

## Lab exercise 3 (TDD)

- Write a test suite for testing User class:
  - Test setup: initialize \$user as instance of User, with some initial name and email.
  - \$user->name() should retrieve name.
  - \$user->name('Samy') should change user's name to Samy.
  - \$user->email() should retrieve email.
  - \$user->email('samy@gmail.com') should change user's email to samy@gmail.com.
  - Test teardown: \$user->delete();
- Run the suite and write the code to make it all pass.

```
<phpunit
bootstrap="bootstrap.php">
  <testsuites>
    <testsuite name="Sample">
      <directory>tests</directory>
    </testsuite>
  </testsuites>
</phpunit>
```

```
<?php
// my app includes
spl_autoload_register(function($class)
{
    @include_once(__DIR__ .
"/classes/{$class}.php");
});
?>
```

```
phpunit.xml
1 <!--
2 * Unit & Automated Testing - Lab 2 Exercise 3
3 * Author: Mahmoud Mohamed Kamal
4 * Date: 28-03-1998
5 * File: phpunit.xml
6 -->
7
8 <phpunit bootstrap="bootstrap.php">
9   <testsuites>
10     <testsuite name="Sample">
11       <directory>tests</directory>
12     </testsuite>
13   </testsuites>
14 </phpunit>
```

```
bootstrap.php
1 /*
2 * Unit & Automated Testing - Lab 2 Exercise 3
3 * Author: Mahmoud Mohamed Kamal
4 * Date: 28-03-1998
5 * File: bootstrap.php
6 */
7
8 <?php
9 // my app includes
10 spl_autoload_register(function($class) {
11     @include_once(__DIR__ . "/classes/{$class}.php");
12 });
13 ?>
```

```
<?php
class SampleTest extends PHPUnit\Framework\TestCase {
    private static $user;

    public static function setUpBeforeClass() : void{
        self::$user = new User('Mahmoud','mahmoud@email.com');
    }

    public function test_name(){
        $this->assertTrue(self::$user->getName()=='Mahmoud');
    }

    public function test_email(){
        $this->assertTrue(self::$user->getEmail()=='mahmoud@email.com');
    }

    public function test_name_Samy(){
        self::$user->setName('samy');
        $this->assertTrue(self::$user->getName()=='samy');
    }

    public function test_email_Samy(){
        self::$user->setEmail('samy@gmail.com');
        $this->assertTrue(self::$user->getEmail()=='samy@gmail.com');
    }

    public static function tearDownAfterClass() : void{
        self::$user->delete();
    }
}
?>
```

\$ gedit Desktop/lab2/class/User.php &

```
<?php
class User {
    private $username, $email;

    public function __construct($username, $email) {
        $this->username = $username;
        $this->email = $email;
    }

    public function getName(){
        return $this->username;
    }

    public function getEmail(){
        return $this->email;
    }

    public function setName($n){
        $this->username = $n;
    }

    public function setEmail($e){
        $this->email = $e;
    }

    public function delete(){
        $this->username = NULL;
        $this->email = NULL;
    }
}
?>
```

```
SampleTest.php
1 /*
2  * Unit & Automated Testing - Lab 2 Exercise 3
3  * Author: Mahmoud Mohamed Kamal
4  * Date: 28-03-1998
5  * File: SampleTest.php
6  */
7
8 <?php
9 class SampleTest extends PHPUnit\Framework\TestCase {
10     private static $user;
11
12     public static function setUpBeforeClass() : void{
13         self::$user = new User('Mahmoud','mahmoud@email.com');
14     }
15     public function test_name(){
16         $this->assertTrue(self::$user->getName()=='Mahmoud');
17     }
18     public function test_email(){
19         $this->assertTrue(self::$user->getEmail()=='mahmoud@email.com');
20     }
21     public function test_name_Samy(){
22         self::$user->setName('samy');
23         $this->assertTrue(self::$user->getName()=='samy');
24     }
25     public function test_email_Samy(){
26         self::$user->setEmail('samy@gmail.com');
27         $this->assertTrue(self::$user->getEmail()=='samy@gmail.com');
28     }
29     public static function tearDownAfterClass() : void{
30         self::$user->delete();
31     }
32 }
33 ?>
```

```
User.php
1 /*
2  * Unit & Automated Testing - Lab 2 Exercise 3
3  * Author: Mahmoud Mohamed Kamal
4  * Date: 28-03-1998
5  * File: User.php
6  */
7
8 <?php
9 class User {
10     private $username, $email;
11
12     public function __construct($username, $email) {
13         $this->username = $username;
14         $this->email = $email;
15     }
16
17     public function getName(){
18         return $this->username;
19     }
20
21     public function getEmail(){
22         return $this->email;
23     }
24
25     public function setName($n){
26         $this->username = $n;
27     }
28
29     public function setEmail($e){
30         $this->email = $e;
31     }
32
33     public function delete(){
34         $this->username = NULL;
35         $this->email = NULL;
36     }
37 }
38 ?>
```

