REST API Design

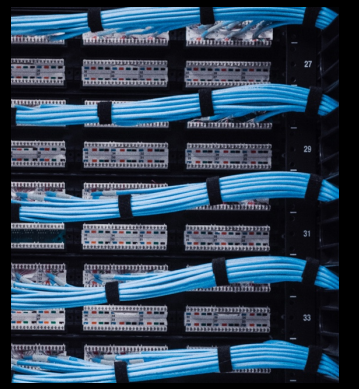
REST at ease!

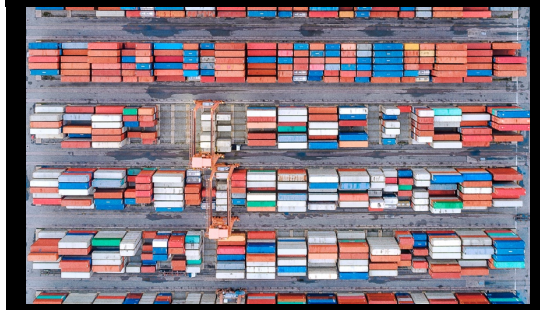
Inspired on presentation with a similar name given by Michael Smart

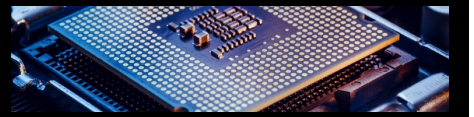
InterSystems®
Health | Business | Government

Amir Samary

Manager, Solution Architecture







Part One

Design

Two parts to every HTTP request

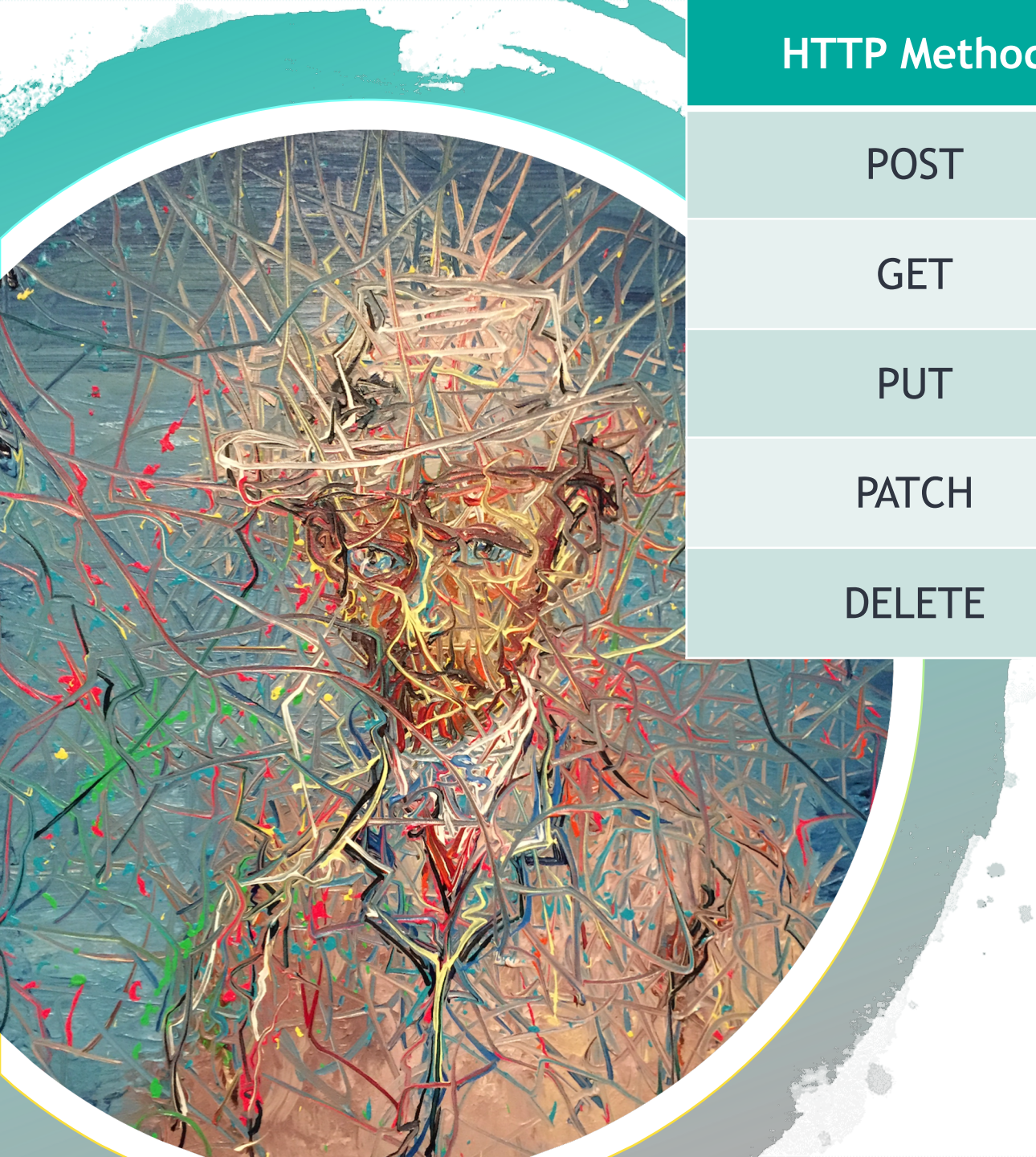
HTTP Method (Verb)

Resource

GET http://localhost:57772/csp/myapp/reservations

Server and Port

 © InterSystems Corporation. All rights reserved. |



HTTP Method	Action	SQL
POST	Create	INSERT
GET	Read	SELECT
PUT	Replace	DELETE + INSERT ?
PATCH	Update	UPDATE
DELETE	Delete	DELETE

Choosing your HTTP method

HTTP Method	Action	SQL
POST	Create	INSERT
GET	Read	SELECT
PUT	Replace	~~DELETE + INSERT ?~~ Maybe “FULL” UPDATE?
PATCH	Update	UPDATE
DELETE	Delete	DELETE

Choosing your HTTP method

How to choose your resource names

GET http://localhost:57772/csp/myapp/rest/**getReservations**

vs.

GET http://localhost:57772/csp/myapp/rest/**reservations**

 © InterSystems Corporation. All rights reserved. |

POST

POST http://localhost:57772/csp/myapp/rest/reservations

```
{
    "customer": "Amir",
    "startLocation": "Brazil",
    "destination": "Boston",
    "startDate": "2018-02-01",
    "endDate": "2018-02-10"
}
```

HTTP Status: 201 Created
Location: /csp/myapp/rest/reservations/**7**

 © InterSystems Corporation. All rights reserved. |

Expectations about “Idempotence” and “Safeness”

HTTP Method	Idempotent?	Safe?
POST	No	No
GET	Yes	Yes
PUT	Yes	No
PATCH	No	No
DELETE	Yes	No

An **idempotent** request can be repeated multiple times with the same end result on the server.

A **safe** request indicates that the data will not be changed on the server.

