Database Connectivity

Web Programming and Testing

Mario Simaremare, S.Kom., M.Sc.
Program Studi Sarjana Sistem Informasi
Institut Teknologi Del

Objectives

- The objective of this session is the following:
  - The students are able to elaborate the role of persistent data storage in a web application.
  - The students are able to interact with relational database.

Outlines

1. Motivation
2. Persistence data storage
   - Examples in PHP: native and PDO-based

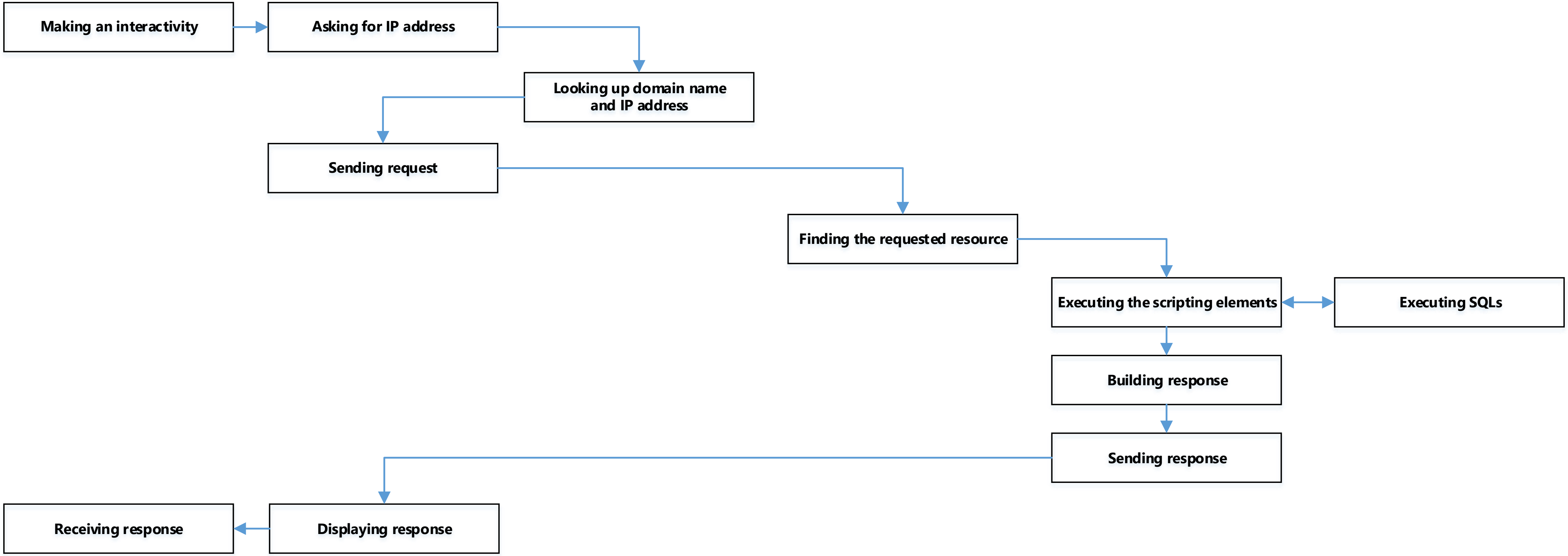
Motivation

Web 2.0: Interactivity

- Web 2.0 is minimal: dynamic content.
  - User should be able to contribute.

- Server-side processing is needed.
  - + persistence data storage.

General Workflow



Making an interactivity

Asking for IP address

Looking up domain name and IP address

Sending request

Finding the requested resource

Executing the scripting elements

Executing SQLs

Building response

Sending response

Displaying response

Receiving response

Persistence Data Storage

Persistence Data Storage

- The storage that store data emerges from user and application interactivity.

- Forms of persistence data storage:
  - Database (relational or non-relational).
  - Plain-text (XML, JSON).

- We could do DDL and DML.

From This Point Forward

- Focus on relational database as the primary data storage.

- PHP is the platform used as our vehicle.

Database Connectivity

- Connectivity can be achieved via three approaches:
  - Native driver, plainly using the vendor’s library.
  - PDO (PHP Data Object), a standard interfaces.
  - ORM (Object-Relational Mapping), will be discussed later.

Native Driver

- Specific database vendor provides its own APIs.
  - Different vendor might provide different set of APIs.

- Advantage:
  - Plain with less abstraction layer.
  - Faster over any other options.

- Disadvantage:
  - Moving from one to another vendor is hard.
  - More time to learn specific vendor APIs.

