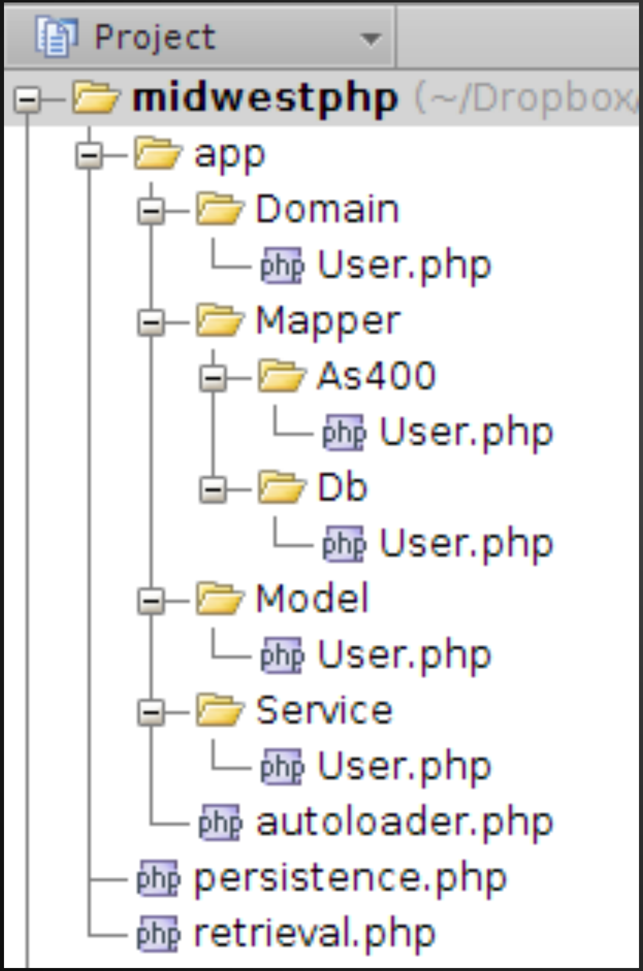
Example Code

- Let's see this in action. Things to remember:
- This is not a whole app - don't treat it as one
- It is example code
- This is not a whole app
- Don't forget, this is not a completed app

Just trying to demonstrate the concepts with the least amount of distraction!

Layout - Overview

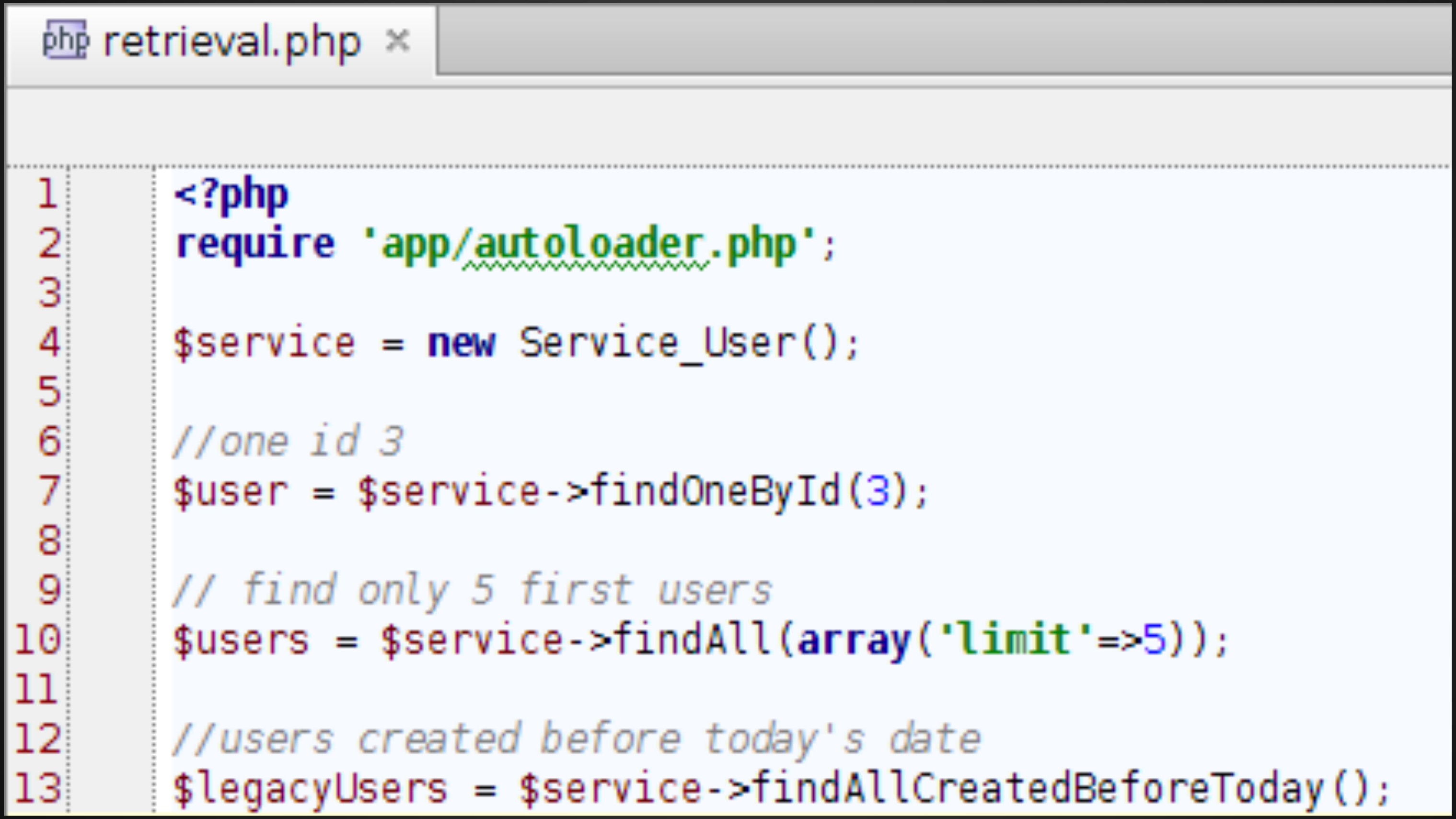


```
Project
midwestphp (~/Dropbox
  app
    Domain
      User.php
    Mapper
      As400
        User.php
      Db
        User.php
    Model
      User.php
    Service
      User.php
    autoloader.php
  persistence.php
  retrieval.php
```

