컨벤션

깃 컨벤션

깃허브 과정

1. 내가 작업한 feat 브랜치를 우선 커밋한다.
2. 원격 dev 브랜치 origin/dev를 최신화하기 위해
   `git fetch origin dev`
3. 내가 작업한 feat 브랜치에서 origin/dev와 merge 한다.
   `git merge origin/dev`
4. 충돌이 발생하는 부분을 수정한다.
5. 충돌 해결 후
   `git add -A`
   `git commit -m "merge: dev to feat 브랜치"`
6. 원격 저장소에 feat 브랜치를 push한다.
   `git push origin feat 브랜치`
7. 깃허브에서 dev로 pr을 생성한다.
8. pr을 적용한다.

`태그 : 제목`의 형태이며, `:`뒤에만 space가 있음에 유의한다.

- `feat` : 새로운 기능 추가
- `fix` : 버그 수정
- `docs` : 문서 수정
- `style` : 코드 포맷팅, 세미콜론 누락, 코드 변경이 없는 경우
- `refactor` : 코드 리펙토링
- `test` : 테스트 코드, 리펙토링 테스트 코드 추가
- `chore` : 빌드 업무 수정, 패키지 매니저 수정

코딩 컨벤션

- 컨트롤러 계층 메서드 네이밍

동작	HTTP 메서드	URL 예시	컨트롤러 메서드 네이밍 예시
저장(생성)	**POST**	`/products`	`createProduct()`
단건 조회	**GET**	`/products/{id}`	`getProduct()`
전체 조회	**GET**	`/products`	`getProducts()`
수정	**PUT/PATCH**	`/products/{id}`	`updateProduct()`
삭제	**DELETE**	`/products/{id}`	`deleteProduct()`

- 서비스 계층 메서드 네이밍
  - 기본 CRUD
    - save
    - findById
    - findAll
    - deleteById
  - 의미 있는 기능을 가지는 메서드
    - **상품 구매 여부 확인**
      - `hasPurchase(userId, productId)` → boolean 리턴
    - **상품 재고 체크**
      - `isOutOfStock(productId)`

DTO

- Inner class

Exception

- BaseException
- BaseResponseStatus로 메세지 처리

색 테마

기본 - slate-600

DB 계정

- username: halo
- pssword: qwer1234
- database 이름: core-bridge

각자 로컬에 계정생성하는 방법은 강사님 DB 설정 파일 참고