PDO (PHP Data Object)

- PDO provides a generic set of standard APIs running on top of vendor PDO implementations.

- Advantage:
  - An extra abstraction layer hides complexity down the road.
  - Moving from one to another vendor is easy.
  - Less time to learn since PDO gives standard.

- Disadvantage:
  - An extra abstraction increase processing time.

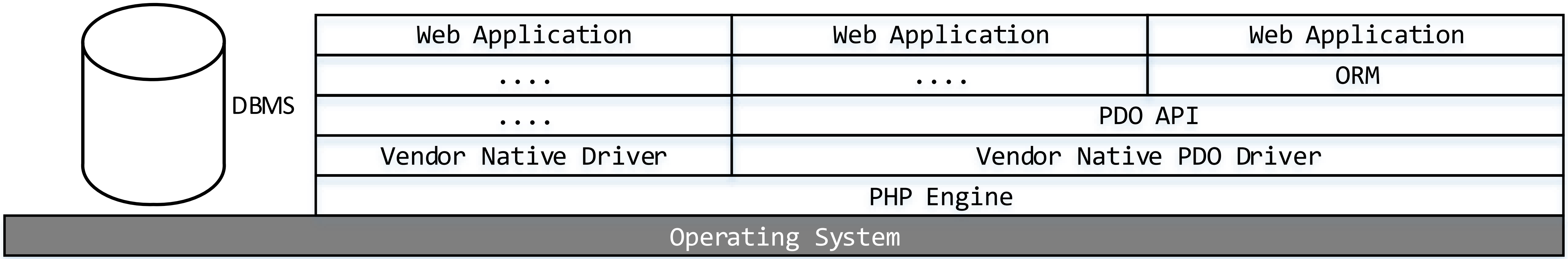
ORM (Object-Relational Mapping)

- Running on top of PDO, proposing a specific point of view.
  - Active record, data mapper, etc.

- Advantage:
  - Everything PDO has.
  - Seeing data in object-oriented sense.

- Disadvantage:
  - An extra abstraction increases processing time.

