

# PHP/MySQL development

## a Twitter-like application

Maxime Martineau

Polytech Tours Département Informatique

December 15, 2016

# Outline

The project

PHP

PHP and MySQL

Work to do

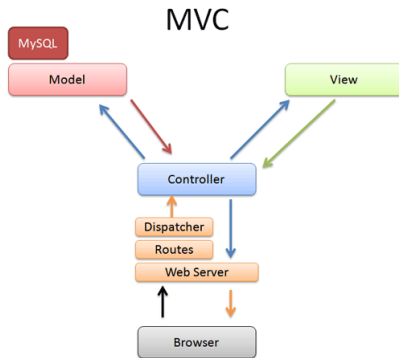
## Twitter application

- ▶ Users : can post, follow a user, ...
- ▶ Post : message from a user
- ▶ Notification system
- ▶ Hashtag : to mark a given topic (ex: #DevelopersBeLike)

# Frame

- ▶ 6 practical works (12h)
- ▶ Work in pairs
- ▶ Code to give back at the end (the database schema + the model/ files)

# Project structure



`http://github.com/prafiny/db-project`

The screenshot shows the GitHub interface for the repository `prafiny/db-project`. At the top, there's a navigation bar with links for 'This repository', 'Search', 'Pull requests', 'Issues', and 'Gist'. Below this, the repository name 'prafiny / db-project' is displayed, along with statistics: 1 Unwatch, 0 Stars, and 0 Forks. A secondary navigation bar includes links for 'Code', 'Issues 0', 'Pull requests 0', 'Projects 0', 'Wiki', 'Pulse', 'Graphs', and 'Settings'. A message states 'No description or website provided. — Edit'. Below this, a summary bar shows '57 commits', '1 branch', '0 releases', and '1 contributor'. The main section features a 'Branch: master' dropdown, a 'New pull request' button, and a row of buttons: 'Create new file', 'Upload files', 'Find file', and 'Clone or download'. A table lists recent commits by 'Maxime Martineau', including files like `config`, `controller`, `instructions`, `lib`, `model`, `tests`, `view`, `www`, `.gitignore`, and `README.md`, with their commit messages and timestamps.

File	Commit Message	Time Ago
config	db, autoload and conf + fixes	a day ago
controller	unfollow and follow	16 hours ago
instructions	instructions 1 + gitignore	14 hours ago
lib	fixes	21 hours ago
model	test and model fixes	16 hours ago
tests	test and model fixes	16 hours ago
view	unfollow and follow	16 hours ago
www	unfollow and follow	16 hours ago
.gitignore	instructions 1 + gitignore	14 hours ago
README.md	Update readme	15 hours ago

# PHP : Data

## ► Declare a variable

---

```
$a = 23; // $a is an int  
$b = 2.3; // $b is a float  
$c = true; // $c is a boolean  
$d = array("a", 1, "b"); // $d is an array
```

---

# PHP: Data

## ► Deal with arrays

---

```
$a = array("a", "b", 1);  
echo $a[0]; // prints "a"  
$a[0] = "A";  
echo $a[0]; // ?  
$a[] = 2; // $a == ?
```

---

## ► Associative arrays

---

```
$fruit = array("name" => "strawberry", "color" => "red");  
echo $fruit["name"]; // ?  
echo $fruit["color"]; // ?  
$fruit["color"] = "black";
```

---



# PHP : Data

- Create and use a stdClass object

---

```
$fruit = (object) array("name" => "strawberry", "color" =>
    "red");
echo $fruit->name; // ?
echo $fruit->color; // ?
```

---

# PHP : Controls

## ► If stanzas

---

```
if($a == 1) {  
    echo "yooohoo :D";  
}  
else {  
    echo "oh :(";  
}
```

---

# PHP : Controls

## ► While loops

---

```
$i = 0;
while($i <= 10) {
    echo $i;
    $i++;
}
```

---

## ► For loops

---

```
for($i = 0; $i <= 10; $i++) {
    echo $i;
}
```

---

## ► Foreach loops

---

```
$arr = array(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10);
foreach($arr as $i) {
    echo $i;
}
```

---

# PHP : Functions

- ▶ Create a function

---

```
function inverse($nb) {  
    return 1/$nb;  
}
```

---

- ▶ Use a function

---

```
echo inverse(3); // ?
```

---

# PHP : Error handling with exceptions

## ► Throw an exception

---

```
function inverse($nb) {  
    if($nb == 0) { throw new Exception('Division by  
        zero.');
```

  
 return 1/\$nb;  
 }  
 echo inverse(0);  
 echo "I can has display ?"; // ?  
}

---

# PHP : Error handling with exceptions

## ► Throw an exception

---

```
function inverse($nb) {  
    if($nb == 0) { throw new Exception('Division by  
        zero.');
```

  
 return 1/\$nb;  
}  
echo inverse(0);  
echo "I can has display ?"; // ?

---

## ► Catch an exception

---

```
try {  
    echo inverse(0);  
}  
catch(Exception $e) {  
    echo $e->getMessage();  
}  
echo "I can has display ?"; // ?
```

---

# PHP and MySQL : PDO

## PHP Data Object

### ► Connection

---

```
$db = new PDO('mysql:host=SERVER_ADDR;dbname=DB_DBNAME',  
              DB_USER, DB_PASS);
```

---

### ► Queries

---

```
$sql = 'SELECT name, color, calories FROM fruit ORDER BY  
        name';  
foreach ($db->query($sql) as $row) {  
    print $row['name'] . "\t";  
    print $row['color'] . "\t";  
    print $row['calories'] . "\n";  
}
```

---

# PHP and MySQL : PDO

## PHP Data Object

- Prepare query (with variables)

---

```
$sql = 'SELECT name, colour, calories FROM fruit WHERE
        calories < :calories AND colour = :colour';
$sth = $dbh->prepare($sql);
$sth->execute(array(':calories' => 150, ':colour' =>
    'red'));
foreach($sth->fetchAll() as $row) {
    print $row['name'] . "\t";
    print $row['color'] . "\t";
    print $row['calories'] . "\n";
}
```

---



# PHP and MySQL : PDO

## ► Error handling (with Exceptions)

---

```
try {  
    $sql = 'SELECT name, color, calories FROM fruit ORDER  
           BY name';  
    foreach ($db->query($sql) as $row){  
        print $row['name'] . "\t";  
        print $row['color'] . "\t";  
        print $row['calories'] . "\n";  
    }  
} catch (\PDOException $e) {  
    print $e->getMessage();  
}
```

---

# Code

`http://github.com/prafiny/db-project`

1. Application can be run in browser (see instructions)
2. Unit tests can be run to check the functions (see instructions)
3. Instructions (in the folder `instructions/`)
4. The goal is to fill out the models (files in folder `model/`)
  - 4.1 Read `0setup.pdf` for environment installation
  - 4.2 Read `1user.pdf` for first work