

1) Repository + Service réteg (Products)

Cél: MySQLi prepared statement + OOP rétegezés (Domain/Repo/Service).

Feladat:

- Hozzon létre products táblát (schema.sql).
- Készítsen Product domain osztályt (id, name, priceFt) validációval.
- Készítsen ProductRepository interface-t és MysqliProductRepository implementációt:
 - all() listázás,
 - add(Product \$p) beszúrás prepared statement-tel.
- Készítsen ProductService-t, ami a repository-t használja.
- index.php:
 - űrlap: név + ár,
 - validáció + PRG + flash üzenet,
 - lista táblázatban.

2) Tranzakció a Service rétegben (Orders + OrderItems)

Cél: domain model + repository + service, ahol a service kezeli a tranzakciót (begin/commit/rollback).

Feladat:

- Táblák: products, orders, order_items (schema.sql).
- Domain: Order + OrderItem.
- Repo-k: ProductRepo (termékek, exists), OrderRepo (insert order, insert items, summary).
- Service: OrderService, ami:
 - begin_transaction(),
 - beszúrja az order-t + itemeket,
 - hiba esetén rollback.
- index.php:
 - vevő + termék-mennyiségek,
 - validáció (0-99, legalább 1 tétel),

- rendelés mentés,
- összesítő lista JOIN+SUM alapján.

3) Active Record stílus (Student) – összehasonlítás célból

Cél: MySQLi + OOP, de **entitásba csomagolt DB műveletek** (tanítási kontraszt a repository-val).

Feladat:

- students tábla (schema.sql).
- Student osztály:
 - konstruktor validál,
 - insert(mysqli \$db),
 - static all(mysqli \$db),
 - static deleteById(mysqli \$db, int \$id).
- index.php:
 - űrlap hallgató felvételre (name, neptun, points),
 - listázás,
 - törlés (POST),
 - PRG + flash.

4) Mini MVC MySQL-lel: Front controller + Router + Controller + Repository

Cél: egyszerű MVC-s felépítés MySQLi-val (controller csak a repo-t hívja, view-k külön).

Feladat:

- tasks tábla (schema.sql).
- Készítsen:
 - Container (mini DI),
 - Router (action alapján dispatch),
 - TodoRepository (CRUD prepared statement-tel),
 - TodoController (list/create/toggle/delete),
 - views/layout.php, views/list.php, views/create.php.

- index.php legyen front controller:
 - container felépítése,
 - route-ok,
 - dispatch és render.
-

5) Query Object + lapozás OOP-ban (Products search)

Cél: keresési paraméterek objektumba (Query Object), lapozás logikája service-ben, MySQLi prepared LIKE + LIMIT/OFFSET.

Feladat:

- products tábla seed adatokkal (schema.sql).
- ProductQuery (q, page, perPage) + like(), offset().
- ProductSearchRepository:
 - count(ProductQuery \$q) prepared,
 - page(ProductQuery \$q) prepared LIMIT/OFFSET.
- ProductSearchService:
 - kiszámolja az oldalszámot,
 - "clamp"-eli a page-et,
 - visszaad total/pages/rows.
- index.php:
 - keresés + lapozás UI.