Comparison



DBMS

Web Application | Web Application | Web Application

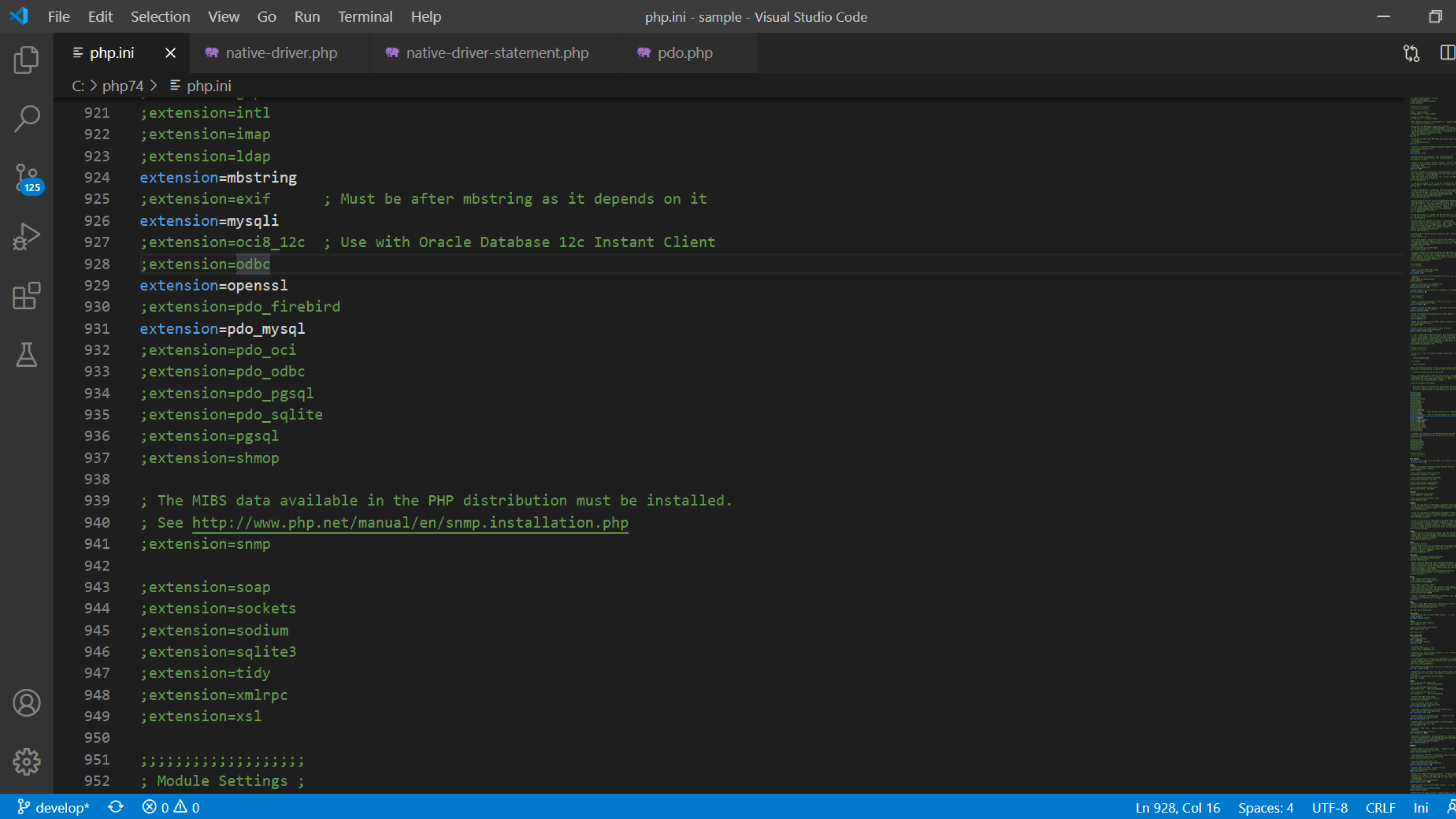
