

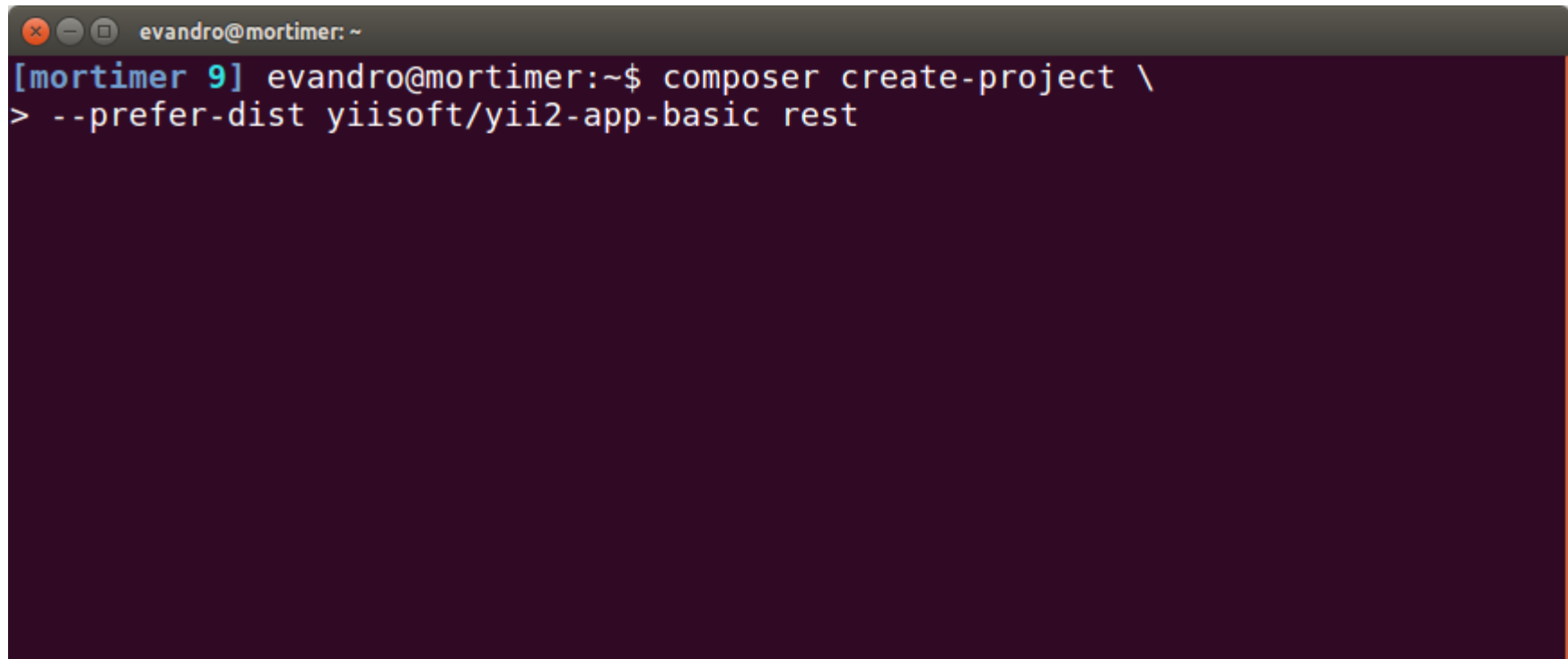


Testes e Qualidade de Software

CteSP Desenvolvimento de Software

Protótipo Yii2 REST

- Criar uma aplicação Yii2 usando o template basic

A terminal window with a dark background and light-colored text. The window title bar shows 'evandro@mortimer: ~'. The terminal content shows a command prompt '[mortimer 9] evandro@mortimer:~\$' followed by the command 'composer create-project \> --prefer-dist yiisoft/yii2-app-basic rest'.

```
evandro@mortimer: ~  
[mortimer 9] evandro@mortimer:~$ composer create-project \  
> --prefer-dist yiisoft/yii2-app-basic rest
```

Protótipo Yii2 REST

- Criar o schema e uma tabela

```
evandro@mortimer: ~  
mysql> create schema test;  
Query OK, 1 row affected (0,00 sec)  
  
mysql> use test  
Database changed  
mysql> create table item (  
    -> id integer auto_increment not null,  
    -> name varchar(255) not null,  
    -> primary key (id)  
    -> );  
Query OK, 0 rows affected (0,03 sec)  
  
mysql>
```