Main layout of files

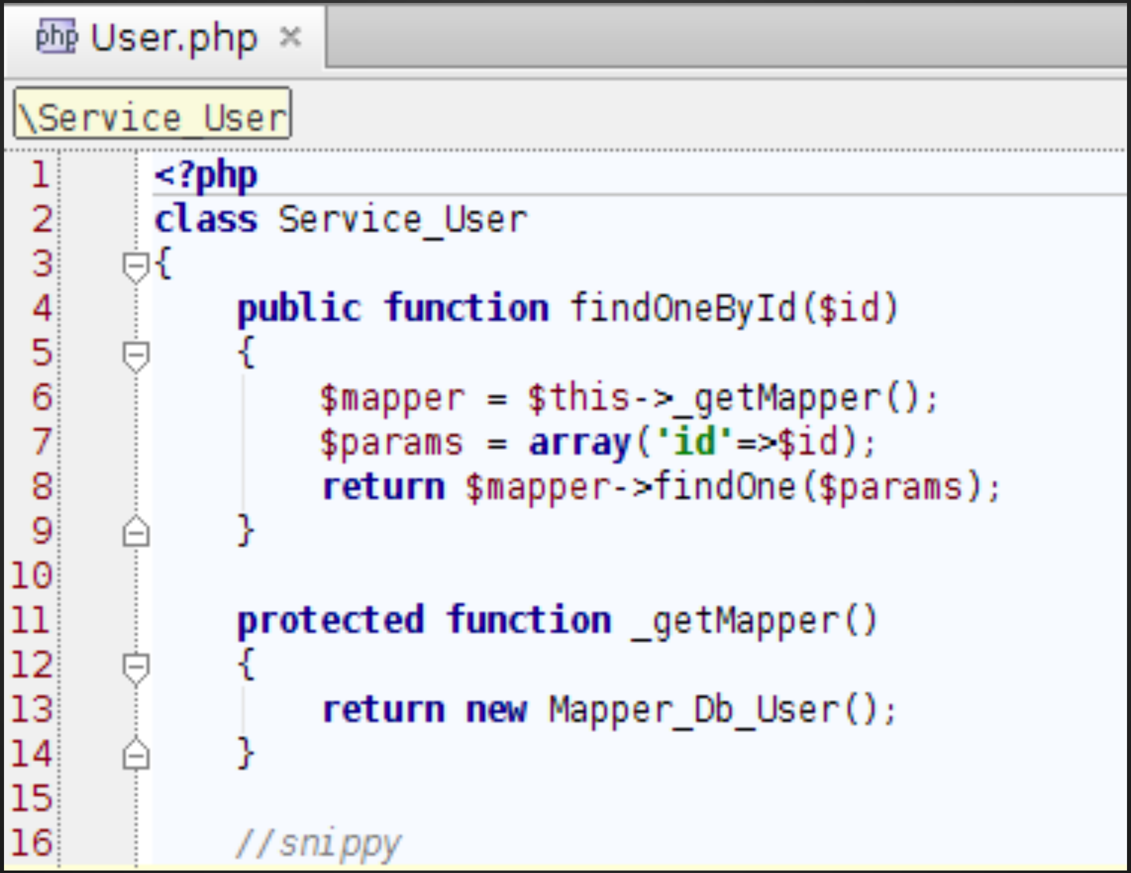
We will cover each code section

Retrieval Options (A Controller)


retrieval.php

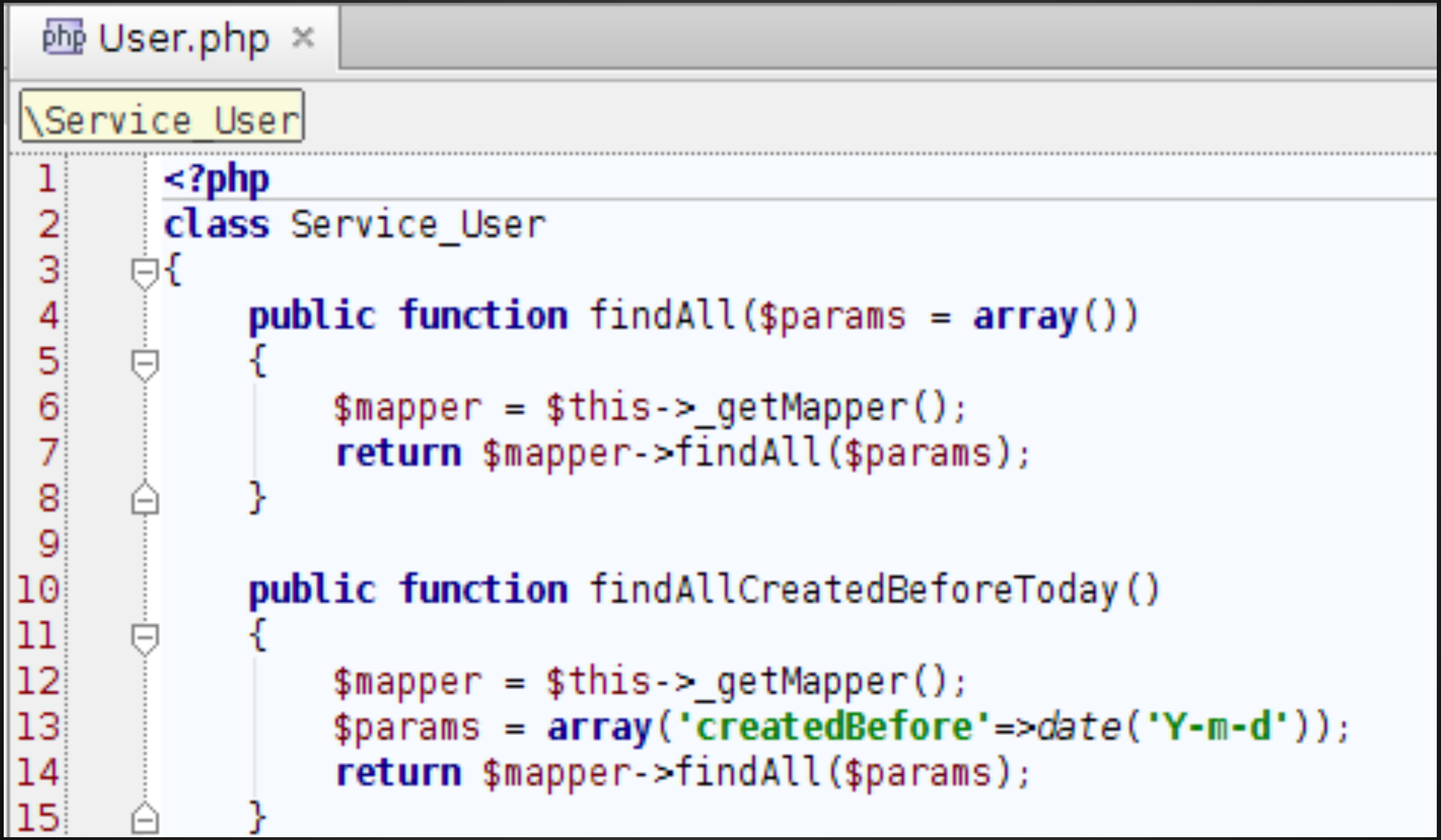
```php
<?php
require 'app/autoloader.php';

$service = new Service_User();

//one id 3
$user = $service->findOneById(3);

// find only 5 first users
$users = $service->findAll(array('limit'=>5));

//users created before today's date
$legacyUsers = $service->findAllCreatedBeforeToday();
```