 © InterSystems Corporation. All rights reserved. |

Statelessness

GET http://localhost:57772/csp/myapp/rest/**reservation/12345**

 © InterSystems Corporation. All rights reserved. |

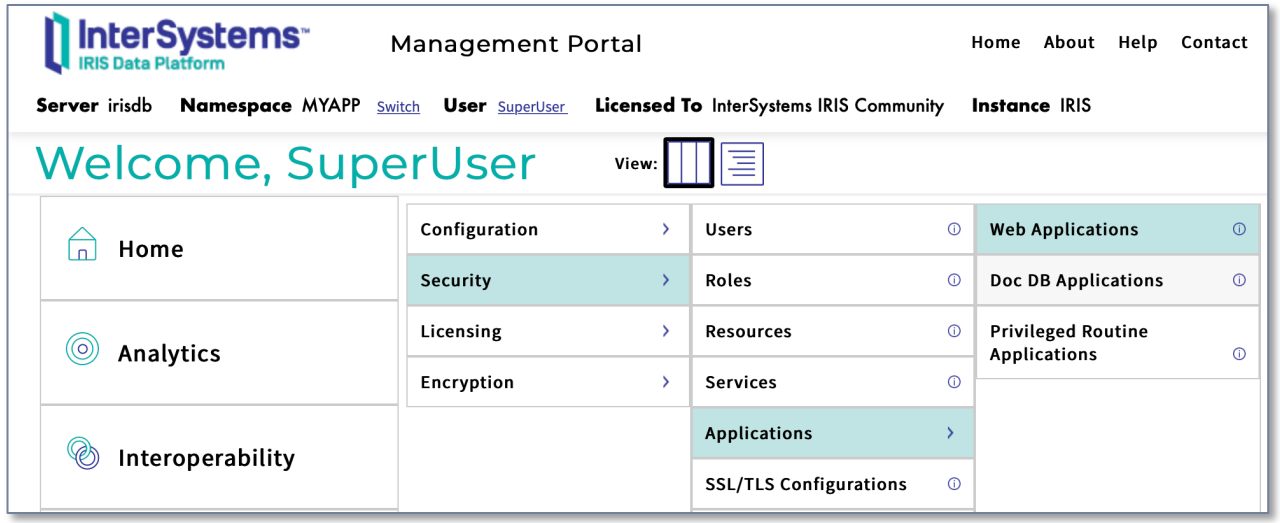
Demo!

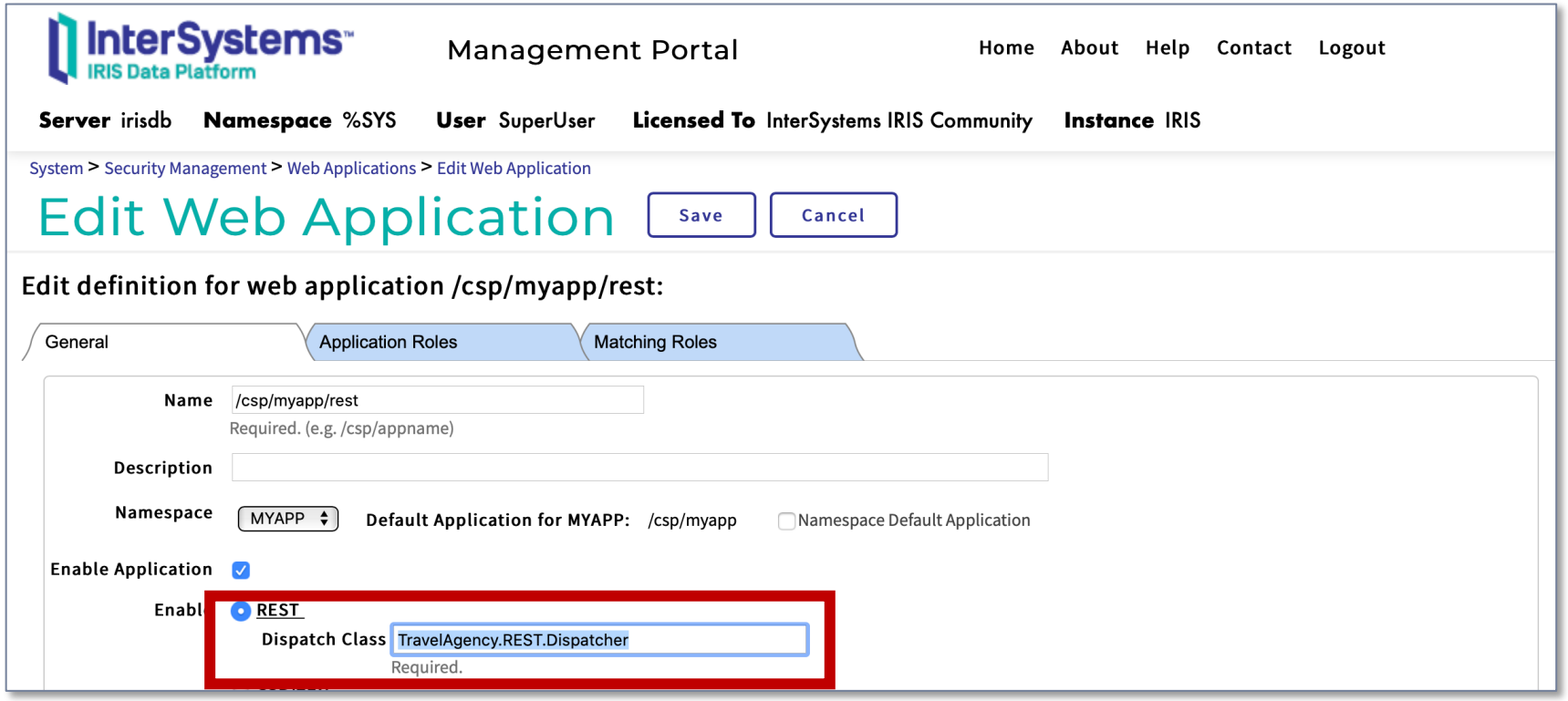
 © InterSystems Corporation. All rights reserved. |

Part Two

Implementation

Configure the REST Dispatcher





 © InterSystems Corporation. All rights reserved. |

Demo!

 © InterSystems Corporation. All rights reserved. |

D.R.Y.

(Don’t Repeat Yourself)

 © InterSystems Corporation. All rights reserved. |

Identifying repeated code

GET	POST	PATCH
Open reservation object	Convert request body into dynamic object	Convert request body into dynamic object
If it does not exist, handle errors	If not valid JSON, handle errors	Open reservation object
Create dynamic object	Create new reservation object	If it does not exist, handle errors
Return JSON string	Set properties and save	Set properties and save
		If not valid JSON, handle errors

 © InterSystems Corporation. All rights reserved. |

Identifying repeated code

GET	POST	PATCH
Open reservation object	Convert request body into dynamic object	Convert request body into dynamic object
If it does not exist, handle errors	If not valid JSON, handle errors	Open reservation object
Create dynamic object	Create new reservation object	If it does not exist, handle errors
Return JSON string	Set properties and save	Set properties and save
		If not valid JSON, handle errors

 © InterSystems Corporation. All rights reserved. |

Identifying repeated code

GET	POST	PATCH
Open reservation object	Convert request body into dynamic object	Convert request body into dynamic object
If it does not exist, handle errors	If not valid JSON, handle errors	If not valid JSON, handle errors
Create dynamic object	Create new reservation object	Open reservation object
Return JSON string	Set properties and save	If it does not exist, handle errors
		Set properties and save

 © InterSystems Corporation. All rights reserved. |

Identifying repeated code

Don’t repeat yourself.

GET	POST	PATCH
Open reservation object	Convert request body into dynamic object	Convert request body into dynamic object
If it does not exist, handle errors	If not valid JSON, handle errors	If not valid JSON, handle errors
Create dynamic object	Create new reservation object	Open reservation object
Return JSON string	Set properties and save	If it does not exist, handle errors
		Set properties and save

 © InterSystems Corporation. All rights reserved. |

Exceptions.

 © InterSystems Corporation. All rights reserved. |

Using helper methods and exceptions

- REST dispatch class calls helper methods
  - Improves readability
  - Modularizes code
- If errors occur in helper methods, throw an exception
  - REST dispatcher responsible for catching exceptions
- When an exception is caught, set the HTTP response status accordingly
  - Use custom exception classes for known problem cases

 © InterSystems Corporation. All rights reserved. |

Demo!

 © InterSystems Corporation. All rights reserved. |

https://github.com/intersystems/rest-and-relaxation

 © InterSystems Corporation. All rights reserved. |

Part Three

Advanced topics.