.... | .... | ORM

.... | PDO API

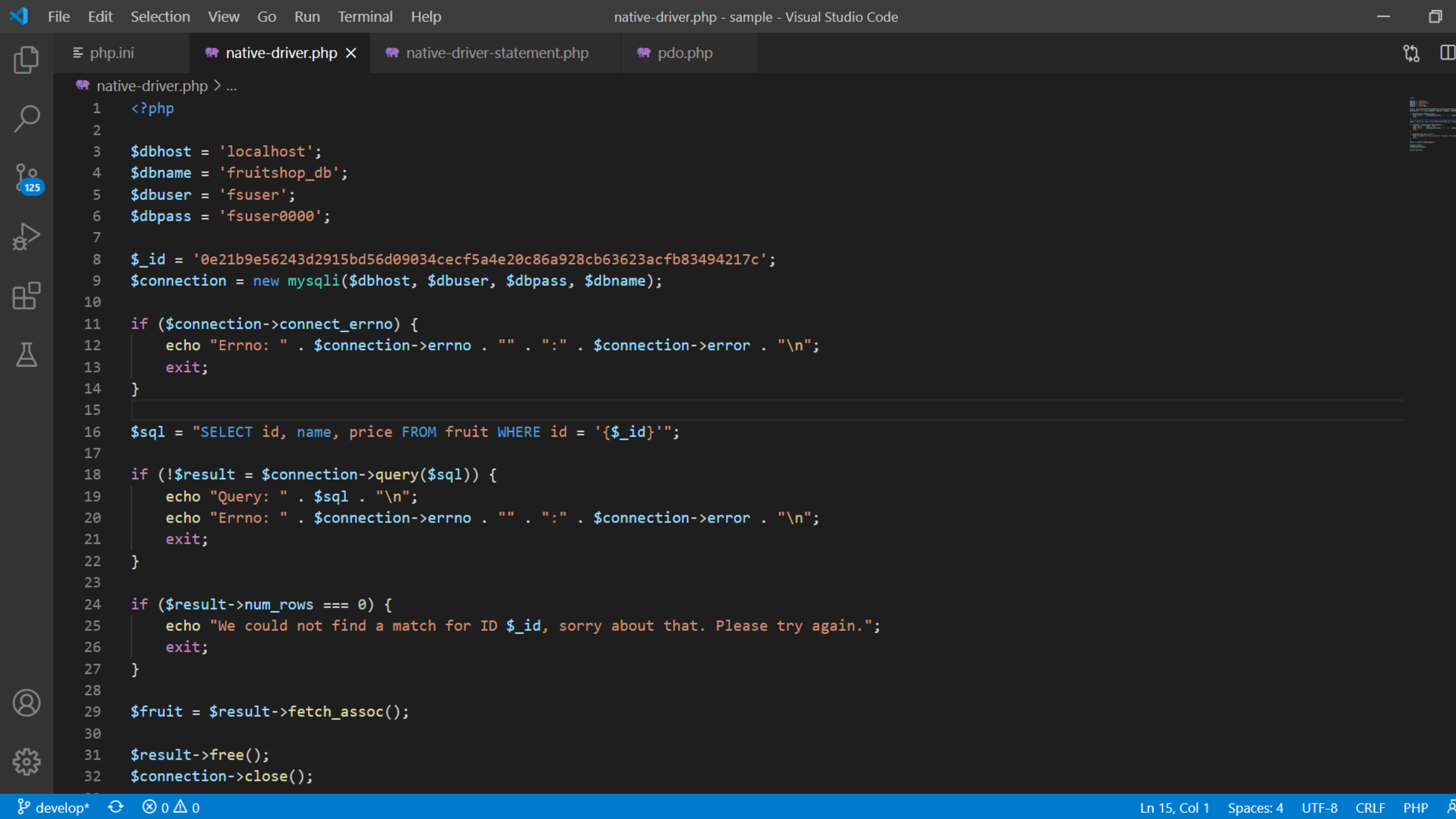
Vendor Native Driver | Vendor Native PDO Driver