Retrieve one by ID



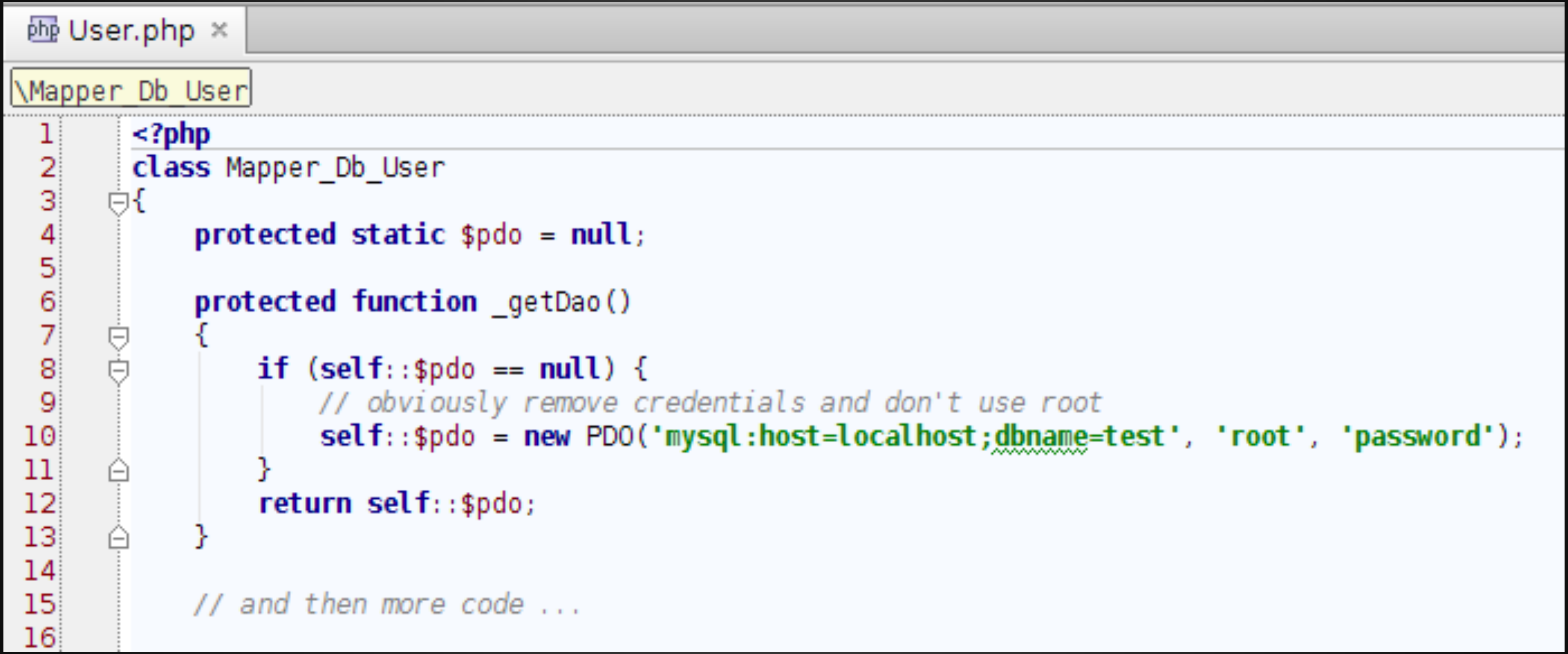
```php
<?php
class Service_User
{
    public function findOneById($id)
    {
        $mapper = $this->_getMapper();
        $params = array('id'=>$id);
        return $mapper->findOne($params);
    }

    protected function _getMapper()
    {
        return new Mapper_Db_User();
    }

    //snippy
```

Other retrieval code



```php
<?php
class Service_User
{
    public function findAll($params = array())
    {
        $mapper = $this->_getMapper();
        return $mapper->findAll($params);
    }

    public function findAllCreatedBeforeToday()
    {
        $mapper = $this->_getMapper();
        $params = array('createdBefore'=>date('Y-m-d'));
        return $mapper->findAll($params);
    }
```