\$ ~/phpunit --testdox

```
/*
 * Unit & Automated Testing - Lab 2 Exercise 3
 * Author: Mahmoud Mohamed Kamal
 * Date: 28-03-1998
 * File: User.php
 */

Sample
✓ Name
✓ Email
✓ Name Samy
✓ Email Samy

Time: 00:00.016, Memory: 18.00 MB

OK (4 tests, 4 assertions)
```

# Lab exercise 4

- Write a test suite to test a class from your most recent PHP assignment 😊

```
<phpunit bootstrap="bootstrap.php">
<testsuites>
  <testsuite name="Sample">
    <directory>tests</directory>
  </testsuite>
</testsuites>
</phpunit>
```

```
<?php
// my app includes
spl_autoload_register(function($class) {
    @include_once(__DIR__ .
"/classes/{ $class }.php");
});
?>
```

```
<?php
class SampleTest extends PHPUnit\Framework\TestCase {
    private static $user;

    public static function setUpBeforeClass() : void{
        self::$user = new User('Mahmoud', 1234567890,
'mahmoud@email.com');
    }

    public function test_name(){
        $this->assertTrue(self::$user->
getUsername()=='Mahmoud');
    }

    public function test_password(){
        $this->assertTrue(self::$user->getPassword()===1234567890);
    }

    public function test_email(){
        $this->assertTrue(self::$user->getEmail()=='mahmoud@email.com');
    }

    public function test_name_Kamal(){
        self::$user->setUsername('Kamal');
        $this->assertTrue(self::$user->getUsername()=='Kamal');
    }

    public function test_password_Kamal(){
        self::$user->setPassword(1234);
        $this->assertTrue(self::$user->getPassword()===1234);
    }

    public function test_email_Kamal(){
        self::$user->setEmail('kamal@email.com');
        $this->assertTrue(self::$user->getEmail()=='kamal@email.com');
    }

    public static function tearDownAfterClass() : void{
        self::$user->delete();
    }
}
?>
```

```
phpunit.xml
1 <!--
2 * Unit & Automated Testing - Lab 2 Exercise 4
3 * Author: Mahmoud Mohamed Kamal
4 * Date: 28-03-1998
5 * File: phpunit.xml
6 -->
7
8 <phpunit bootstrap="bootstrap.php">
9 <testsuites>
10 <testsuite name="Sample">
11 <directory>tests</directory>
12 </testsuite>
13 </testsuites>
14 </phpunit>
```

```
bootstrap.php
1 /*
2 * Unit & Automated Testing - Lab 2 Exercise 4
3 * Author: Mahmoud Mohamed Kamal
4 * Date: 28-03-1998
5 * File: bootstrap.php
6 */
7
8 <?php
9 // my app includes
10 spl_autoload_register(function($class) {
11     @include_once(__DIR__ . "/classes/{ $class }.php");
12 });
13 ?>
```

```
SampleTest.php
1 /*
2 * Unit & Automated Testing - Lab 2 Exercise 4
3 * Author: Mahmoud Mohamed Kamal
4 * Date: 28-03-1998
5 * File: SampleTest.php
6 */
7
8 <?php
9 class SampleTest extends PHPUnit\Framework\TestCase {
10     private static $user;
11
12     public static function setUpBeforeClass() : void{
13         self::$user = new User('Mahmoud', 1234567890, 'mahmoud@email.com');
14     }
15     public function test_name(){
16         $this->assertTrue(self::$user->getUsername()=='Mahmoud');
17     }
18     public function test_password(){
19         $this->assertTrue(self::$user->getPassword()===1234567890);
20     }
21     public function test_email(){
22         $this->assertTrue(self::$user->getEmail()=='mahmoud@email.com');
23     }
24     public function test_name_Kamal(){
25         self::$user->setUsername('Kamal');
26         $this->assertTrue(self::$user->getUsername()=='Kamal');
27     }
28     public function test_password_Kamal(){
29         self::$user->setPassword(1234);
30         $this->assertTrue(self::$user->getPassword()===1234);
31     }
32     public function test_email_Kamal(){
33         self::$user->setEmail('kamal@email.com');
34         $this->assertTrue(self::$user->getEmail()=='kamal@email.com');
35     }
36     public static function tearDownAfterClass() : void{
37         self::$user->delete();
38     }
39 }
40 ?>
```

```

<?php
class User
{
    private $username, $password, $email;

    public function __construct($username, $password, $email)
    {
        $this->setEmail($email);
        $this->setUsername($username);
        $this->setPassword($password);
    }

    function getUsername()
    {
        return $this->username;
    }

    function getPassword()
    {
        return $this->password;
    }

    function getEmail()
    {
        return $this->email;
    }

    function setUsername($username)
    {
        $this->username = $username;
    }

    function setPassword($password)
    {
        $this->password = $password;
    }

    function setEmail($email)
    {
        $this->email = $email;
    }

    function delete()
    {
        $this->setEmail(NULL);
        $this->setUsername(NULL);
        $this->setPassword(NULL);
    }
}