 © InterSystems Corporation. All rights reserved. |

CORS

Cross-Origin Resource Sharing

 © InterSystems Corporation. All rights reserved. |

CORS — Request from same origin

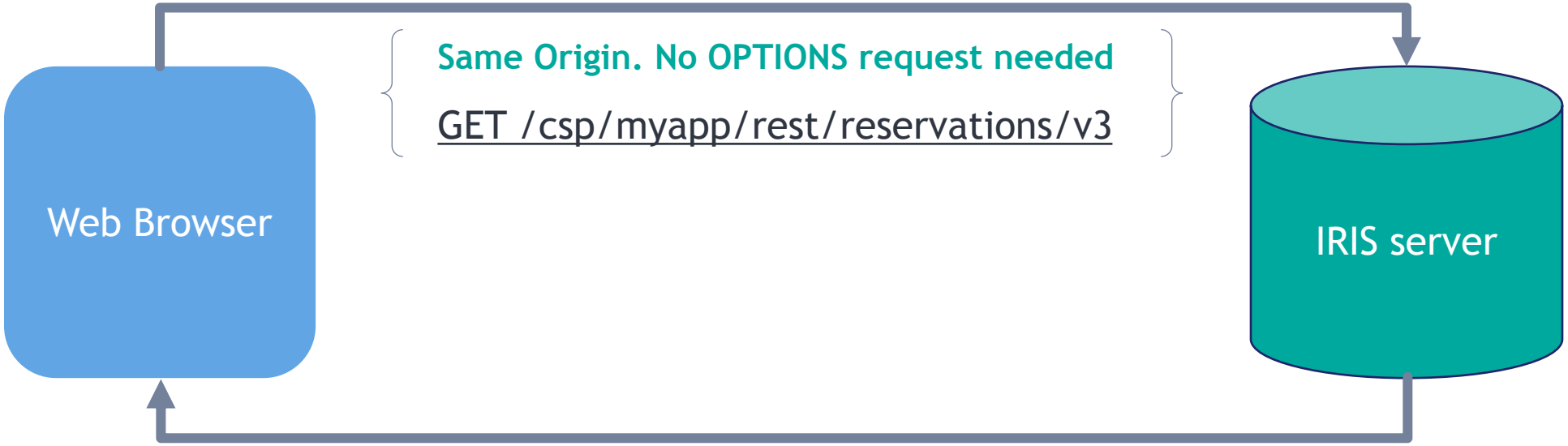
1. Web browser requests web application

2. Web server returns javascript code needed for web application

 © InterSystems Corporation. All rights reserved. |

CORS — Request from same origin

3. Javascript code makes request to REST service



4. REST response returned to browser.

 © InterSystems Corporation. All rights reserved. |

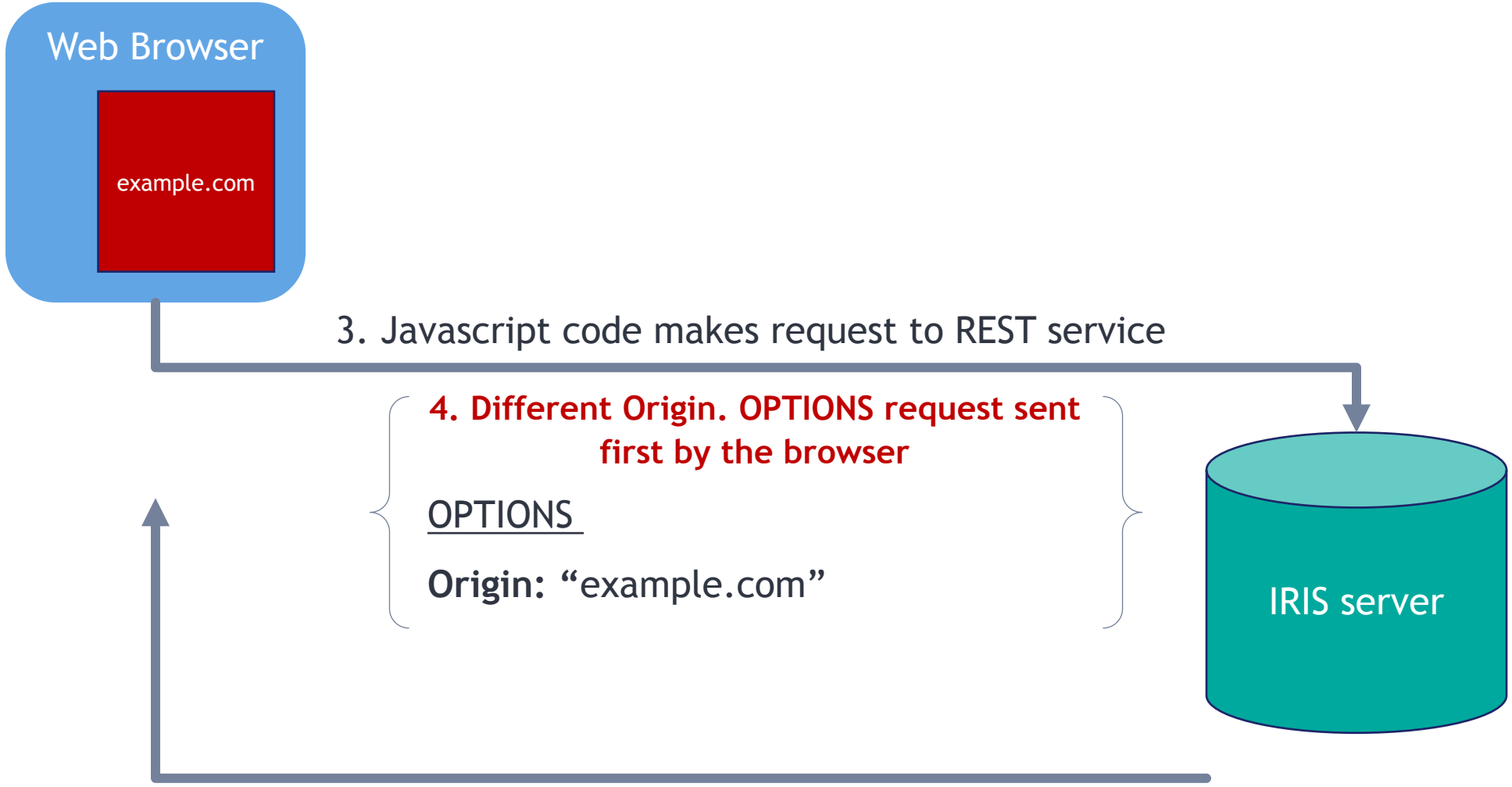
CORS — Request from different origin

1. Web browser requests web application on unrelated web site

2. Web server returns javascript code that calls REST service

 © InterSystems Corporation. All rights reserved. |

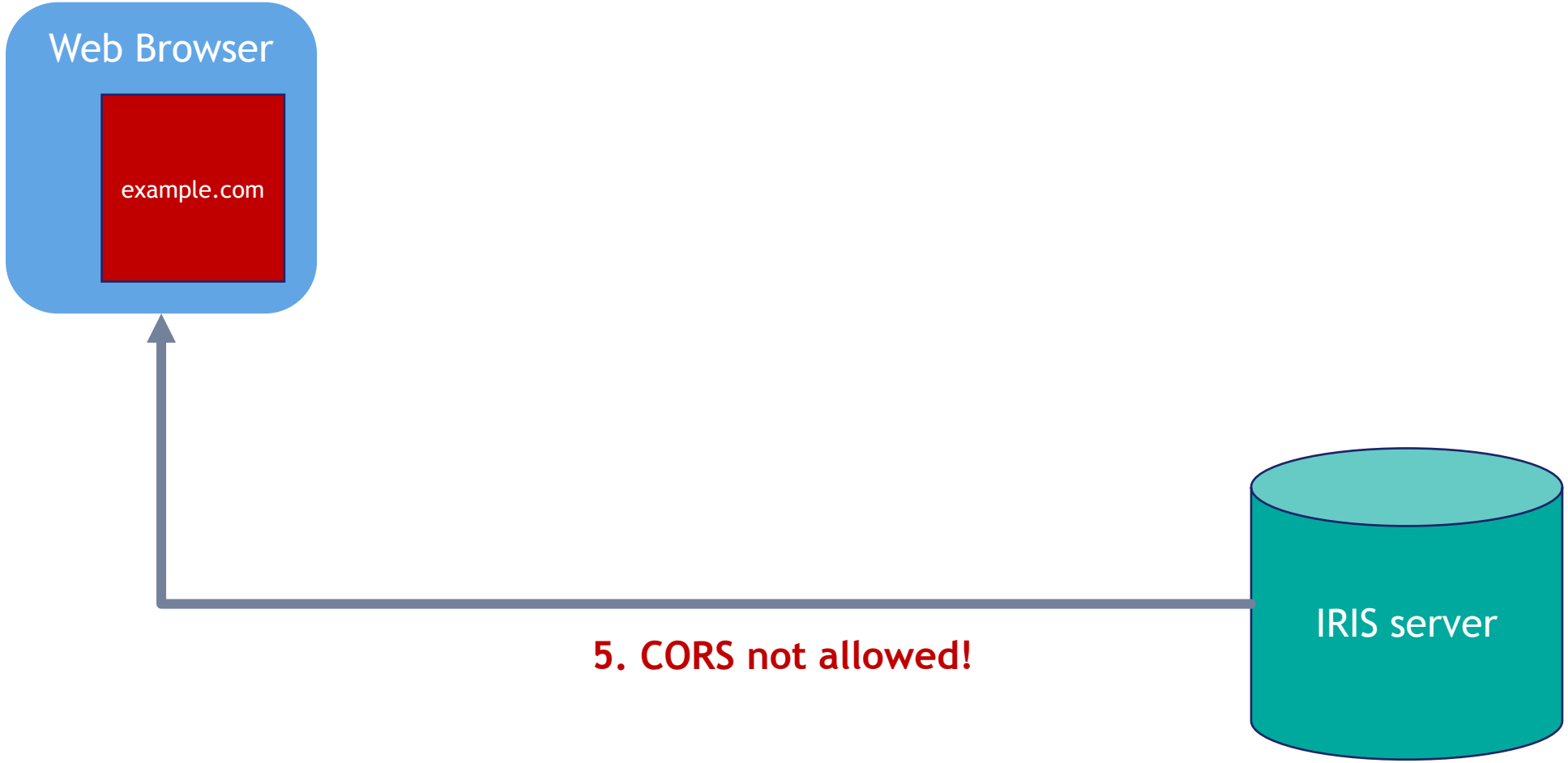
CORS — Request from different origin



5. CORS not allowed!