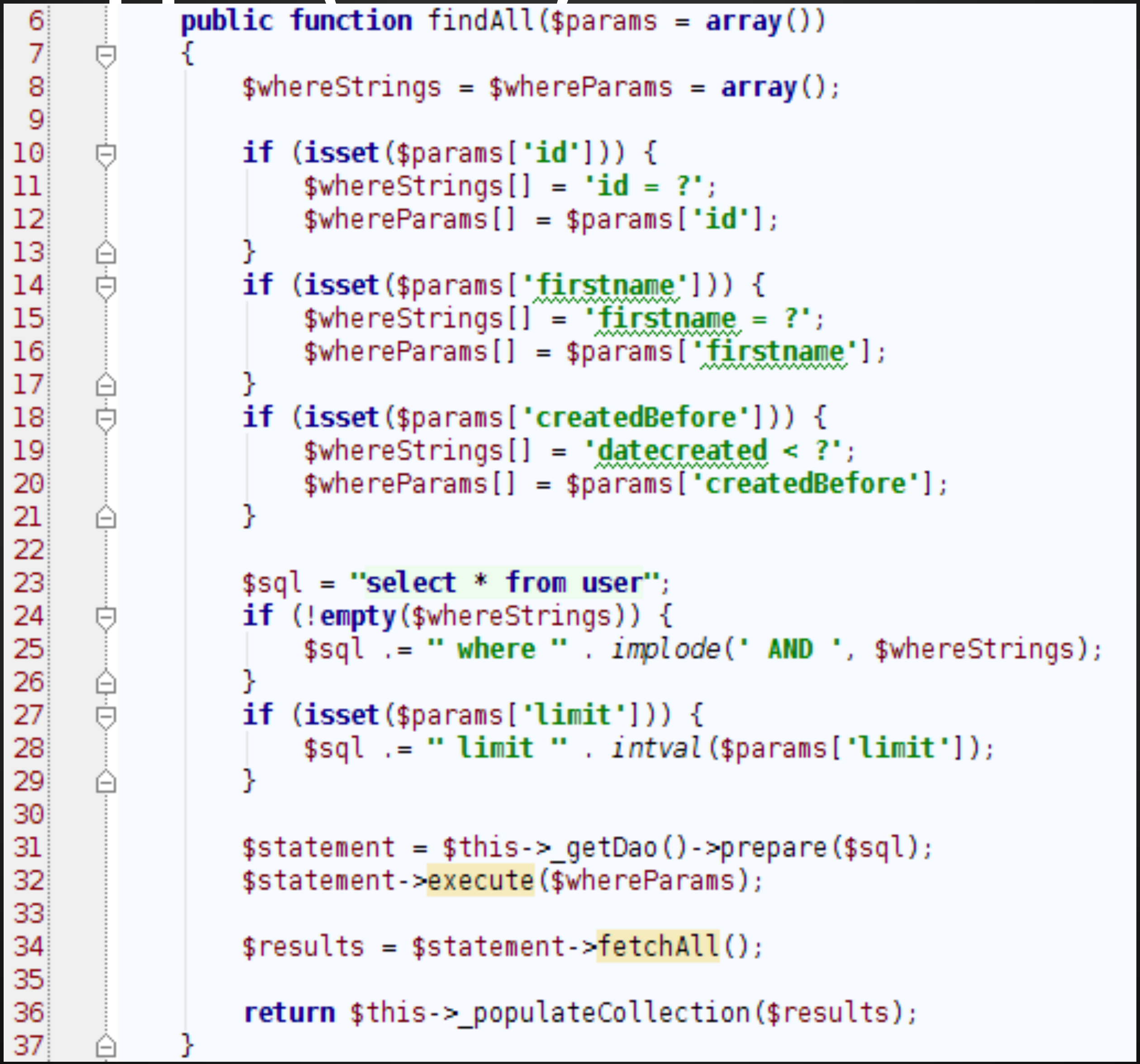
The Mapper


User.php

\Mapper_Db_User

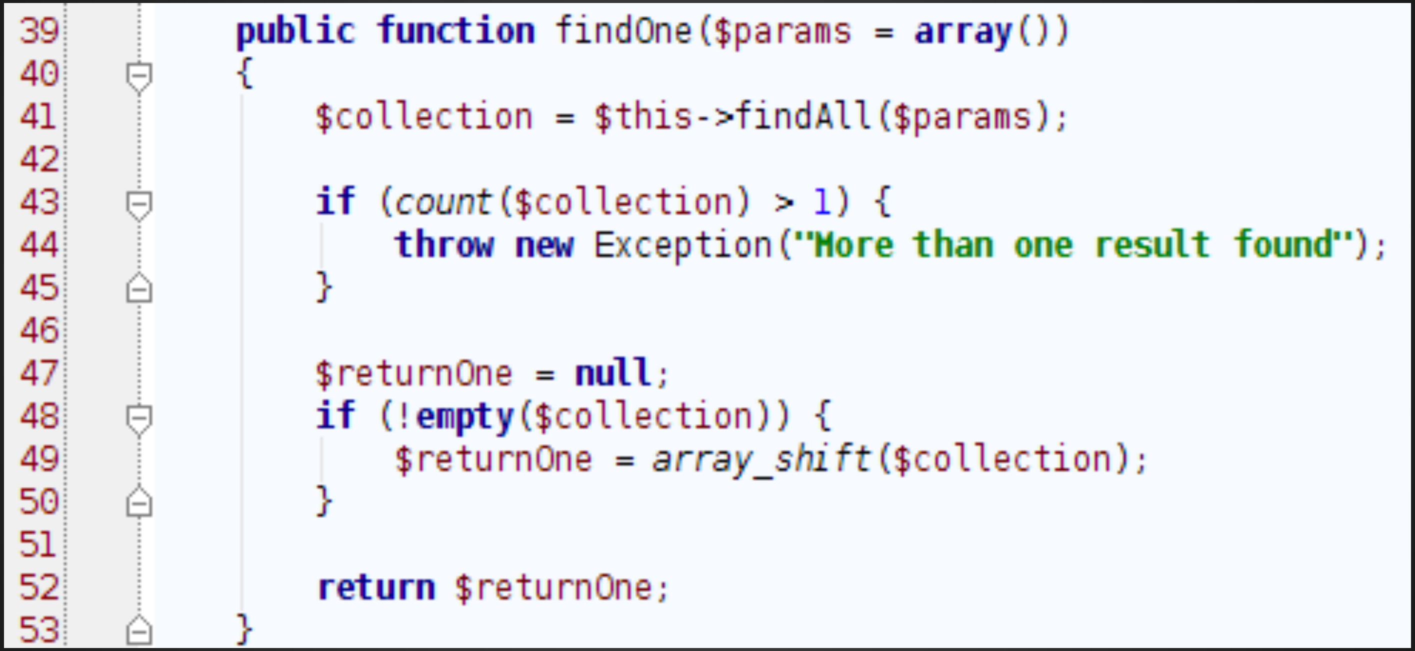
```php
<?php
class Mapper_Db_User
{
    protected static $pdo = null;

    protected function _getDao()
    {
        if (self::$pdo == null) {
            // obviously remove credentials and don't use root
            self::$pdo = new PDO('mysql:host=localhost;dbname=test', 'root', 'password');
        }
        return self::$pdo;
    }

    // and then more code ...

```

The Mapper (con't)



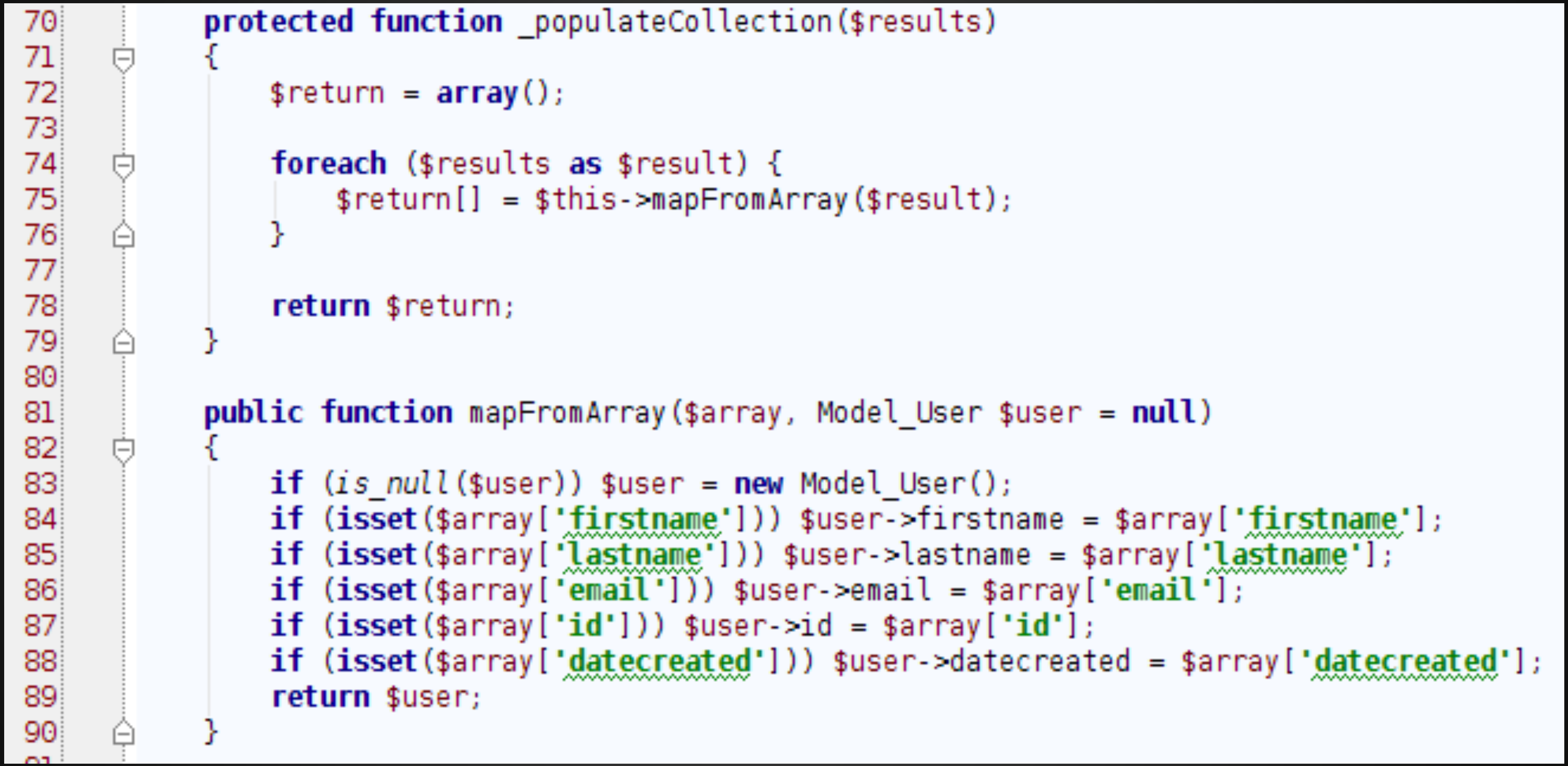
```php
    public function findAll($params = array())
    {
        $whereStrings = $whereParams = array();

        if (isset($params['id'])) {
            $whereStrings[] = 'id = ?';
            $whereParams[] = $params['id'];
        }
        if (isset($params['firstname'])) {
            $whereStrings[] = 'firstname = ?';
            $whereParams[] = $params['firstname'];
        }
        if (isset($params['createdBefore'])) {
            $whereStrings[] = 'datecreated < ?';
            $whereParams[] = $params['createdBefore'];
        }

        $sql = "select * from user";
        if (!empty($whereStrings)) {
            $sql .= " where " . implode(' AND ', $whereStrings);
        }
        if (isset($params['limit'])) {
            $sql .= " limit " . intval($params['limit']);
        }

        $statement = $this->_getDao()->prepare($sql);
        $statement->execute($whereParams);

        $results = $statement->fetchAll();

        return $this->_populateCollection($results);
    }
```