Protótipo Yii2 REST

- Criar o modelo para a tabela criada

```
evandro@mortimer: ~  
[mortimer 9] evandro@mortimer:~$ ./yii gii/model \  
> --tableName=item \  
> --modelClass=Item
```

Protótipo Yii2 REST

- Criar o CRUD para o modelo criado

```
evandro@mortimer: ~  
[mortimer 9] evandro@mortimer:~$ ./yii gii/crud \  
> --modelClass=app\models\Item \  
> --searchModelClass=app\models\ItemSearch \  
> --controllerClass=app\controllers\ItemController
```

Protótipo Yii2 REST

- Criar o módulo para implementar uma API em REST

```
evandro@mortimer: ~  
[mortimer 9] evandro@mortimer:~$ ./yii gii/module \  
> --moduleClass=app\modules\api\Module \  
> --moduleID=api
```

A. SOAP vs REST

- SOAP:
 - SOAP brings its own protocol and focuses on exposing pieces of application logic (not data) as services. SOAP exposes operations. SOAP is focused on accessing named operations, each implement some business logic through different interfaces.
 - Though SOAP is commonly referred to as “web services” this is a misnomer. SOAP has very little if anything to do with the Web. REST provides true “Web services” based on URIs and HTTP
- REST:
 - RESTs sweet spot is when you are exposing a public API over the internet to handle CRUD operations on data. REST is focused on accessing named resources through a single consistent interface.

A. SOAP vs REST

why REST

- Since REST uses standard HTTP it is much simpler in just about every way. Creating clients, developing APIs, the documentation is much easier to understand and there aren't very many things that REST doesn't do easier/better than SOAP.
- REST permits many different data formats where as SOAP only permits XML. While this may seem like it adds complexity to REST because you need to handle multiple formats, in my experience it has actually been quite beneficial. JSON usually is a better fit for data and parses much faster. REST allows better support for browser clients due to its support for JSON.
- REST has better performance and scalability. REST reads can be cached, SOAP based reads cannot be cached.
- It's a bad argument (by authority), but it's worth mentioning that Yahoo uses REST for all their services including Flickr and del.icio.us. Amazon and Ebay provide both though Amazon's internal usage is nearly all REST source. Google used to provide only SOAP for all their services, but in 2006 they deprecated in favor of REST source. It's interesting how there has been an internal battle between rest vs soap at amazon. For the most part REST dominates their architecture for web services.

A. SOAP vs REST

why SOAP

- WS-Security: while SOAP supports SSL (just like REST) it also supports WS-Security which adds some enterprise security features. Supports identity through intermediaries, not just point to point (SSL). It also provides a standard implementation of data integrity and data privacy. Calling it “Enterprise” isn’t to say it’s more secure, it simply supports some security tools that typical internet services have no need for, in fact they are really only needed in a few “enterprise” scenarios.
- WS-AtomicTransaction: need ACID Transactions over a service, you’re going to need SOAP. While REST supports transactions, it isn’t as comprehensive and isn’t ACID compliant. Fortunately ACID transactions almost never make sense over the internet. REST is limited by HTTP itself which can’t provide two-phase commit across distributed transactional resources, but SOAP can. Internet apps generally don’t need this level of transactional reliability, enterprise apps sometimes do.
- WS-ReliableMessaging: Rest doesn’t have a standard messaging system and expects clients to deal with communication failures by retrying. SOAP has successful/retry logic built in and provides end-to-end reliability even through SOAP intermediaries.

A. SOAP vs REST

- Em resumo:
 - In Summary, SOAP is clearly useful, and important. For instance, if I was writing an iPhone application to interface with my bank I would definitely need to use SOAP. All three features above are required for banking transactions. For example, if I was transferring money from one account to the other, I would need to be certain that it completed. Retrying it could be catastrophic if it succeed the first time, but the response failed.