PHP Engine

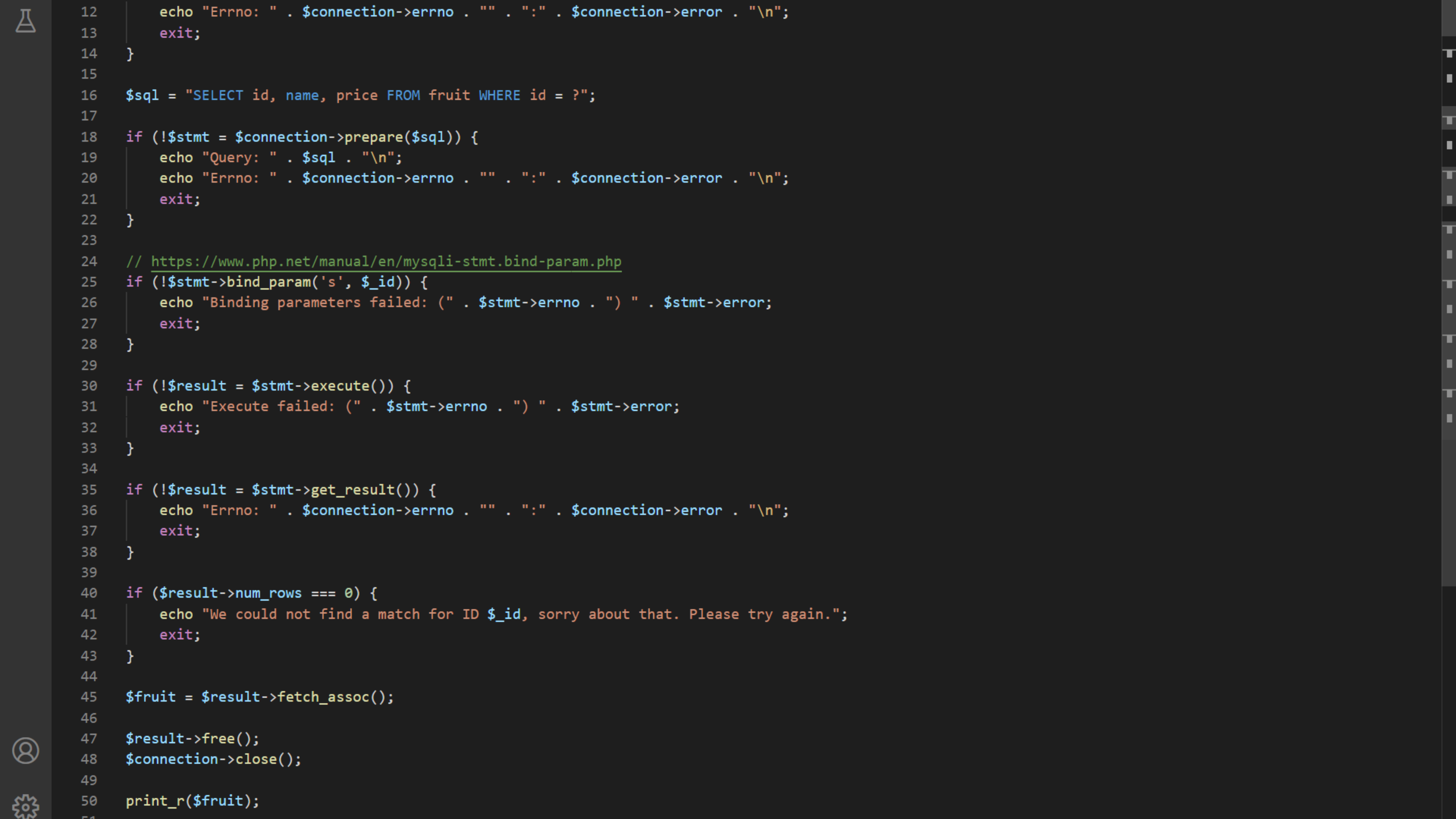
Operating System



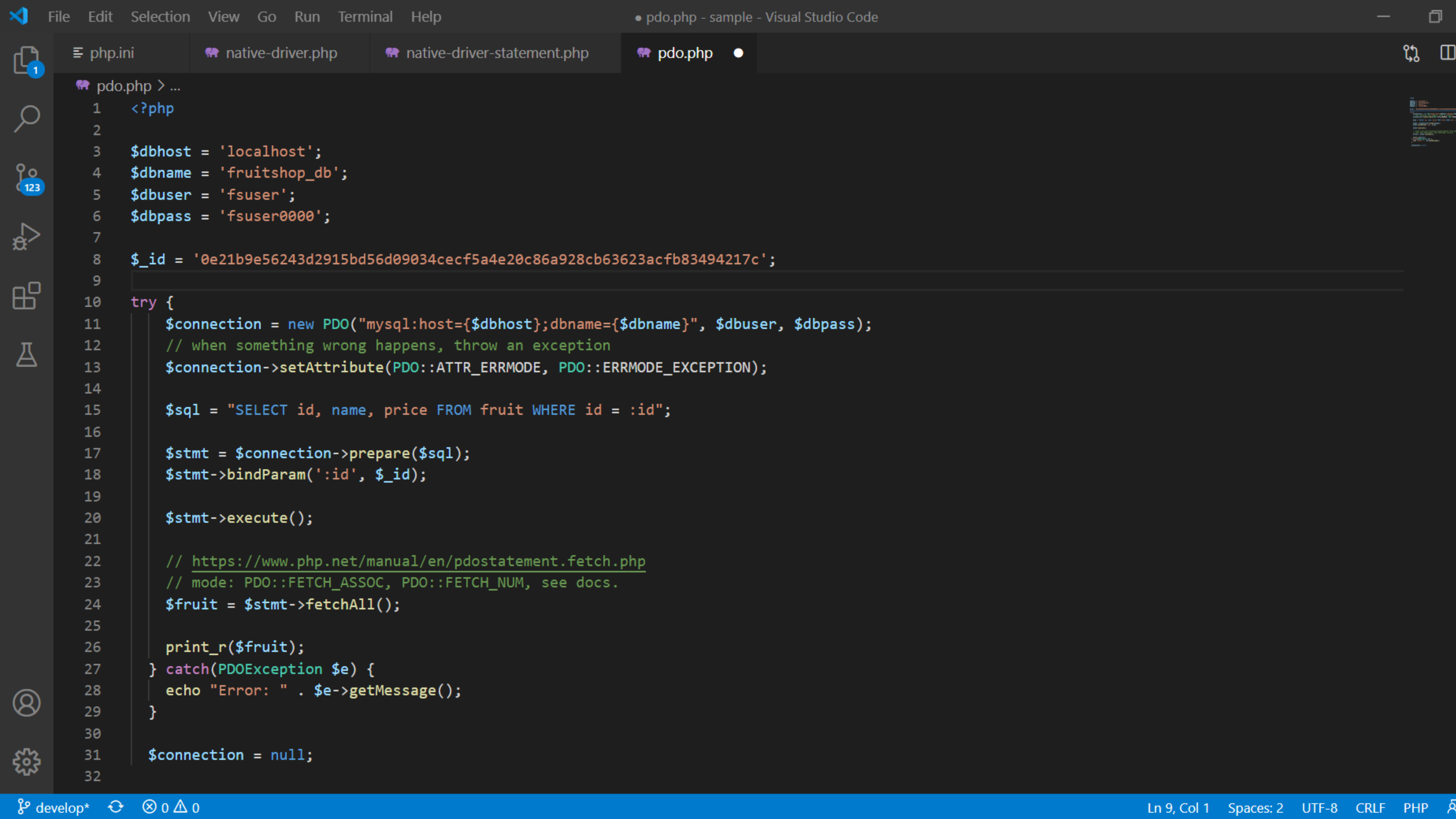
```
;extension=intl
;extension=imap
;extension=ldap
extension=mbstring
;extension=exif      ; Must be after mbstring as it depends on it
extension=mysqli
;extension=oci8_12c  ; Use with Oracle Database 12c Instant Client
;extension=odbc
extension=openssl
;extension=pdo_firebird
extension=pdo_mysql
;extension=pdo_oci
;extension=pdo_odbc
;extension=pdo_pgsql
;extension=pdo_sqlite
;extension=pgsql
;extension=shmop

; The MIBS data available in the PHP distribution must be installed.
; See http://www.php.net/manual/en/snmp.installation.php
;extension=snmp

;extension=soap
;extension=sockets
;extension=sodium
;extension=sqlite3
;extension=tidy
;extension=xmlrpc
;extension=xsl

;;;;;;;;;;;;;;;;;;;
; Module Settings ;
```



```php
<?php

$dbhost = 'localhost';
$dbname = 'fruitshop_db';
$dbuser = 'fsuser';
$dbpass = 'fsuser0000';

$_id = '0e21b9e56243d2915bd56d09034cecf5a4e20c86a928cb63623acfb83494217c';
$connection = new mysqli($dbhost, $dbuser, $dbpass, $dbname);

if ($connection->connect_errno) {
    echo "Errno: " . $connection->errno . "" . ":" . $connection->error . "\n";
    exit;
}

$sql = "SELECT id, name, price FROM fruit WHERE id = '{$_id}'";

if (!$result = $connection->query($sql)) {
    echo "Query: " . $sql . "\n";
    echo "Errno: " . $connection->errno . "" . ":" . $connection->error . "\n";
    exit;
}

if ($result->num_rows === 0) {
    echo "We could not find a match for ID $_id, sorry about that. Please try again.";
    exit;
}

$fruit = $result->fetch_assoc();

$result->free();
$connection->close();
```



```php
    echo "Errno: " . $connection->errno . "" . ":" . $connection->error . "\n";
    exit;
}

$sql = "SELECT id, name, price FROM fruit WHERE id = ?";

if (!$stmt = $connection->prepare($sql)) {
    echo "Query: " . $sql . "\n";
    echo "Errno: " . $connection->errno . "" . ":" . $connection->error . "\n";
    exit;
}