More Mapper



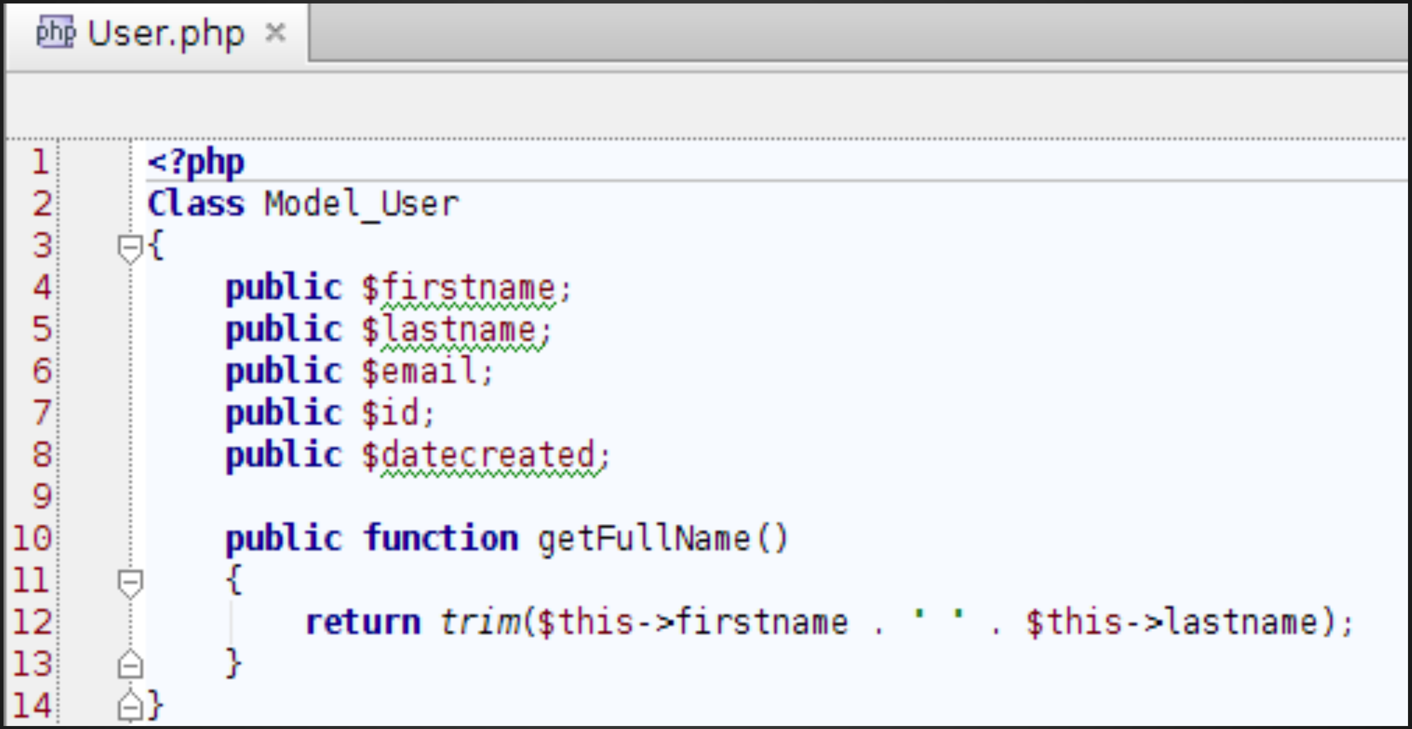
```
public function findOne($params = array())
{
    $collection = $this->findAll($params);

    if (count($collection) > 1) {
        throw new Exception("More than one result found");
    }

    $returnOne = null;
    if (!empty($collection)) {
        $returnOne = array_shift($collection);
    }

    return $returnOne;
}
```

Mapper (populating / mapping)



```
    protected function _populateCollection($results)
    {
        $return = array();

        foreach ($results as $result) {
            $return[] = $this->mapFromArray($result);
        }

        return $return;
    }

    public function mapFromArray($array, Model_User $user = null)
    {
        if (is_null($user)) $user = new Model_User();
        if (isset($array['firstname'])) $user->firstname = $array['firstname'];
        if (isset($array['lastname'])) $user->lastname = $array['lastname'];
        if (isset($array['email'])) $user->email = $array['email'];
        if (isset($array['id'])) $user->id = $array['id'];
        if (isset($array['datecreated'])) $user->datecreated = $array['datecreated'];
        return $user;
    }
```

Data Model



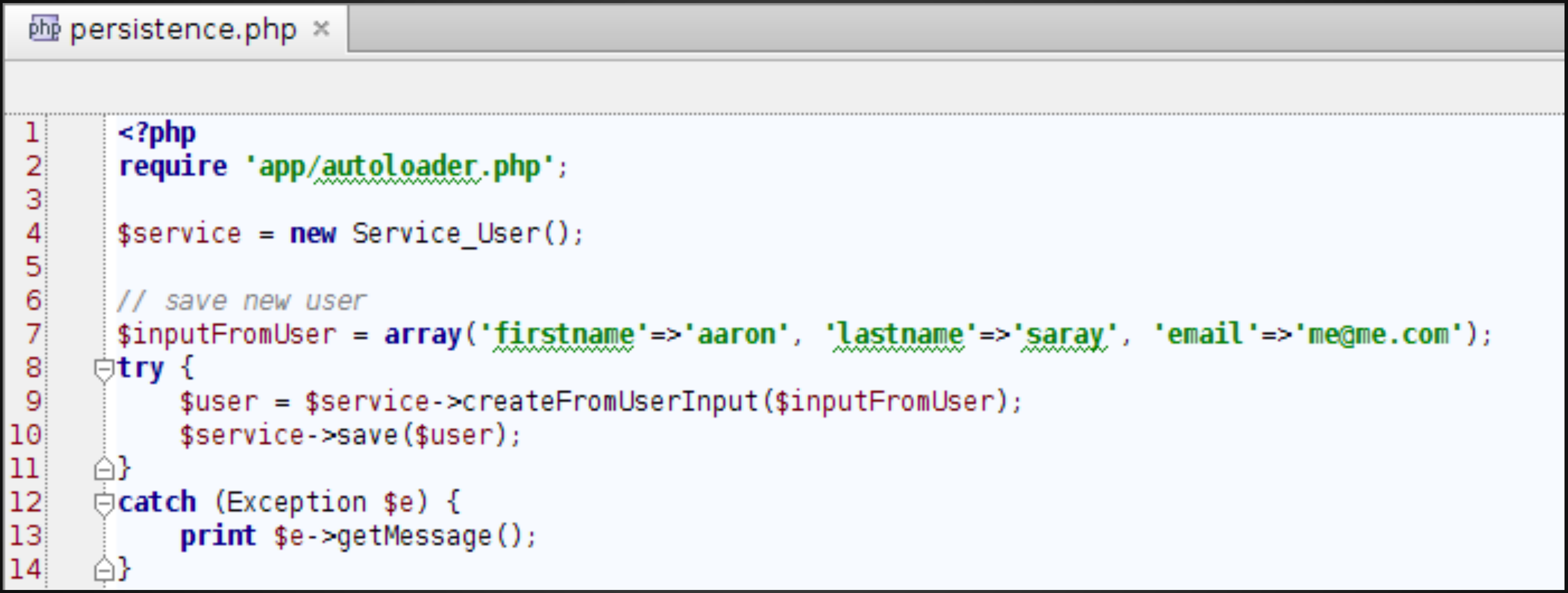
```php
<?php
Class Model_User
{
    public $firstname;
    public $lastname;
    public $email;
    public $id;
    public $datecreated;

    public function getFullName()
    {
        return trim($this->firstname . ' ' . $this->lastname);
    }
}
```

Next Up...

Retrieval is done...