 © InterSystems Corporation. All rights reserved. |

CORS — Request from different origin



 © InterSystems Corporation. All rights reserved. |

CORS — Request from different origin

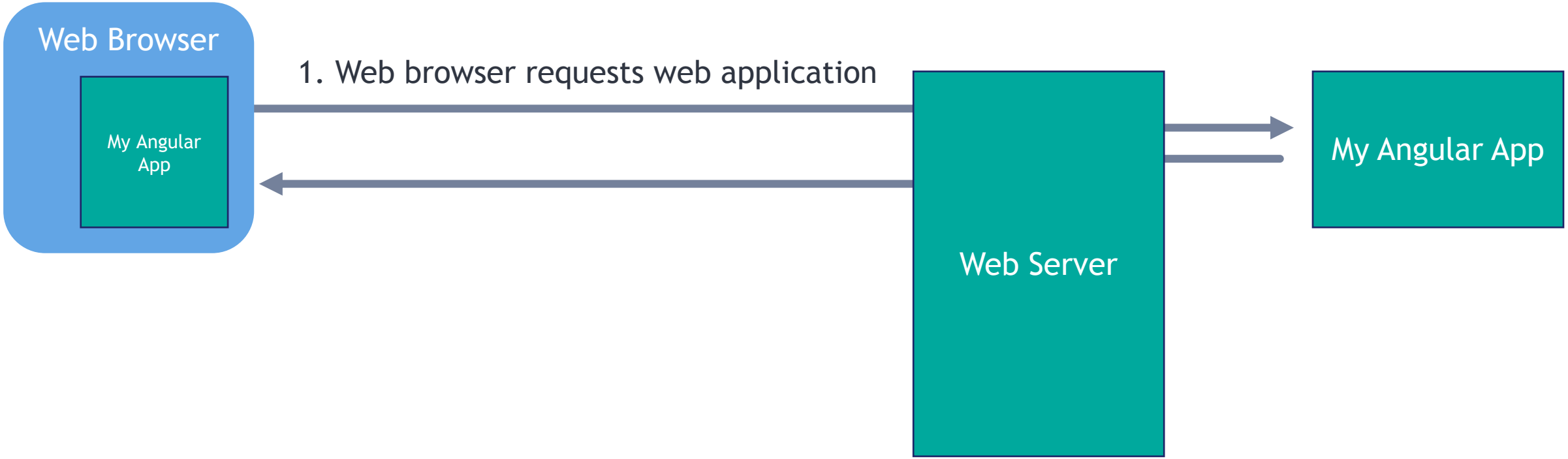
 © InterSystems Corporation. All rights reserved. |

CORS whitelist

- Read the documentation about CORS
- Add to your Dispatcher Class:
  - Parameter HandleCorsRequest = 1;
  - ClassMethod OnHandleCorsRequest(url)
  - Make sure CSPSystem user has access to your database!
- You should only enable this if you are providing REST API services for third party Web applications to use.

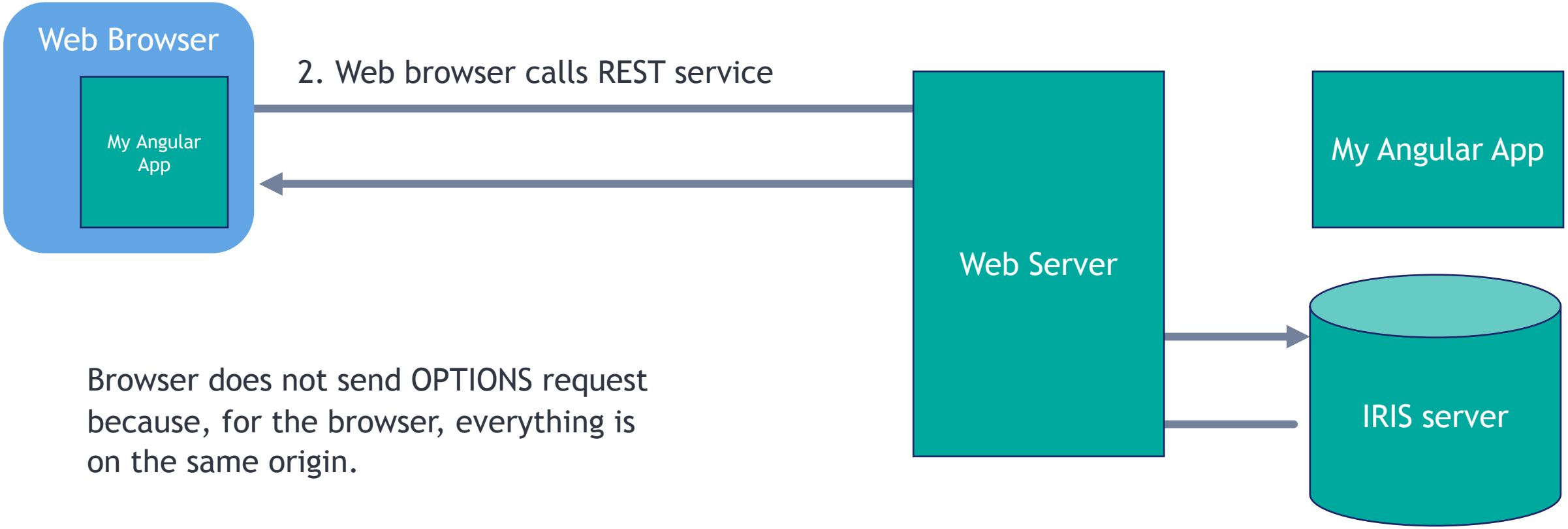
 © InterSystems Corporation. All rights reserved. |

CORS — Request from your own Application



 © InterSystems Corporation. All rights reserved. |

CORS — Request from your own Application



Browser does not send OPTIONS request because, for the browser, everything is on the same origin.