// https://www.php.net/manual/en/mysqli-stmt.bind-param.php
if (!$stmt->bind_param('s', $_id)) {
    echo "Binding parameters failed: (" . $stmt->errno . ") " . $stmt->error;
    exit;
}

if (!$result = $stmt->execute()) {
    echo "Execute failed: (" . $stmt->errno . ") " . $stmt->error;
    exit;
}

if (!$result = $stmt->get_result()) {
    echo "Errno: " . $connection->errno . "" . ":" . $connection->error . "\n";
    exit;
}

if ($result->num_rows === 0) {
    echo "We could not find a match for ID $_id, sorry about that. Please try again.";
    exit;
}

$fruit = $result->fetch_assoc();

$result->free();
$connection->close();

print_r($fruit);
```



```php
<?php

$dbhost = 'localhost';
$dbname = 'fruitshop_db';
$dbuser = 'fsuser';
$dbpass = 'fsuser0000';

$_id = '0e21b9e56243d2915bd56d09034cecf5a4e20c86a928cb63623acfb83494217c';

try {
    $connection = new PDO("mysql:host={$dbhost};dbname={$dbname}", $dbuser, $dbpass);
    // when something wrong happens, throw an exception
    $connection->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

    $sql = "SELECT id, name, price FROM fruit WHERE id = :id";

    $stmt = $connection->prepare($sql);
    $stmt->bindParam(':id', $_id);

    $stmt->execute();

    // https://www.php.net/manual/en/pdostatement.fetch.php
    // mode: PDO::FETCH_ASSOC, PDO::FETCH_NUM, see docs.
    $fruit = $stmt->fetchAll();

    print_r($fruit);
  } catch(PDOException $e) {
    echo "Error: " . $e->getMessage();
  }

  $connection = null;

```

To-dos

1. Implement database connectivity into your project.
   - Always use PDO whenever possible.
2. Some questions:
   - Is it possible to have multiple connections to different database vendors in a single file?
   - And, could you read tuples from one of the two and write them to the other?
3. ORM will be discussed in another time.

References

Srinivasan, M. (2012). Web Technology: Theory and Practice. Pearson.

Tatroe, K., et. al. (2020). Programming PHP. O'Reilly.

PHP Manual https://www.php.net/manual/en/

Thank you