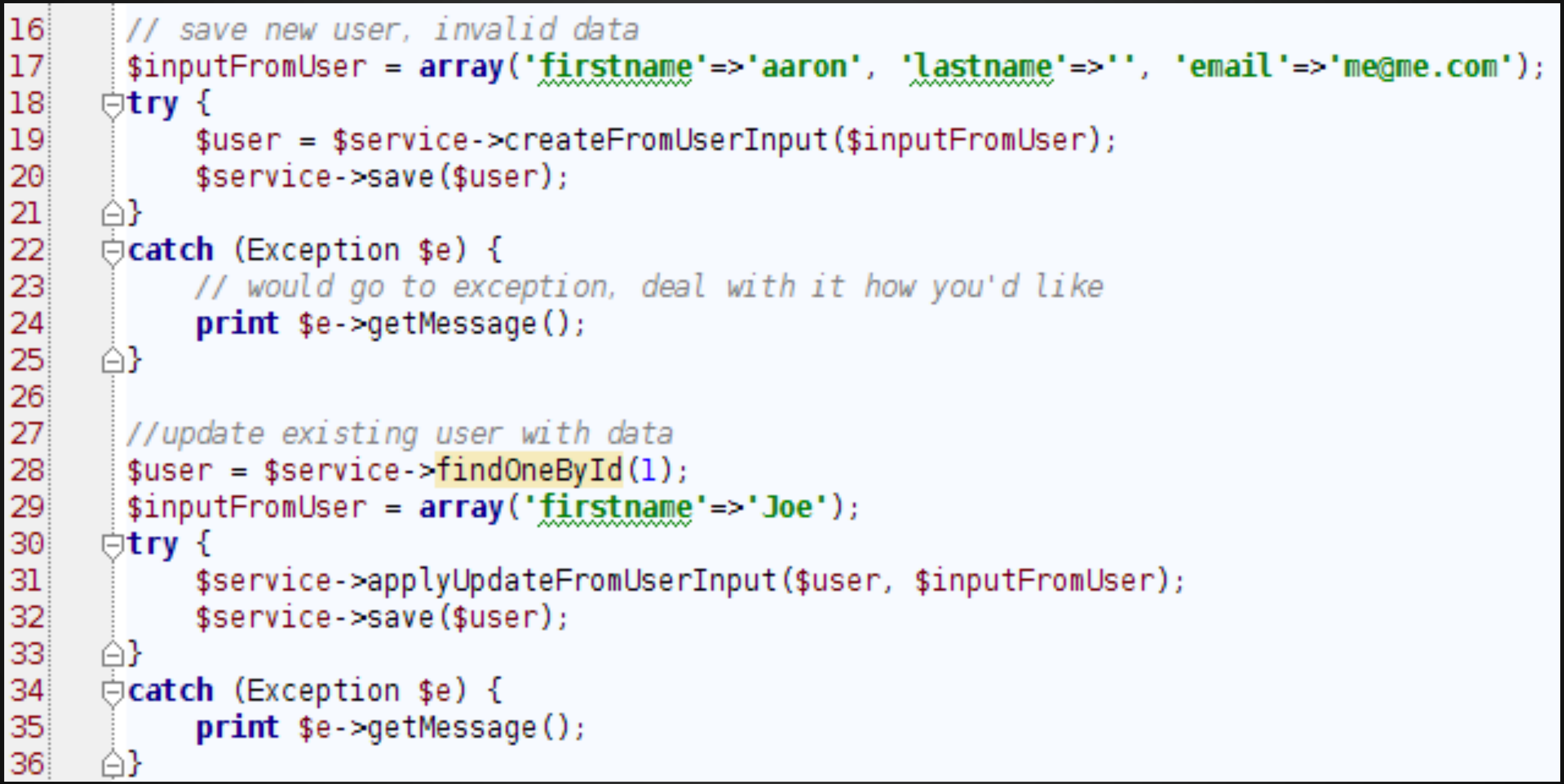
Let's look at persistence

Persistence Controller (1 of 2)


persistence.php

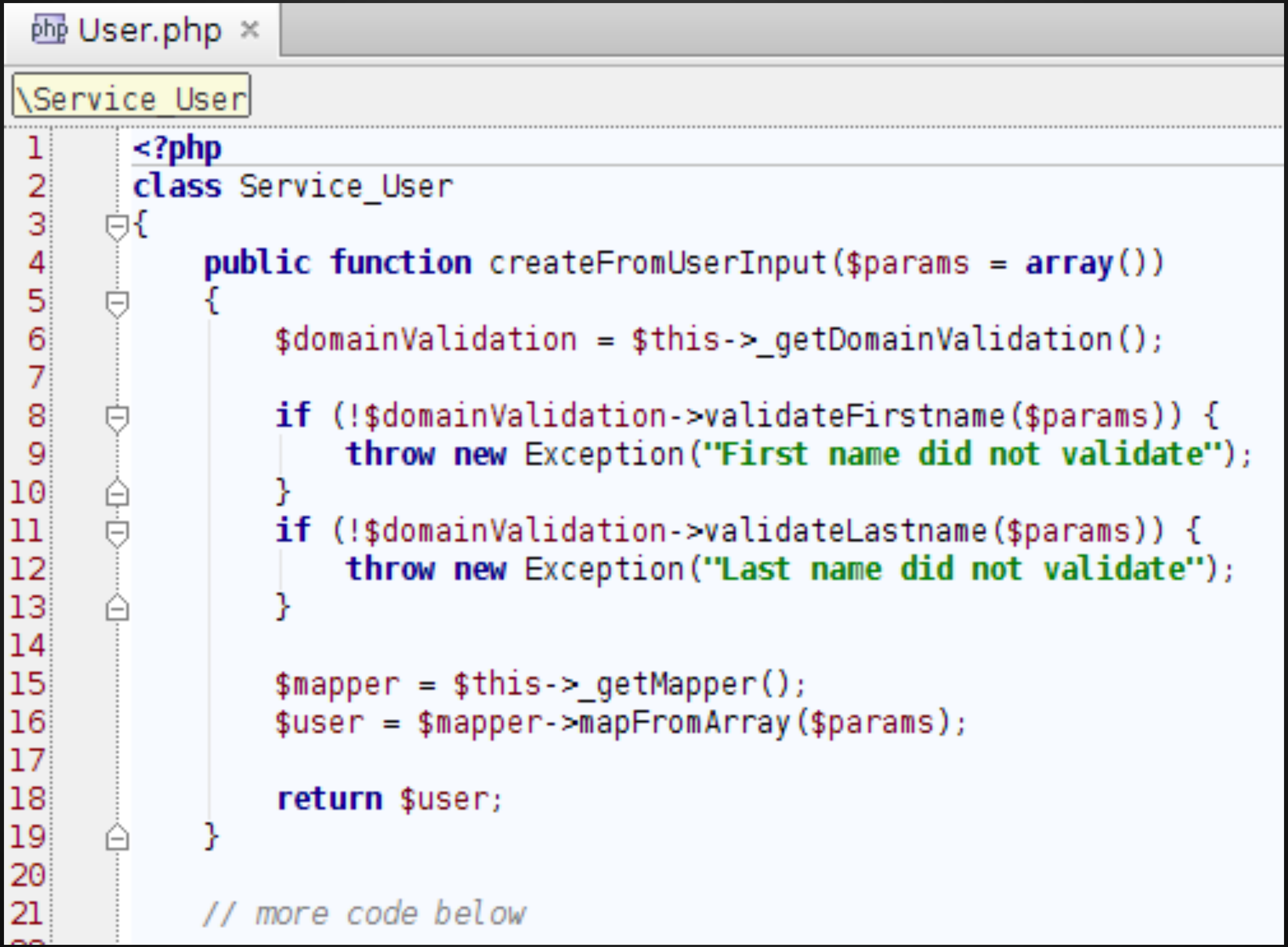
```php
<?php
require 'app/autoloader.php';

$service = new Service_User();

// save new user
$inputFromUser = array('firstname'=>'aaron', 'lastname'=>'saray', 'email'=>'me@me.com');
try {
    $user = $service->createFromUserInput($inputFromUser);
    $service->save($user);
}
catch (Exception $e) {
    print $e->getMessage();
}
```

Persistence Controller (2 of 2)



```php
// save new user, invalid data
$inputFromUser = array('firstname'=>'aaron', 'lastname'=>'', 'email'=>'me@me.com');
try {
    $user = $service->createFromUserInput($inputFromUser);
    $service->save($user);
}
catch (Exception $e) {
    // would go to exception, deal with it how you'd like
    print $e->getMessage();
}

//update existing user with data
$user = $service->findOneById(1);
$inputFromUser = array('firstname'=>'Joe');
try {
    $service->applyUpdateFromUserInput($user, $inputFromUser);
    $service->save($user);
}
catch (Exception $e) {
    print $e->getMessage();
}
```