B. HTTP e REST

- O protocolo HTTP está estruturado numa arquitetura Cliente-Servidor que funciona no modo Pedido-Resposta
- Um pedido é sempre iniciado pelo cliente (user agent) e a resposta é composta por informação de estado e pode ou não conter um conteúdo
- Um navegador web é um exemplo de um user agent. Outros tipos de clientes: web crawlers, aplicações de texto, desktop ou mobile

B. HTTP e REST

```
josh@blackbox:~$ telnet en.wikipedia.org 80
Trying 208.80.152.2...
Connected to rr.pmtpa.wikimedia.org.
Escape character is '^]'.
GET /wiki/Main_Page http/1.1
Host: en.wikipedia.org

HTTP/1.0 200 OK
Date: Thu, 03 Jul 2008 11:12:06 GMT
Server: Apache
X-Powered-By: PHP/5.2.5
Cache-Control: private, s-maxage=0, max-age=0, must-revalidate
Content-Language: en
Vary: Accept-Encoding, Cookie
X-Vary-Options: Accept-Encoding;list-contains=gzip, Cookie;string-contains=enwikiToken;string-contains=enwikiLoggedOut;string-contains=enwiki_session;
string-contains=centralauth_Token;string-contains=centralauth_Session;string-contains=centralauth_LoggedOut
Last-Modified: Thu, 03 Jul 2008 10:44:34 GMT
Content-Length: 54218
Content-Type: text/html; charset=utf-8
X-Cache: HIT from sq39.wikimedia.org
X-Cache-Lookup: HIT from sq39.wikimedia.org:3128
Age: 3
X-Cache: HIT from sq38.wikimedia.org
X-Cache-Lookup: HIT from sq38.wikimedia.org:80
Via: 1.0 sq39.wikimedia.org:3128 (squid/2.6.STABLE18), 1.0 sq38.wikimedia.org:80 (squid/2.6.STABLE18)
Connection: close

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en" dir="ltr">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <meta name="keywords" content="Main Page,1778,1844,1863,1938,1980 Summer Olympics,2008,2008 Guizhou riot,2008 Jerusal
...
... This content has been removed to save space
...
"Non-profit organization">nonprofit</a> <a href="http://en.wikipedia.org/wiki/Charitable_organization" title="Charitable organization">charity</a>.<b
r /></li>
    <li id="privacy"><a href="http://wikimediafoundation.org/wiki/Privacy_policy" title="wikimedia:Privacy policy">Privac
y policy</a></li>
    <li id="about"><a href="/wiki/Wikipedia:About" title="Wikipedia:About">About Wikipedia</a></li>
    <li id="disclaimer"><a href="/wiki/Wikipedia:General_disclaimer" title="Wikipedia:General disclaimer">Disclaimers</a>
</li>
  </ul>
</div>
</div>

  <script type="text/javascript">if (window.runOnloadHook) runOnloadHook();</script>
<!-- Served by srv93 in 0.050 secs. --></body></html>
Connection closed by foreign host.
josh@blackbox:~$
```

Request

Response headers

Response body

B. HTTP e REST

- Formato da mensagem do Pedido
 - Pedido: método + recurso + versão do protocolo
 - GET images/logo.png HTTP/1.1
 - Cabeçalhos
 - Accept-Language: en
 - Linha em branco
 - Conteúdo da mensagem (opcional)
- Métodos do Pedido
 - GET, HEAD, POST, PUT, DELETE, TRACE, OPTIONS, CONNECT e PATCH

B. HTTP e REST

- Tabela resumo dos métodos

HTTP method ↕	RFC ↕	Request has Body ↕	Response has Body ↕	Safe ↕	Idempotent ↕	Cacheable ↕
GET	RFC 7231 ↗	No	Yes	Yes	Yes	Yes
HEAD	RFC 7231 ↗	No	No	Yes	Yes	Yes
POST	RFC 7231 ↗	Yes	Yes	No	No	Yes
PUT	RFC 7231 ↗	Yes	Yes	No	Yes	No
DELETE	RFC 7231 ↗	No	Yes	No	Yes	No
CONNECT	RFC 7231 ↗	No	Yes	No	No	No
OPTIONS	RFC 7231 ↗	Optional	Yes	Yes	Yes	No
TRACE	RFC 7231 ↗	No	Yes	Yes	Yes	No
PATCH	RFC 5789 ↗	Yes	Yes	No	No	No

B. HTTP e REST

- Formato da mensagem da Resposta
 - Código e mensagem de estado do pedido
 - HTTP/1.1 200 OK
 - Cabeçalhos
 - Content-Type: text/html
 - Linha em branco
 - Conteúdo da mensagem (opcional)
- Códigos de estado
 - 1XX Informação
 - 2XX Sucesso
 - 3XX Redirecionar
 - 4XX Erro no pedido do cliente
 - 5XX Erro no servidor

