프레임워크 학습을 앞두고

1. 들어가며

지금까지 Python, Web 기초와 알고리즘 과정을 마쳤습니다.

이제는 더 나아가 **Django**라는 프레임워크(Framework)를 학습할 차례입니다.

프레임워크는 단순히 알고리즘 문제를 풀듯 "코드를 작성하는" 방식과는 다른 접근이 필요합니다.

프레임워크를 어떻게 바라봐야 하고, **왜 배우는지**, **어떻게 학습하면 좋을지**를 간단히 알아봅시다.

---

2. 알고리즘 학습과의 차이점

2.1 문제 풀이 방식의 차이

- **알고리즘 학습**
    - 주어진 문제를 해결하기 위해 **입력** → **처리** → **출력** 과정을 **함수**나 간단한 **코드**로 직접 구현
    - 원하는 결과를 얻기 위해 **어떤 자료구조**를 쓰고, **어떤 로직**을 짜야 하는지 고민
    - 코드 규모가 작고, 핵심은 **효율과 정확성**
- **프레임워크 학습**
    - 웹 애플리케이션 전체를 구성하는 **큰 틀**과 **흐름**을 이해
    - 이미 만들어진 **기능**(로그인, DB 관리, URL 매핑 등)을 어떻게 활용하고 조합할지 학습
    - 개발 속도를 빠르게 하고 유지보수를 수월하게 하는 **"구조"**가 중요한 키워드

2.2 구현 vs 활용

- **알고리즘**에서는 필요한 로직을 **직접 구현**해왔습니다.
- **프레임워크**는 이미 제공되는 구성 요소(기능, 라이브러리, 설정)를 **적절히 활용**해 **애플리케이션**을 만드는 과정입니다.

---

3. 프레임워크란 무엇인가?

3.1 큰 틀(Framework)의 개념

- **프레임워크**는 "틀" 혹은 "골격"을 의미합니다.
- 웹 개발에서 **자주 사용하는** 기능들을 미리 만들어 놓은 **코드 집합**으로, 개발자가 원하는 기능을 빠르게 조립하고 확장하도록 돕습니다.
- **Django**는 웹사이트에 필요한 **데이터베이스 연동**, **URL 라우팅**, **템플릿 처리** 등 다양한 기능을 이미 갖추고 있습니다.

3.2 왜 배우는가?

1. 개발 속도 향상
    - 반복적으로 구현해야 할 작업을 자동화하거나 간소화해줘서, **핵심 로직**에 집중할 수 있음.
2. 유지보수 편리
    - 프레임워크가 제공하는 구조(프로젝트 폴더 구조, 앱 구조 등)를 따름으로써, 팀원들과 협업 시 코드가 체계적이고 일관성 있게 관리됨.
3. 실무 표준
    - 대부분의 실제 웹 서비스들이 Django, Spring, Vue, React 등 **프레임워크**를 활용해 개발됨.
    - 이를 배우면 본격적인 실무적 프로젝트에 가까워짐.

---

4. 어떤 관점으로 학습해야 할까?

4.1 내부 동작을 전부 이해할 필요는 없다

- 프레임워크 내부에는 방대한 코드와 복잡한 동작 원리가 있습니다.
  이를 모든 것을 다 이해하려고 하면 불필요한 **시간과 에너지를 낭비**할 수 있습니다.
- 대신 **프레임워크가 제공하는 주요 기능**을 중심으로, **"어떤 기능이 어디에 있고, 어떻게 활용할 수 있는지"**에 집중하세요.

4.2 "어떻게 만들지?"보다 "어떻게 `조합` 할지?"에 집중

- 지금까지는 문제를 **어떻게 구현**할지 직접 구상했지만,
  이제는 **프레임워크가 제공하는 기능**을 **어떻게 조합하고 연결**해서 하나의 웹 애플리케이션을 만들지 고민하는 게 핵심입니다.

4.3 예시 프로젝트와 실습

- 공식 문서 및 여러 자료를 읽고, 예시 코드를 실제로 돌려보며 **손으로 익히는** 과정이 중요합니다.
- 처음에는 작은 웹 앱(게시판, 할 일 목록 등)을 만들고, 그 흐름을 파악해보세요.

---

5. 프레임워크를 통해 최종적으로 무엇을 할까?

1. 웹 애플리케이션 개발
   - 로그인, 회원가입, 게시판, 댓글, 파일 업로드 등 **실제 웹 서비스**에 가까운 기능을 다뤄볼 수 있음.
2. 데이터베이스 연동
   - 프레임워크서 제공하는 모델(Model) 시스템으로, 데이터베이스를 **간편하게** 다룰 수 있음.
3. 프로젝트 구조화 및 협업
   - 프로덕션(배포, 실제 서비스) 단계에서의 디렉토리 구조, 앱 분리, 설정 파일 관리 등을 학습.
   - 협업 시 다른 사람이 짠 코드와 **일관성** 있게 맞출 수 있음.

---

6. 정리

1. 프레임워크는 `"이미 만들어진 도구"`
   - 모든 내부 코드를 해부하려 하기보다, 문서와 예제 코드를 보며 **활용법**을 빠르게 습득.
2. 프로젝트 단위 실습
   - 간단한 CRUD 기능(게시판, todo 등)을 만들어보고, 프로젝트 흐름을 익힌 뒤 확장.
3. 협업을 준비하며
   - 가독성, 구조화, 디자인 패턴 등을 의식하며 코드를 작성.

> 프레임워크 학습은 `"조립"`과 `"활용"`에 초점을 맞추는 것이 핵심입니다.
>
> 알고리즘처럼 한정된 문제를 효율적으로 해결하는 것이 아니며, **주어진 기능을 조합해 실제 서비스를 만드는 과정**입니다.
>
> 충분한 실습과 함께 점차 **Django가 제공하는 기능**들을 익혀나가면, **웹 개발 전반**을 이해하게 될 것입니다.