Creating a User

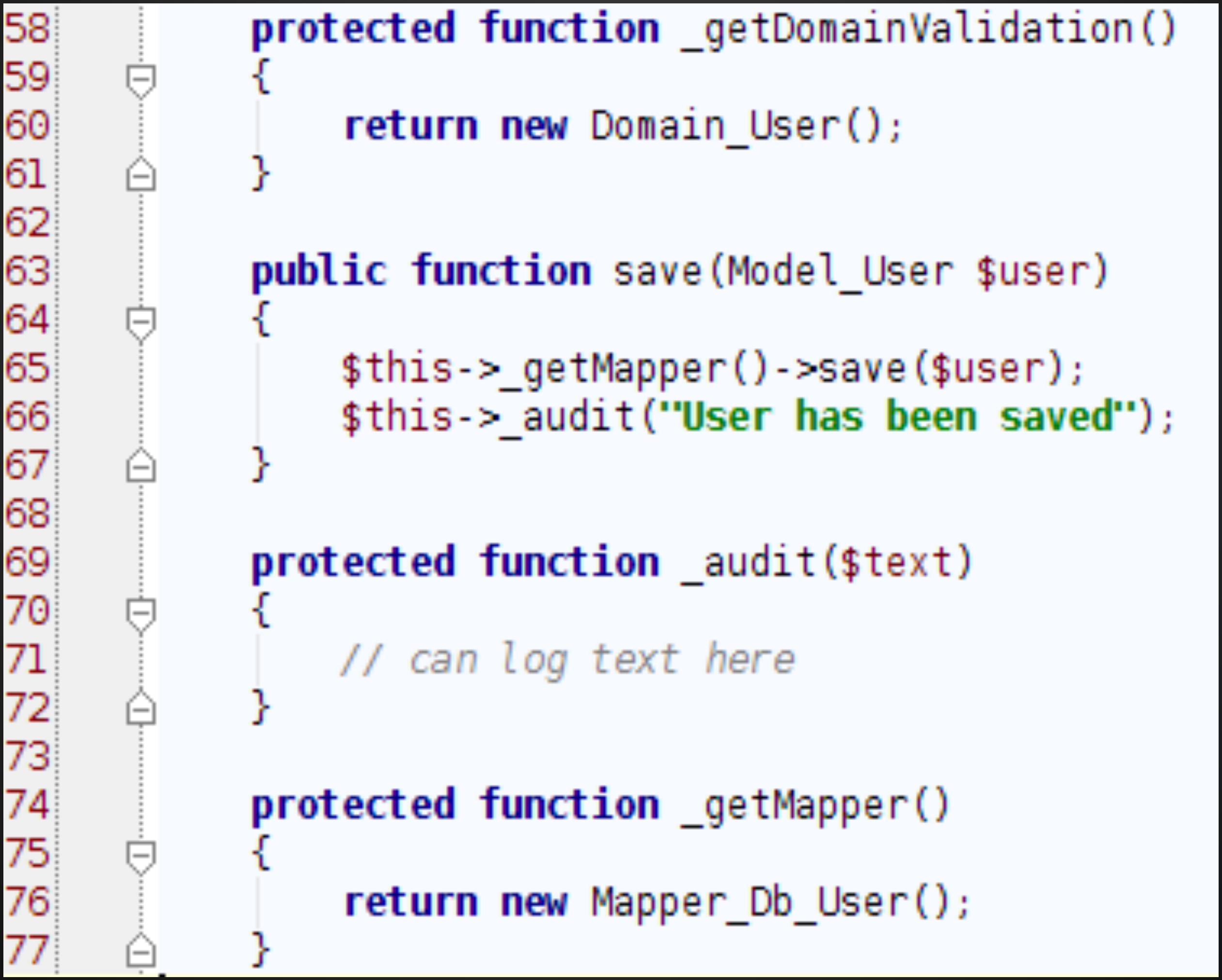


User.php

\Service_User

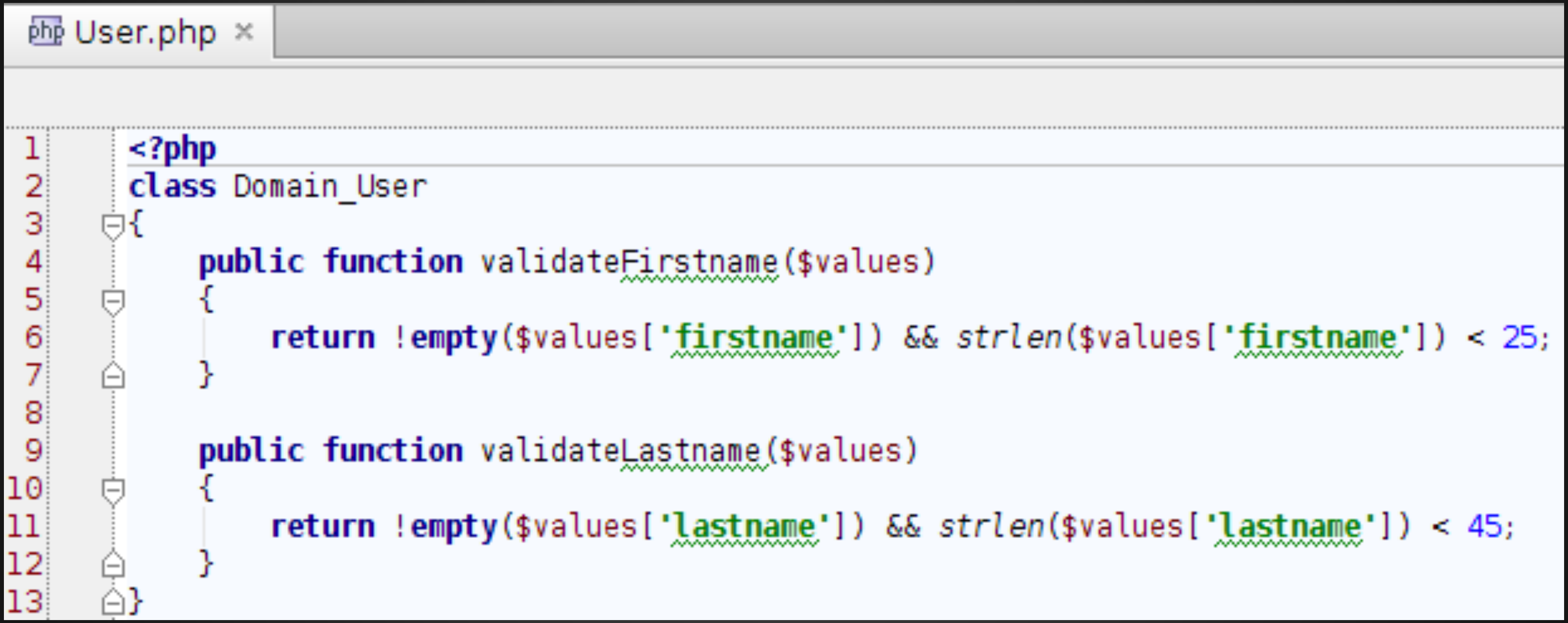
```php
<?php
class Service_User
{
    public function createFromUserInput($params = array())
    {
        $domainValidation = $this->_getDomainValidation();

        if (!$domainValidation->validateFirstname($params)) {
            throw new Exception("First name did not validate");
        }
        if (!$domainValidation->validateLastname($params)) {
            throw new Exception("Last name did not validate");
        }

        $mapper = $this->_getMapper();
        $user = $mapper->mapFromArray($params);

        return $user;
    }

    // more code below
```

Creating User (con't)