B. HTTP e REST

- Exemplo de um pedido

```
GET / HTTP/1.1
Host: www.example.com
```

- Exemplo de uma resposta

```
HTTP/1.1 200 OK
Date: Mon, 23 May 2005 22:38:34 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 138
Last-Modified: Wed, 08 Jan 2003 23:11:55 GMT
Server: Apache/1.3.3.7 (Unix) (Red-Hat/Linux)
ETag: "3f80f-1b6-3e1cb03b"
Accept-Ranges: bytes
Connection: close

<html>
  <head>
    <title>An Example Page</title>
  </head>
  <body>
    <p>Hello World, this is a very simple HTML document.</p>
  </body>
</html>
```


B. HTTP e REST

- Links úteis
 - https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol
 - https://en.wikipedia.org/wiki/List_of_HTTP_header_fields
 - https://en.wikipedia.org/wiki/List_of_HTTP_status_codes
- Ferramentas recomendadas
 - Mozilla Firefox, Google Chrome / Chromium
 - RESTED
 - <https://github.com/RESTEDClient/RESTED>
 - telnet

B. HTTP e REST

- Caminho URI em serviços REST

URI	<code>://api.ipb.pt/alunos</code>	<code>://api.ipb.pt/alunos/a13610</code>
GET	Devolve os URIs para aceder aos alunos	Devolve um aluno
POST	Cria um novo aluno	Cria um aluno
PUT	Substitui ou cria um conjunto de alunos	Substitui ou cria um aluno
PATCH	Atualiza o conjunto de alunos	Atualiza um aluno podendo também criar um aluno
DELETE	Apaga todos os alunos	Apaga um aluno

B. HTTP e REST

- Converter URIs “tradicionais” em “Clean URLs”

Uncleaned URL	Clean URL
http://example.com/index.php?page=name	http://example.com/name
http://example.com/about.html	http://example.com/about
http://example.com/index.php?page=consulting/marketing	http://example.com/consulting/marketing
http://example.com/products?category=12&pid=25	http://example.com/products/12/25
http://example.com/cgi-bin/feed.cgi?feed=news&frm=rss	http://example.com/news.rss
http://example.com/services/index.jsp?category=legal&id=patents	http://example.com/services/legal/patents
http://example.com/kb/index.php?cat=8&id=41	http://example.com/kb/8/41
http://example.com/index.php?mod=profiles&id=193	http://example.com/profiles/193
http://en.wikipedia.org/w/index.php?title=Clean_URL	http://en.wikipedia.org/wiki/Clean_URL



B. HTTP e REST

- HATEOAS - Hypermedia As The Engine Of Application State
 - Faz parte da arquitetura REST
 - O cliente não precisa saber todos os caminhos disponíveis num serviço, a cada pedido é indicado todos os caminhos possíveis seguintes
 - O cliente apenas precisa saber o ponto de entrada (raiz/endereço do serviço)
 - Permite a evolução das funcionalidade implementadas no servidor sem ser preciso informar o cliente previamente

B. HTTP e REST

- HATEOAS - Hypermedia As The Engine Of Application State
 - Exemplo de um pedido

```
GET /accounts/12345/ HTTP/1.1
Host: bank.example.com
Accept: application/xml
...
```

- Exemplo da resposta com os possíveis caminhos

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: ...

<?xml version="1.0"?>
<account>
  <account_number>12345</account_number>
  <balance currency="usd">100.00</balance>
  <link rel="deposit" href="/accounts/12345/deposit" />
  <link rel="withdraw" href="/accounts/12345/withdraw" />
  <link rel="transfer" href="/accounts/12345/transfer" />
  <link rel="close" href="/accounts/12345/close" />
</account>
```

C. Aplicação Yii2 REST

- criar módulo: usar o gerador Gii e não esquecer de adicionar o módulo em config/web.php

```
1 <?php
2
3 $params = require(__DIR__ . '/params.php');
4
5 $config = [
6     'id' => 'basic',
7     'basePath' => dirname(__DIR__),
8     'bootstrap' => ['log'],
9     'components' => [
10         ...
11     ],
12     'params' => $params,
13     'modules' => [
14         'api-v1' => [
15             'class' => 'app\modules\api\Module',
16         ],
17     ],
18 ];
19
20 return $config;
21
```

C. Aplicação Yii2 REST

- criar controladores com herança de yii\rest\ActiveController