```

```

User.php
1 <!--
2 Lab 3
3 Author: Mahmoud Mohamed Kamal
4 Date: 26-02-2022
5 File: User.php
6 -->
7
8 <?php
9 class User
10 {
11     private $username, $password, $email;
12
13     public function __construct($username, $password, $email)
14     {
15         $this->setEmail($email);
16         $this->setUsername($username);
17         $this->setPassword($password);
18     }
19
20     function getUsername()
21     {
22         return $this->username;
23     }
24
25     function getPassword()
26     {
27         return $this->password;
28     }
29
30     function getEmail()
31     {
32         return $this->email;
33     }
34
35     function setUsername($username)
36     {
37         $this->username = $username;
38     }
39
40     function setPassword($password)
41     {
42         $this->password = $password;

```

\$ ~/phpunit --testdox

```

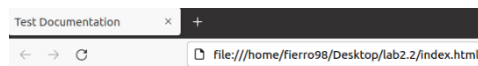
Sample
✓ Name
✓ Password
✓ Email
✓ Name Kamal
✓ Password Kamal
✓ Email Kamal

Time: 00:00.006, Memory: 18.00 MB

OK (6 tests, 6 assertions)

```

\$ ~/phpunit --testdox-html index.html



## Sample

- ✓ Name
- ✓ Password
- ✓ Email
- ✓ Name Kamal
- ✓ Password Kamal
- ✓ Email Kamal

\$ ~/phpunit --testdox-text index.txt

	index.html
1	Sample
2	[x] Name
3	[x] Password
4	[x] Email
5	[x] Name Kamal
6	[x] Password Kamal
7	[x] Email Kamal

\$ ~/phpunit --testdox-xml index.xml

index.xml	index.html	index.txt
<pre>1 &lt;?xml version="1.0" encoding="UTF-8"?&gt; 2 &lt;tests&gt; 3   &lt;test className="SampleTest" methodName="test_name" prettifiedClassName="Sample" prettifiedMethodName="Name" status="0" tne="0.000415496" size="-1" groups="default"&gt; 4     &lt;group name="default"/&gt; 5   &lt;/test&gt; 6   &lt;test className="SampleTest" methodName="test_password" prettifiedClassName="Sample" prettifiedMethodName="Password" status="0" tne="0.000180646" size="-1" groups="default"&gt; 7     &lt;group name="default"/&gt; 8   &lt;/test&gt; 9   &lt;test className="SampleTest" methodName="test_email" prettifiedClassName="Sample" prettifiedMethodName="Email" status="0" tne="0.000661703" size="-1" groups="default"&gt; 10    &lt;group name="default"/&gt; 11  &lt;/test&gt; 12  &lt;test className="SampleTest" methodName="test_name_Kamal" prettifiedClassName="Sample" prettifiedMethodName="Name Kamal" status="0" tne="0.000144162" size="-1" groups="default"&gt; 13    &lt;group name="default"/&gt; 14  &lt;/test&gt; 15  &lt;test className="SampleTest" methodName="test_password_Kamal" prettifiedClassName="Sample" prettifiedMethodName="Password Kamal" status="0" tne="0.000204525" size="-1" groups="default"&gt; 16    &lt;group name="default"/&gt; 17  &lt;/test&gt; 18  &lt;test className="SampleTest" methodName="test_email_Kamal" prettifiedClassName="Sample" prettifiedMethodName="Email Kamal" status="0" tne="0.000261289" size="-1" groups="default"&gt; 19    &lt;group name="default"/&gt; 20  &lt;/test&gt; 21 &lt;/tests&gt;</pre>		