 © InterSystems Corporation. All rights reserved. |

Always have a Web Server in front of your IRIS Instance

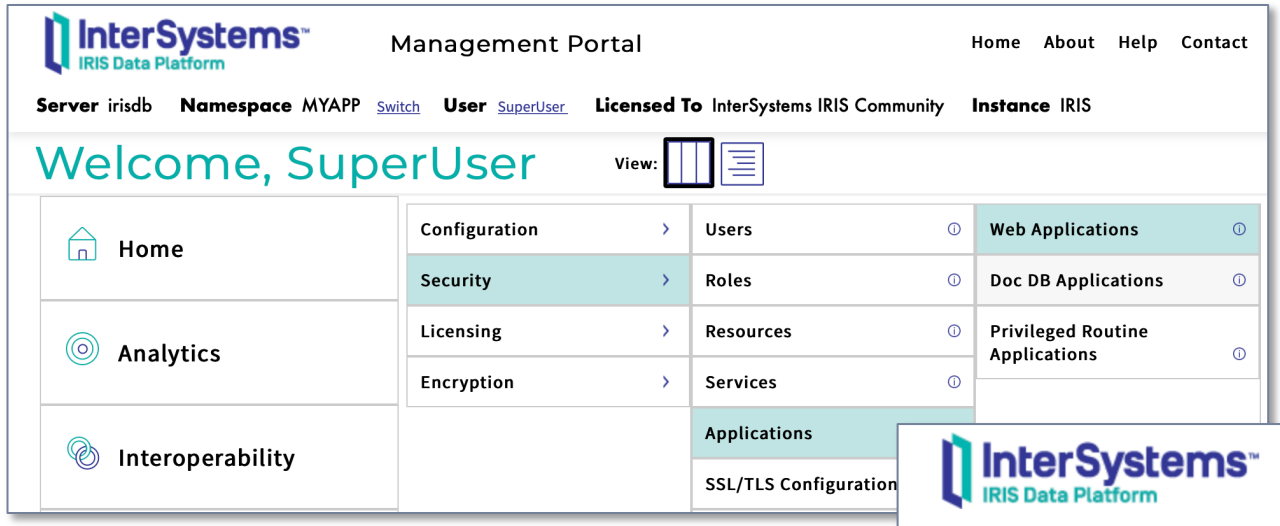
- Fixes the CORS issue for your application
- Protects your backend
- Provides much better performance than the embedded Web Server that comes with IRIS

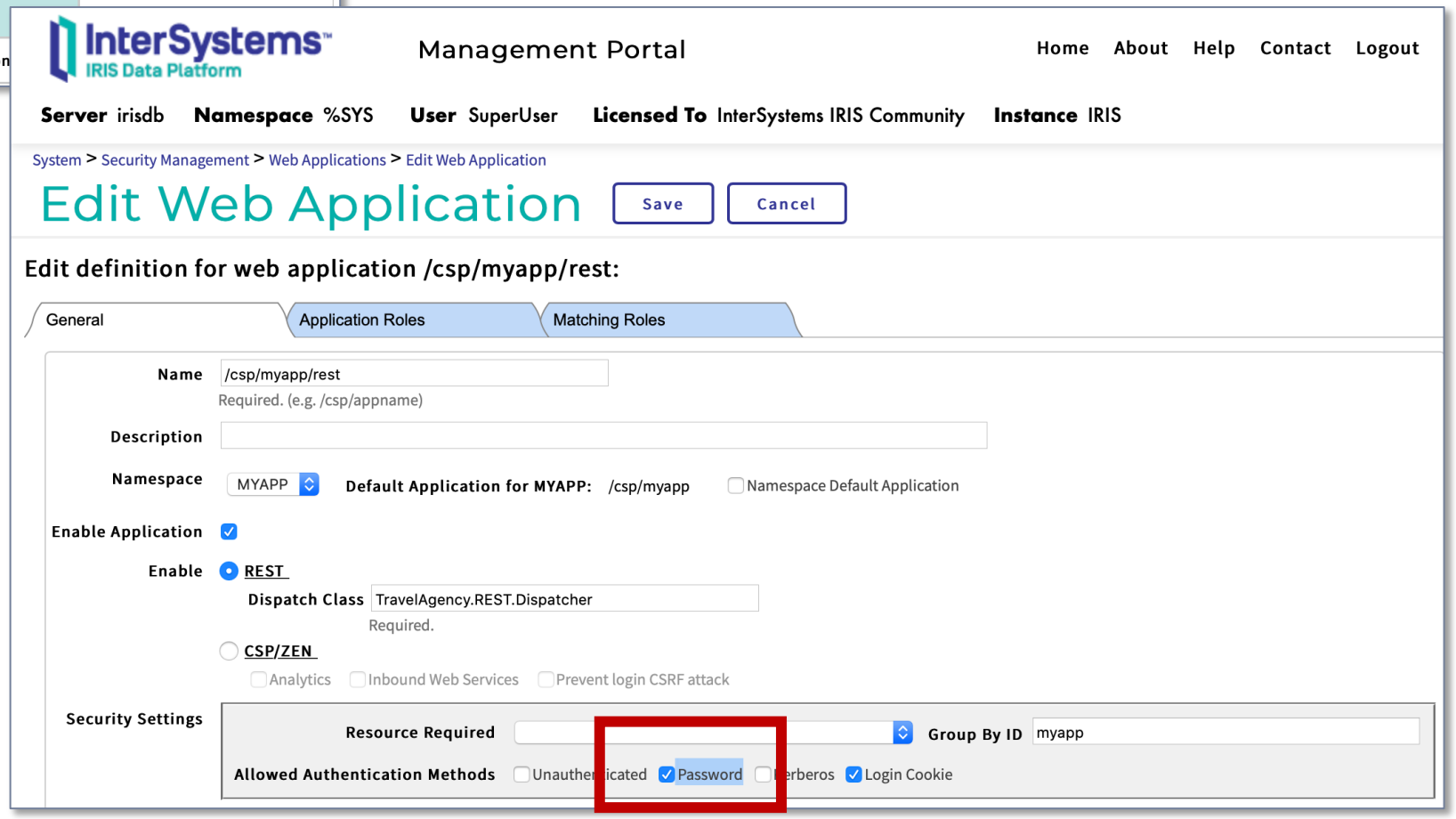
 © InterSystems Corporation. All rights reserved. |

Authentication.