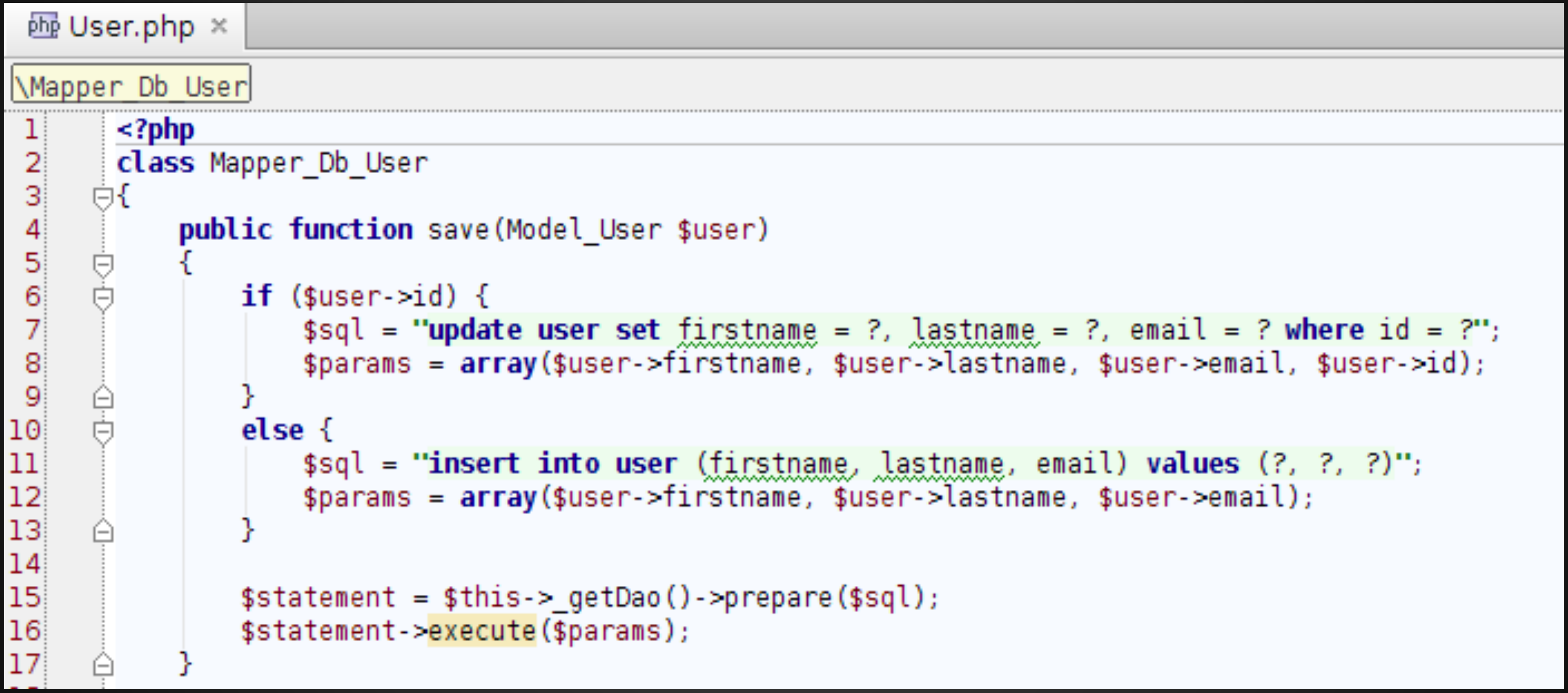
```
    protected function _getDomainValidation()
    {
        return new Domain_User();
    }

    public function save(Model_User $user)
    {
        $this->_getMapper()->save($user);
        $this->_audit("User has been saved");
    }

    protected function _audit($text)
    {
        // can log text here
    }

    protected function _getMapper()
    {
        return new Mapper_Db_User();
    }
```

Domain User Model



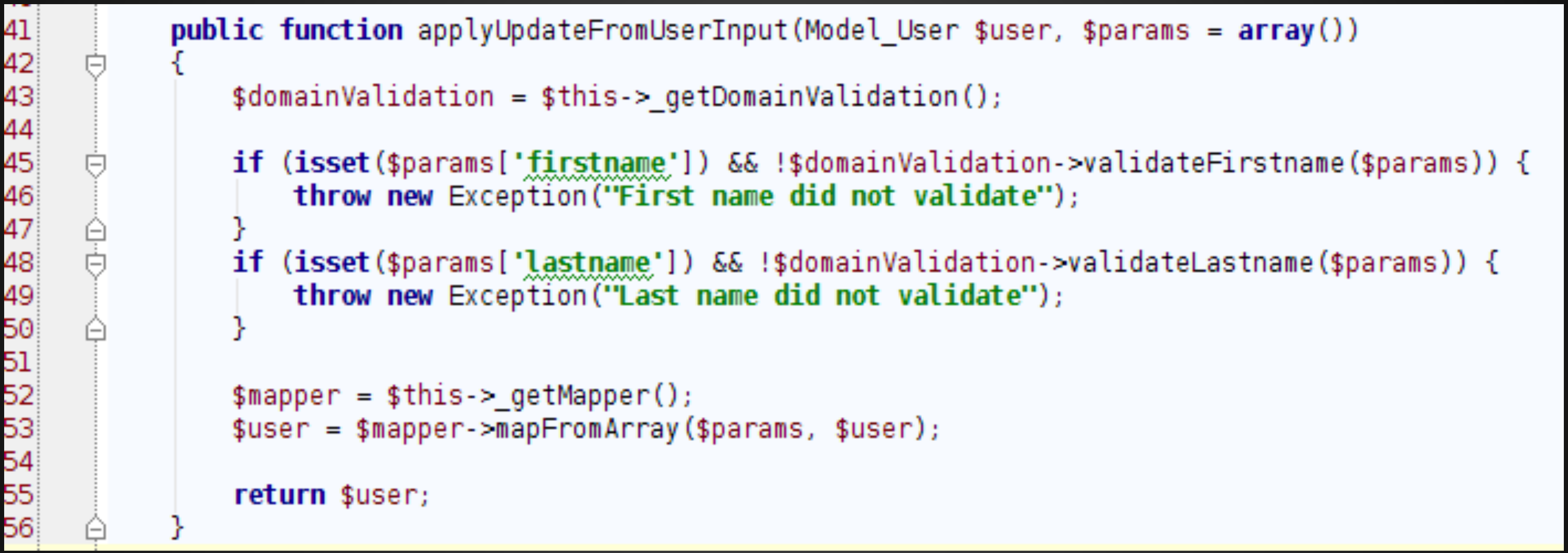
```php
<?php
class Domain_User
{
    public function validateFirstname($values)
    {
        return !empty($values['firstname']) && strlen($values['firstname']) < 25;
    }

    public function validateLastname($values)
    {
        return !empty($values['lastname']) && strlen($values['lastname']) < 45;
    }
}
```

Mapper DB - Save



```php
<?php
class Mapper_Db_User
{
    public function save(Model_User $user)
    {
        if ($user->id) {
            $sql = "update user set firstname = ?, lastname = ?, email = ? where id = ?";
            $params = array($user->firstname, $user->lastname, $user->email, $user->id);
        }
        else {
            $sql = "insert into user (firstname, lastname, email) values (?, ?, ?)";
            $params = array($user->firstname, $user->lastname, $user->email);
        }

        $statement = $this->_getDao()->prepare($sql);
        $statement->execute($params);
    }
```

Apply Update From User



```php
    public function applyUpdateFromUserInput(Model_User $user, $params = array())
    {
        $domainValidation = $this->_getDomainValidation();

        if (isset($params['firstname']) && !$domainValidation->validateFirstname($params)) {
            throw new Exception("First name did not validate");
        }
        if (isset($params['lastname']) && !$domainValidation->validateLastname($params)) {
            throw new Exception("Last name did not validate");
        }

        $mapper = $this->_getMapper();
        $user = $mapper->mapFromArray($params, $user);

        return $user;
    }
```