```
1 <?php
2 namespace app\modules\api\controllers;
3
4 use app\models\Category;
5 use Yii;
6 use webvimark\modules\UserManagement\components\GhostAccessControl;
7 use yii\data\ActiveDataProvider;
8 use yii\filters\RateLimiter;
9 use yii\filters\auth\HttpBasicAuth;
10 use yii\helpers\ArrayHelper;
11 use yii\rest\ActiveController;
12 use yii\web\ServerErrorHttpException;
13
14 class CategoryController extends ActiveController {
15     public $modelClass = 'app\models\Category';
16 }
17
18
```

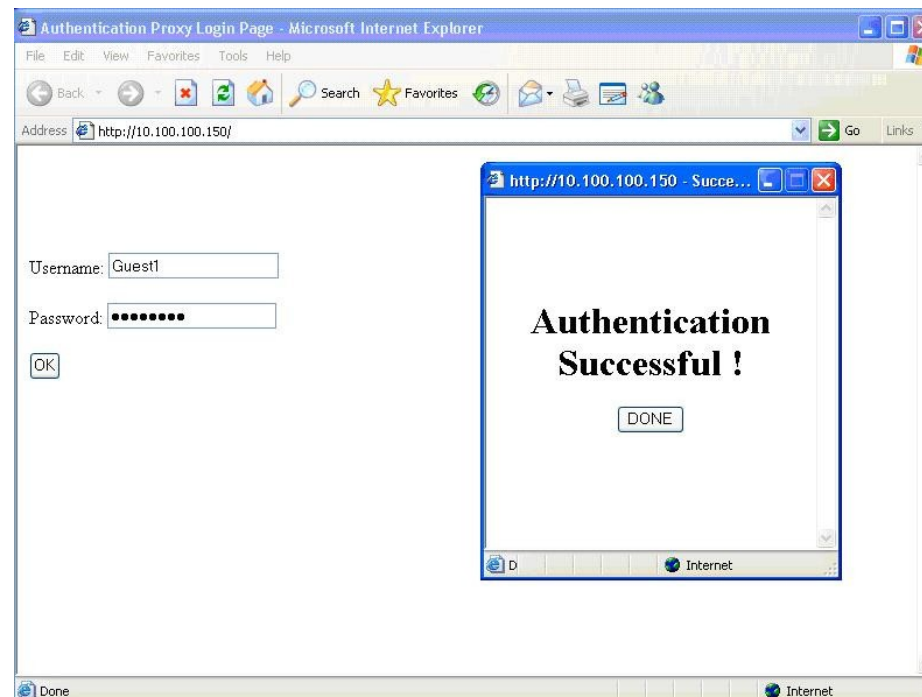
C. Aplicação Yii2 REST

- configurar caminhos RESTfull em config/web.php

```
1 <?php
2
3 $params = require(__DIR__ . '/params.php');
4
5 $config = [
6     'id' => 'basic',
7     'basePath' => dirname(__DIR__),
8     'bootstrap' => ['log'],
9     'components' => [
10         ...
11         'urlManager' => [
12             'enablePrettyUrl' => true,
13             'rules' => [
14                 ['class' => 'yii\rest\UrlRule', 'controller' => 'api-v1/auth'],
15                 ['class' => 'yii\rest\UrlRule', 'controller' => 'api-v1/category'],
16             ],
17         ],
18     ],
19     'params' => $params,
20     'modules' => [
21         'api-v1' => [
22             'class' => 'app\modules\api\Module',
23         ],
24     ],
25 ];
26
27 return $config;
28
```


D. Autenticação e Autorização

- Autenticação é o ato de confirmar a verdade acerca de um atributo
- O atributo pode ser a identidade de um utilizador
- Quando um utilizador se autentica num serviço primeiro indica o atributo (username, login, id, etc...) e depois confirma o atributo como verdadeiro com uma palavra-chave (password, chave ou outro)



D. Autenticação e Autorização

- Autorização é o ato de especificar os direitos de acesso ou privilégios sobre um determinado recurso
- Durante a operação é retornado o direito ou privilégio em função da consulta de regras de acesso utilizador ↔ recurso
- Assim, a autorização apenas se pode aplicar depois de autenticado o acesso

Subject	Object	Action
Sales department	Customer record	Insert
Order transactions	Customer record	Read

D. Autenticação e Autorização - *impl*

- estender a classe `yii\filters\auth\HttpBasicAuth` para validar `username:password`