 © InterSystems Corporation. All rights reserved. |

Configure Security Settings



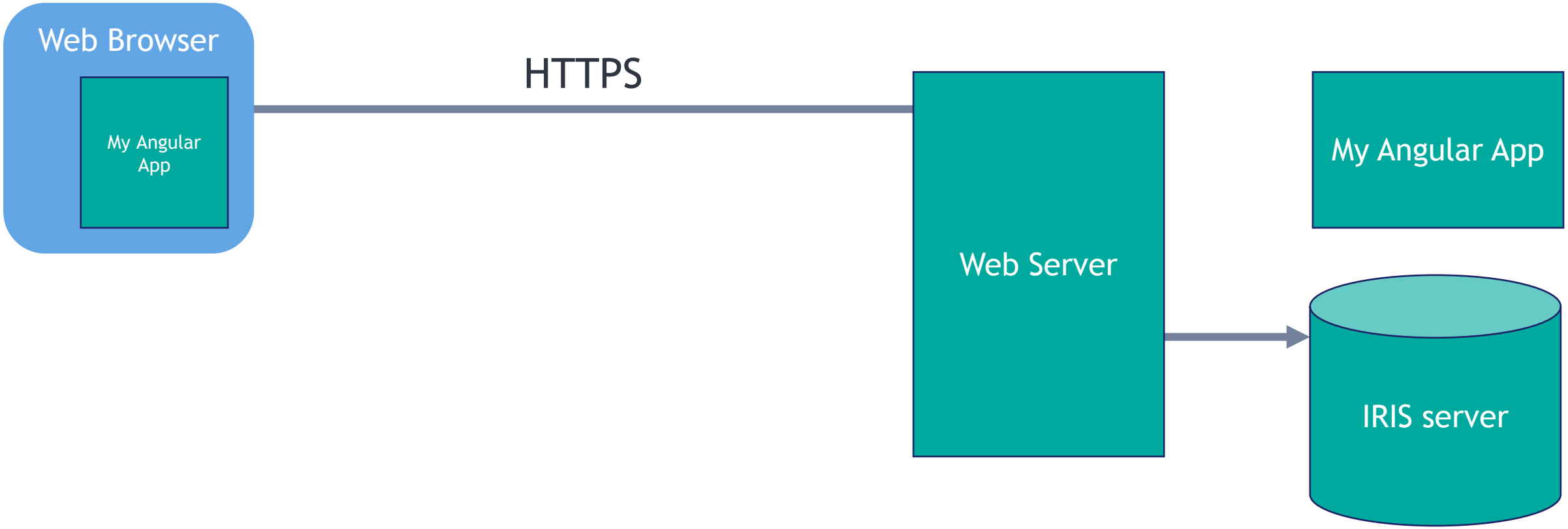


 © InterSystems Corporation. All rights reserved. |

Demo!

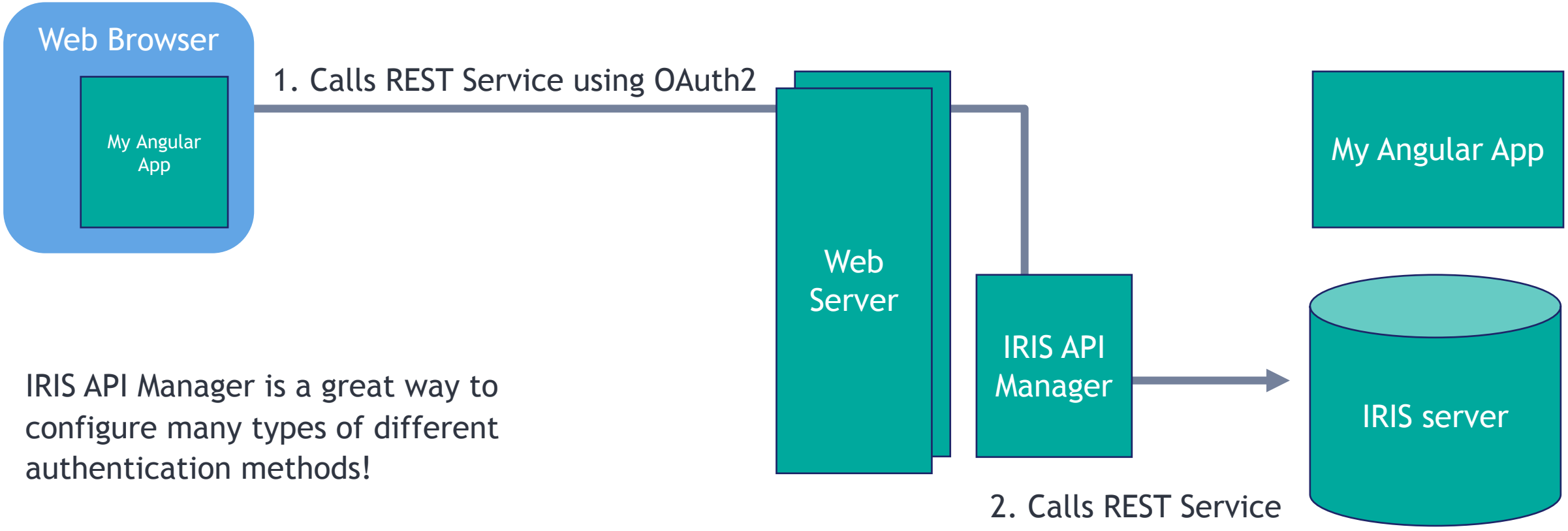
 © InterSystems Corporation. All rights reserved. |

Use TLS (HTTPS) to protect the personal information



 © InterSystems Corporation. All rights reserved. |

IRIS API Manager to enhance Security of your APIs



IRIS API Manager is a great way to configure many types of different authentication methods!

 © InterSystems Corporation. All rights reserved. |

Caching.

 © InterSystems Corporation. All rights reserved. |

Demo!

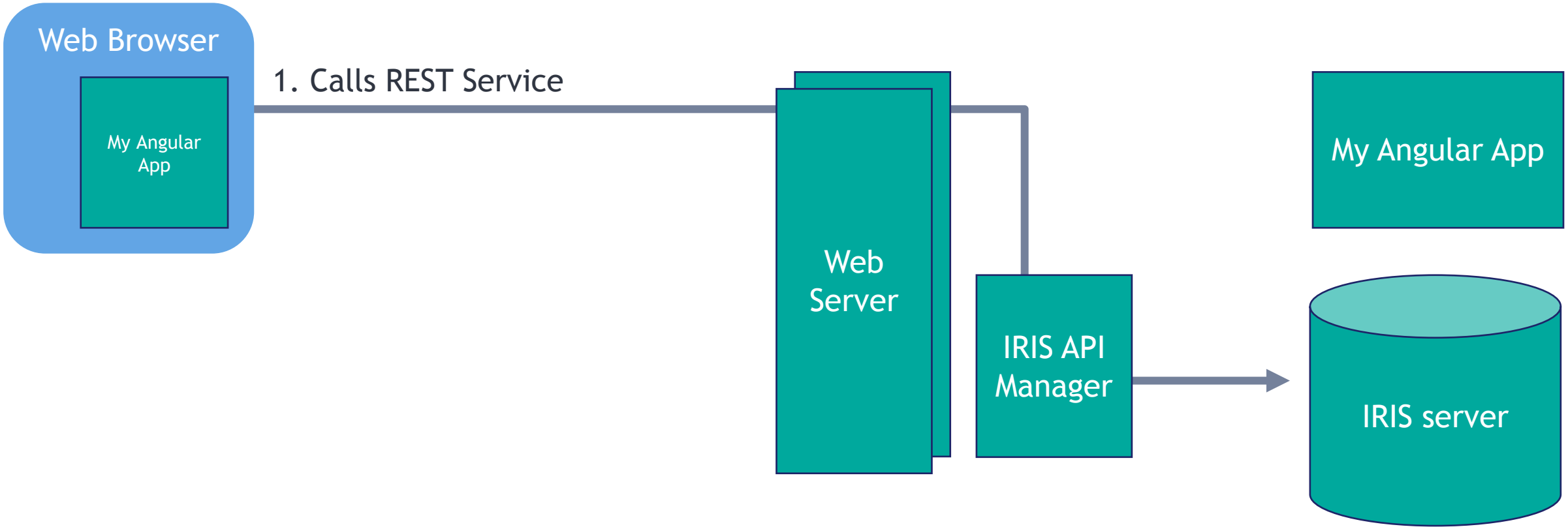
 © InterSystems Corporation. All rights reserved. |

Caching

- By default, IRIS doesn’t send any caching headers
- Caching is not easy to test
  - If you call your REST API using your browsers directly, the browser will ignore the caching headers and always make the call
  - Postman does the same thing. Both assume you WANT to ignore the cache to test your API
  - You must have a web page that, after loaded, calls your REST service.

