리.

📌 머신러닝 기본 개념

- 지도학습(Supervised Learning): 입력과 출력(label)이 주어진 학습 방식.
- 비지도학습(Unsupervised Learning): 출력(label) 없이 패턴을 찾는 학습 방식.
- 강화학습(Reinforcement Learning): 보상을 기반으로 학습하는 방식.
- 과적합(Overfitting): 학습 데이터에 너무 최적화되어 일반화가 어려운 현상.

- 정규화(Normalization): 데이터의 스케일을 조정하여 학습 안정성을 높이는 방법.
- 특징 공학(Feature Engineering): 데이터에서 중요한 특징을 추출하고 변형하는 과정.

📌 딥러닝 개념

- 뉴런(Neuron): 신경망에서 정보를 처리하는 기본 단위.
- 활성화 함수(Activation Function): 뉴런이 활성화되는 정도를 결정하는 함수 (예: ReLU, Sigmoid, Softmax).
- 가중치(Weight)와 편향(Bias): 학습 가능한 파라미터로, 뉴런의 출력에 영향을 미침.
- 손실 함수(Loss Function): 모델의 예측값과 실제값의 차이를 측정하는 함수.
- 역전파(Backpropagation): 오차를 줄이기 위해 가중치를 업데이트하는 과정.
- 전이 학습(Transfer Learning): 사전 학습된 모델을 활용하여 새로운 문제를 해결하는 방식.
- Fine-tuning: 사전 학습된 모델을 특정 태스크에 맞춰 추가 학습하는 과정.

📌 자연어 처리(NLP) 개념

- 토큰화(Tokenization): 문장을 단어나 subword 단위로 나누는 과정.
- 스톱워드(Stopword): 분석에서 제거되는 의미 없는 단어 (예: "the", "is").
- 어간 추출(Stemmming)과 표제어 추출(Lemmatization): 단어의 기본형을 찾는 과정.
- Bag-of-Words (BoW): 단어 출현 빈도를 기반으로 문서를 표현하는 방식.
- TF-IDF (Term Frequency-Inverse Document Frequency): 단어의 중요도를 계산하는 방법.
- Word2Vec: 단어를 벡터로 변환하는 방법으로, CBOW와 Skip-gram 방식이 있음.
- BERT (Bidirectional Encoder Representations from Transformers): 양방향 문맥을 학습하는 사전 학습 모델.

📌 RAG (Retrieval-Augmented Generation) 관련 개념

- RAG (Retrieval-Augmented Generation): 검색과 생성 모델을 결합하여 정보를 더욱 정확하게 생성하는 방식.

- Vector Store: 검색을 위한 벡터 데이터베이스 (예: FAISS, Pinecone, ChromaDB).
- Embedding Model: 텍스트를 벡터 형태로 변환하는 모델 (예: OpenAI Ada, BERT Embeddings).
- FAISS (Facebook AI Similarity Search): 빠른 유사도 검색을 위한 라이브러리.
- BM25: 전통적인 키워드 검색 알고리즘으로, RAG에서 Hybrid Search에 활용됨.
- Hybrid Search: BM25와 Vector Search를 결합하여 검색 성능을 향상시키는 방식.
- Context Window: LLM이 한 번에 처리할 수 있는 최대 토큰 수.
- Reranker: 검색 결과를 LLM이 더 적절한 순서로 재정렬하는 기술.

📌 웹 개발 기본 개념

- HTTP (HyperText Transfer Protocol): 클라이언트와 서버 간 데이터를 주고받는 프로토콜.
- REST API (Representational State Transfer API): HTTP 기반으로 자원을 CRUD 방식으로 관리하는 아키텍처 스타일.
- JSON (JavaScript Object Notation): 서버와 클라이언트 간 데이터를 교환하는 경량 데이터 형식.
- MVC (Model-View-Controller): 웹 애플리케이션의 구조를 나누는 설계 패턴.
- Middleware (미들웨어): 요청과 응답 사이에서 추가적인 처리를 수행하는 소프트웨어 계층.

📌 CRUD (Create, Read, Update, Delete) 개념

- Create (생성): 새로운 데이터를 데이터베이스에 추가하는 작업 (POST).
- Read (조회): 데이터베이스에 저장된 데이터를 가져오는 작업 (GET).
- Update (수정): 기존 데이터를 변경하는 작업 (PUT 또는 PATCH).
- Delete (삭제): 데이터를 제거하는 작업 (DELETE).

📌 데이터베이스 개념

- RDBMS (Relational Database Management System): 관계형 데이터베이스 관리 시스템 (예:

MySQL, PostgreSQL).

- NoSQL: 비관계형 데이터베이스로, 문서, 키-값, 그래프 데이터 저장 방식 지원 (예: MongoDB).
- ORM (Object-Relational Mapping): 객체와 데이터베이스를 매핑하는 기술 (예: SQLAlchemy, Django ORM).
- Foreign Key (외래 키): 다른 테이블과 관계를 맺는 키.
- Index (인덱스): 데이터 검색 속도를 높이기 위한 데이터베이스 구조.
- ACID (Atomicity, Consistency, Isolation, Durability): 트랜잭션의 신뢰성을 보장하는 데이터베이스 속성.

📌 웹 프레임워크 및 API 개발

- Flask: Python 기반의 가벼운 웹 프레임워크.
- FastAPI: 비동기 처리 기반의 고성능 웹 프레임워크.
- Django: Python 기반의 풀스택 웹 프레임워크.
- JWT (JSON Web Token): 사용자 인증과 권한 관리를 위한 토큰 기반 인증 방식.
- OAuth 2.0: API 인증을 위한 표준 프로토콜.
- CORS (Cross-Origin Resource Sharing): 웹 애플리케이션에서 다른 도메인의 리소스를 호출할 수 있도록 허용하는 정책.
- Rate Limiting (요청 제한): API 요청 횟수를 제한하는 기술.

📌 게시판 기능 관련 개념

- Pagination (페이지네이션): 게시글 목록을 여러 페이지로 나누어 출력하는 기능.
- Sorting (정렬): 최신 게시글, 인기 게시글 등을 기준으로 정렬하는 기능.
- Filtering (필터링): 특정 조건(예: 태그, 작성자 등)에 따라 게시글을 검색하는 기능.
- File Upload (파일 업로드): 게시글에 이미지, 파일을 첨부하는 기능.
- User Authentication (사용자 인증): 로그인/회원가입 기능 (예: JWT, OAuth).
- Authorization (권한 관리): 사용자 역할(관리자, 일반 사용자)에 따라 기능을 제한하는 방

식.

- WebSocket: 실시간 댓글, 알림 기능을 구현할 때 사용되는 양방향 통신 프로토콜.

📌 성능 최적화 및 보안 개념

- Caching (캐싱): 데이터 조회 속도를 높이기 위해 Redis, Memcached 등을 사용하는 기법.
- Lazy Loading (지연 로딩): 필요할 때만 데이터를 로드하는 방식으로 성능 최적화.
- SQL Injection (SQL 인젝션): 악성 SQL을 주입하여 데이터베이스를 공격하는 방식.
- XSS (Cross-Site Scripting): 악성 스크립트를 삽입하여 사용자의 데이터를 탈취하는 공격.
- CSRF (Cross-Site Request Forgery): 사용자의 의도와 다르게 요청을 보내는 공격.
- HTTPS (HyperText Transfer Protocol Secure): 데이터를 암호화하여 안전하게 전송하는 프로토콜.