```
1 <?php
2
3 namespace app\modules\api\components;
4
5 use webvmark\modules\UserManagement\models\User;
6
7 class HttpBasicAuth extends \yii\filters\auth\HttpBasicAuth
8 {
9     public function __construct($config = []) {
10         parent::__construct($config);
11         $this->auth = function ($username, $password) {
12             $user = User::findByUsername($username);
13             if (isset($user)) {
14                 if ($user->validatePassword($password)) {
15                     return $user;
16                 } else {
17                     return null;
18                 }
19             }
20             return null;
21         };
22     }
23 }
```

D. Autenticação e Autorização - *impl*

- adicionar behaviors no controlador de autenticação que obtem auth key adicionando 'authenticator' => 'app\modules\api\components\

```
1 <?php
2
3 namespace app\modules\api\controllers;
4
5 use app\modules\api\v1\components\HttpBasicAuth;
6 use webvimark\modules\UserManagement\components\GhostAccessControl;
7 use webvimark\modules\UserManagement\models\User;
8 use yii\helpers\ArrayHelper;
9
10 /**
11  * This controller can only be requested by active users with the right permissions using HTTP Basic authentication
12  * using username:password only!
13  * @author evandro
14  *
15  */
16 class AuthController extends \yii\rest\Controller {
17     public $defaultAction = 'auth-key';
18
19     /**
20      * {@inheritdoc}
21      * @see \yii\rest\Controller::behaviors()
22      */
23     public function behaviors() {
24         return ArrayHelper::merge(parent::behaviors(), [
25             'ghost-access' => [
26                 'class' => GhostAccessControl::class,
27             ],
28             'authenticator' => [
29                 'class' => HttpBasicAuth::class,
30             ]
31         ]);
32     }
33 }
```

D. Autenticação e Autorização - *impl*

- adicionar behaviors nos controladores ActiveControllers adicionando 'authenticator' => 'yii\filters\auth\HttpBasicAuth'

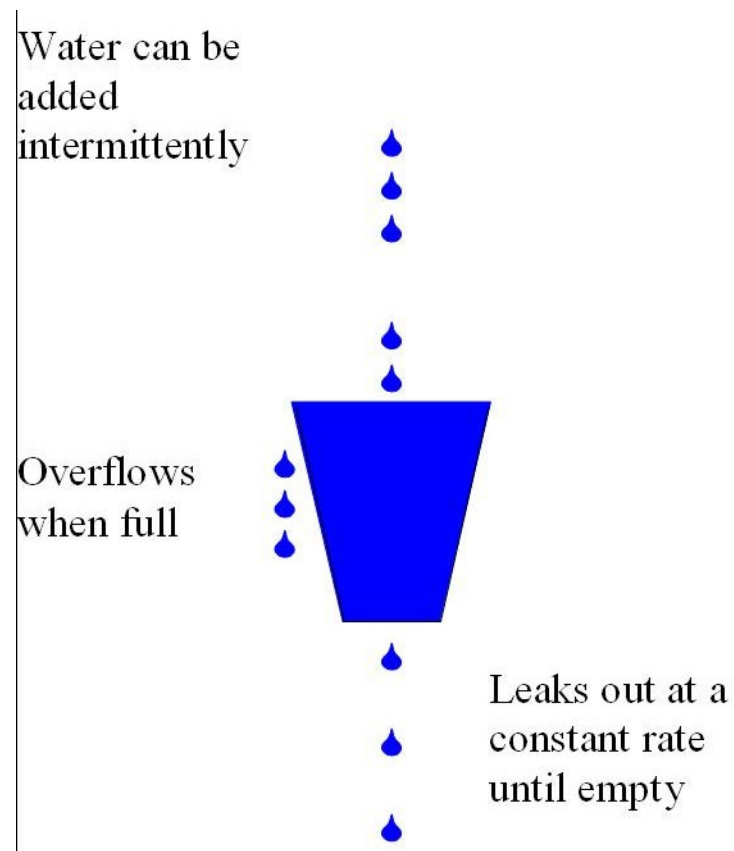
```
1 <?php
2 namespace app\modules\api\controllers;
3
4 use app\models\Category;
5 use Yii;
6 use webvimark\modules\UserManagement\components\GhostAccessControl;
7 use yii\data\ActiveDataProvider;
8 use yii\filters\RateLimiter;
9 use yii\filters\auth\HttpBasicAuth;
10 use yii\helpers\ArrayHelper;
11 use yii\rest\ActiveController;
12 use yii\web\ServerErrorHttpException;
13
14 class CategoryController extends ActiveController {
15     public $modelClass = 'app\models\Category';
16
17     /**
18      *
19      * {@inheritdoc}
20      * @see \yii\rest\Controller::behaviors()
21      */
22     public function behaviors() {
23         return ArrayHelper::merge(parent::behaviors(), [
24             'ghost-access' => [
25                 'class' => GhostAccessControl::class,
26             ],
27             'authenticator' => [
28                 'class' => HttpBasicAuth::class,
29             ],
30         ]);
31     }
32 }
```

E. Rate Limiting

- Rate Limiting é a limitação do numero de pedidos que um determinado cliente pode efetuar numa determinada janela de tempo. Exemplo:
 - Cliente com a sessão *xpto* pode efetuar 5 pedidos por cada 10 segundos
 - Nos últimos 10 segundos já efetuou 5 pedidos, pelo que o pedido atual será rejeitado
- A implementação do *Rate Limiting* não deve ser ignorada, pois é uma forma de mitigar os ataques tipo *Denial of Service*
- O algoritmo de limitação de pedidos deve ser implementado na aplicação do lado do servidor e nunca no cliente nem no servidor web
- No caso de uma sessão atingir os limite de pedidos, a resposta HTTP deverá ser *429: Too Many Requests*

E. Rate Limiting

- Na framework Yii2 há um filtro que implementa a limitação de pedidos usando o algoritmo *Leaky Bucket Algorithm* (algoritmo do balde furado)



E. Rate Limiting - *impl*

- extender o componente 'user' apontando a propriedade \$identityClass para uma classe que faz herança de webvimark\modules\UserManagement\models\User