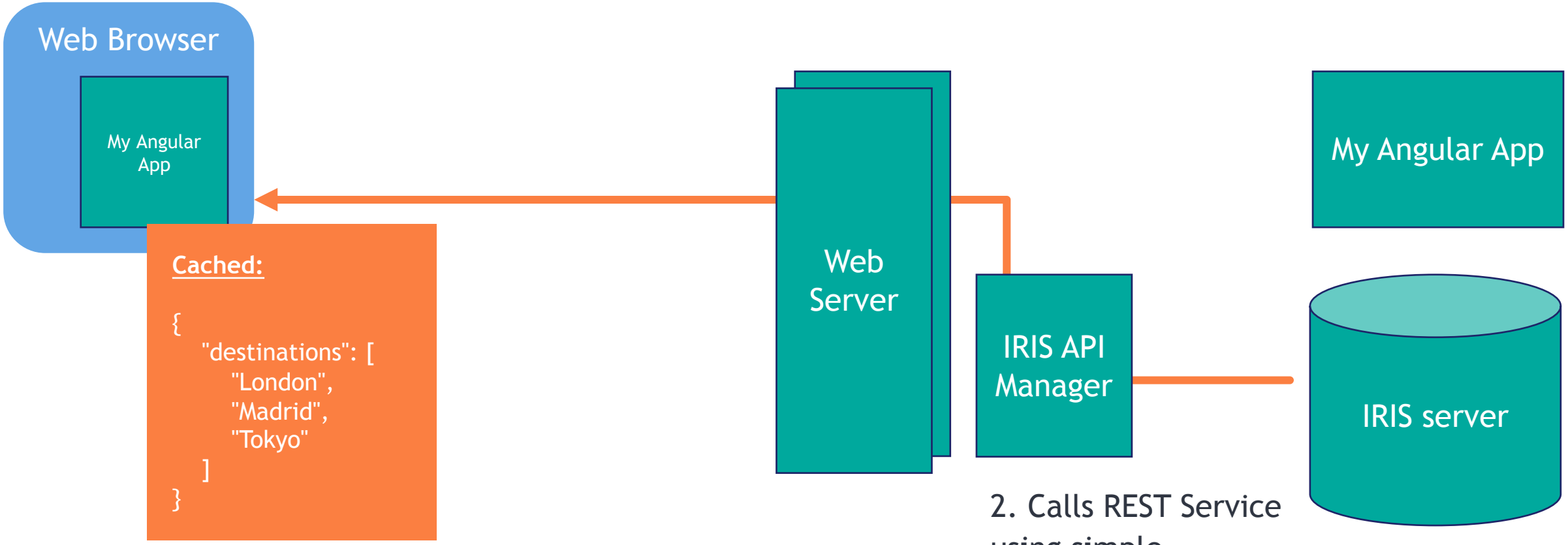
 © InterSystems Corporation. All rights reserved. |