```
1 <?php
2 namespace app\components;
3
4
5 class UserConfig extends \webvimark\modules\UserManagement\components\UserConfig
6 {
7     const ROLE_USER = 'role_User';
8
9     public $identityClass = 'app\models\User';
10 }
11
```


E. Rate Limiting - *impl*

- extender webvimark\modules\UserManagement\models\User e implementar a interface yii\filters\RateLimitInterface

```
<?php
namespace app\models;
use yii\filters\RateLimitInterface;

class User extends \webvimark\modules\UserManagement\models\User implements RateLimitInterface
{
    private $rateLimit = 5;
    private $allowance = 1;
    private $allowance_updated_at;

    public function getRateLimit($request, $action)...

    public function loadAllowance($request, $action)...

    public function saveAllowance($request, $action, $allowance, $timestamp)...
```

E. Rate Limiting - *impl*

- adicionar o behavior 'rateLimiter' => 'yii\filters\RateLimiter' aos controladores ActionController

```
1 <?php
2 namespace app\modules\api\controllers;
3
4 use app\models\Category;
5 use Yii;
6 use webvimark\modules\UserManagement\components\GhostAccessControl;
7 use yii\data\ActiveDataProvider;
8 use yii\filters\RateLimiter;
9 use yii\filters\auth\HttpBasicAuth;
10 use yii\helpers\ArrayHelper;
11 use yii\rest\ActiveController;
12 use yii\web\ServerErrorHttpException;
13
14 class CategoryController extends ActiveController {
15     public $modelClass = 'app\models\Category';
16
17     /**
18      *
19      * {@inheritdoc}
20      * @see \yii\rest\Controller::behaviors()
21      */
22     public function behaviors() {
23         return ArrayHelper::merge(parent::behaviors(), [
24             'ghost-access' => [
25                 'class' => GhostAccessControl::class,
26             ],
27             'authenticator' => [
28                 'class' => HttpBasicAuth::class,
29             ],
30             'rateLimiter' => [
31                 'class' => RateLimiter::class,
32             ],
33         ]);
34     }
35 }
```