Caching on the Browser



 © InterSystems Corporation. All rights reserved. |

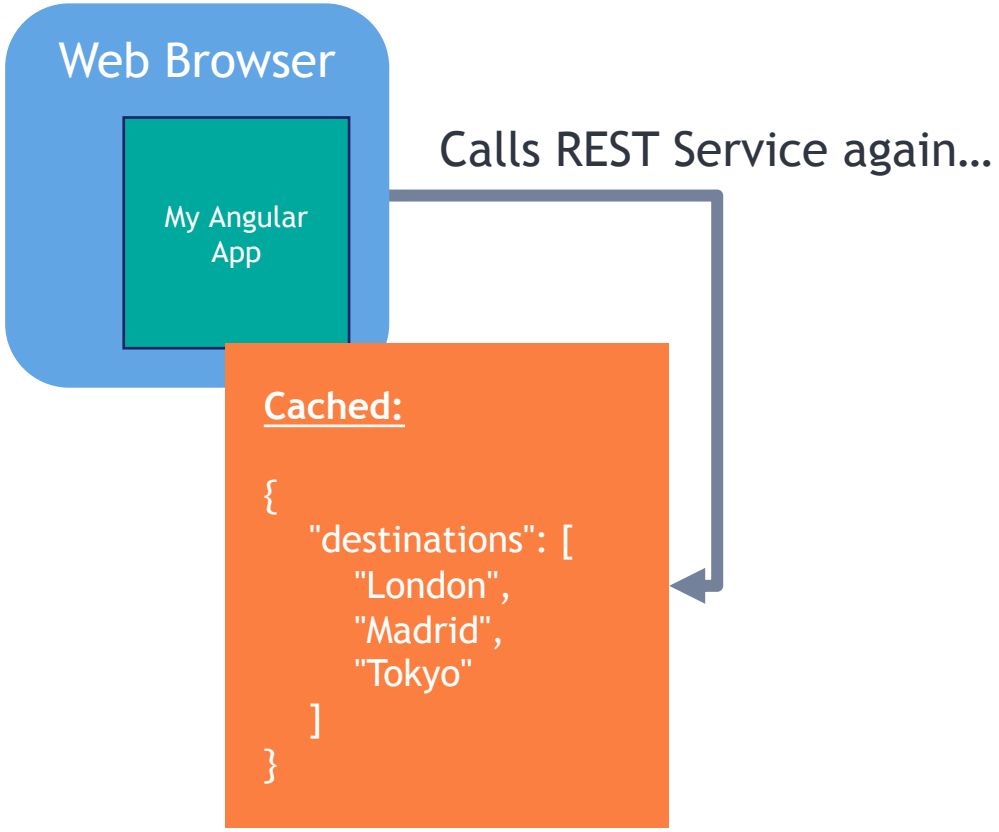
Caching on the Browser

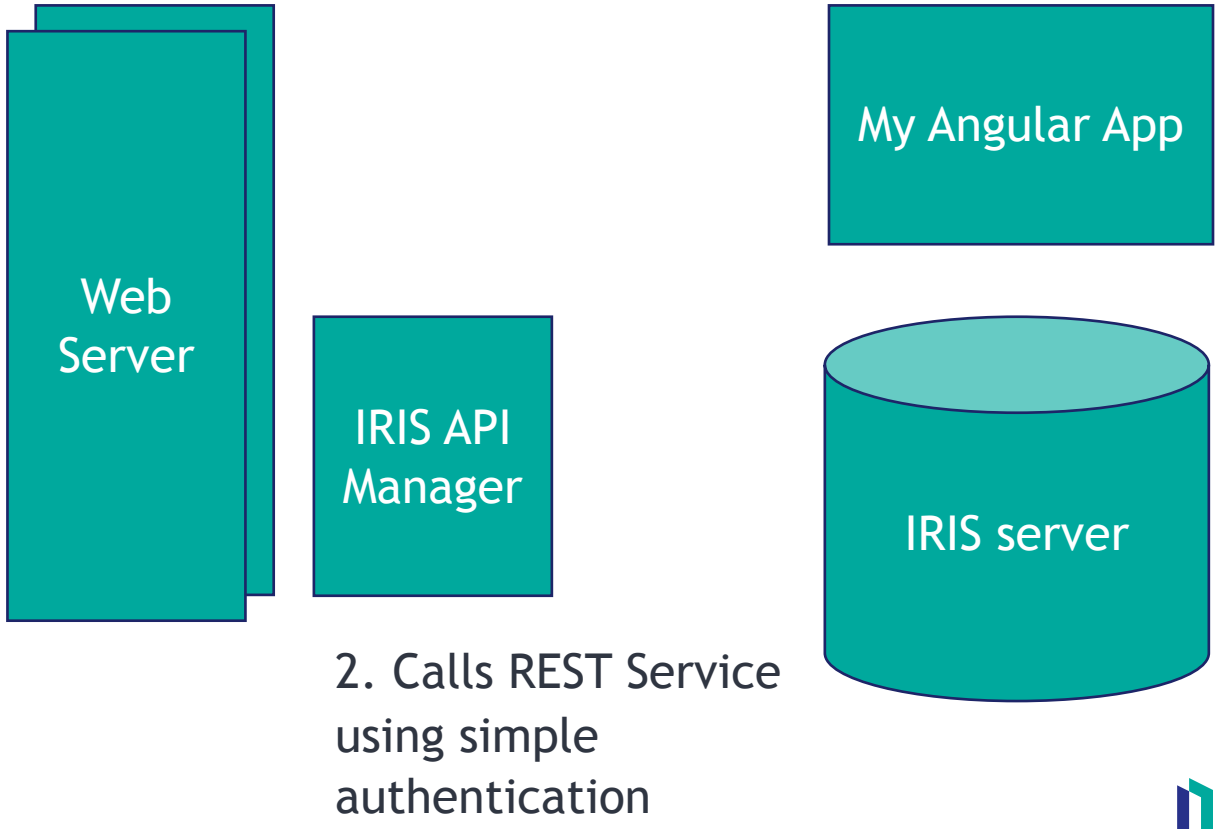


2. Calls REST Service using simple authentication

 © InterSystems Corporation. All rights reserved. |

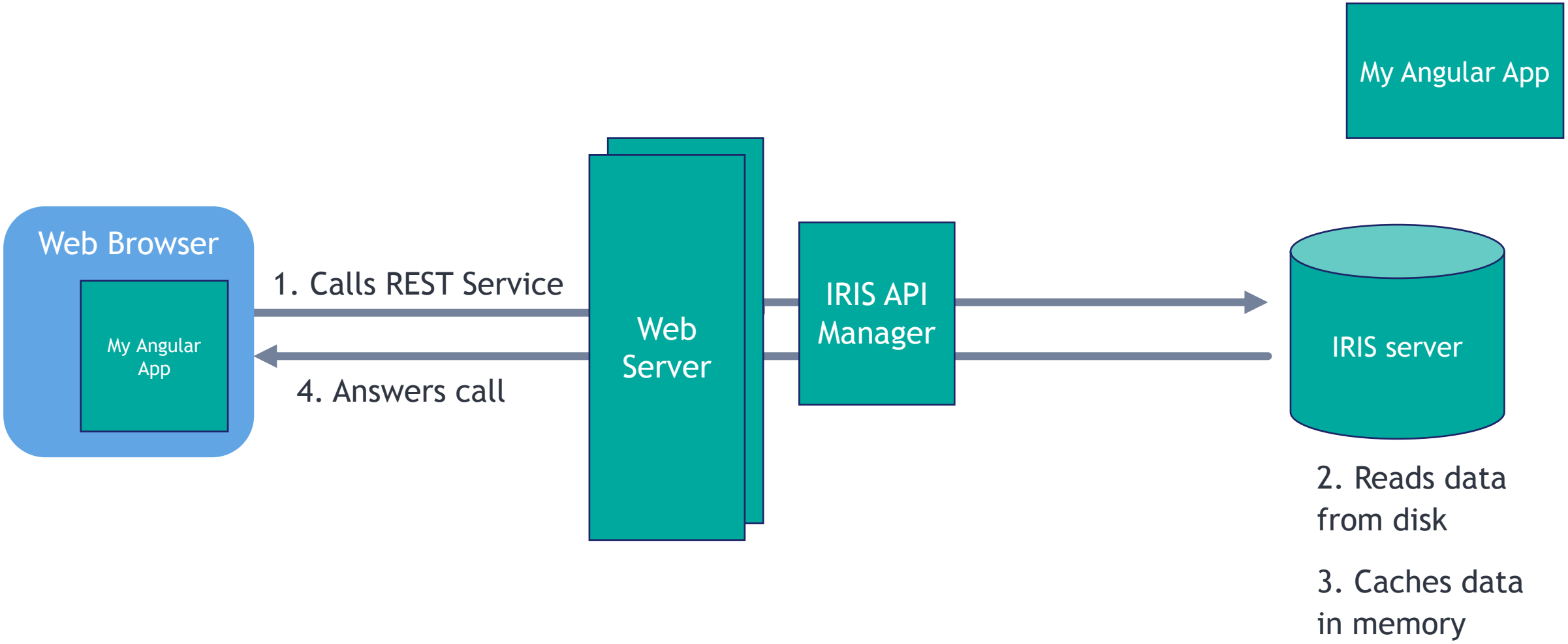
Caching on the Browser





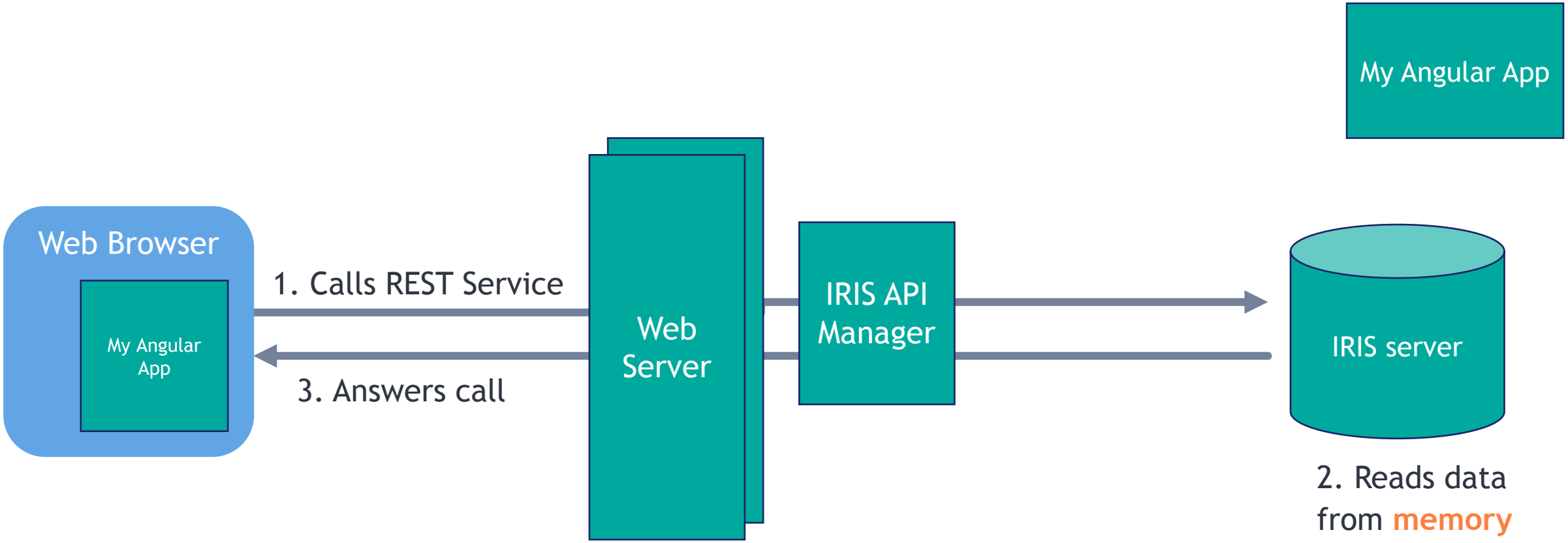
 © InterSystems Corporation. All rights reserved. |

Data Caching on the Server (1)



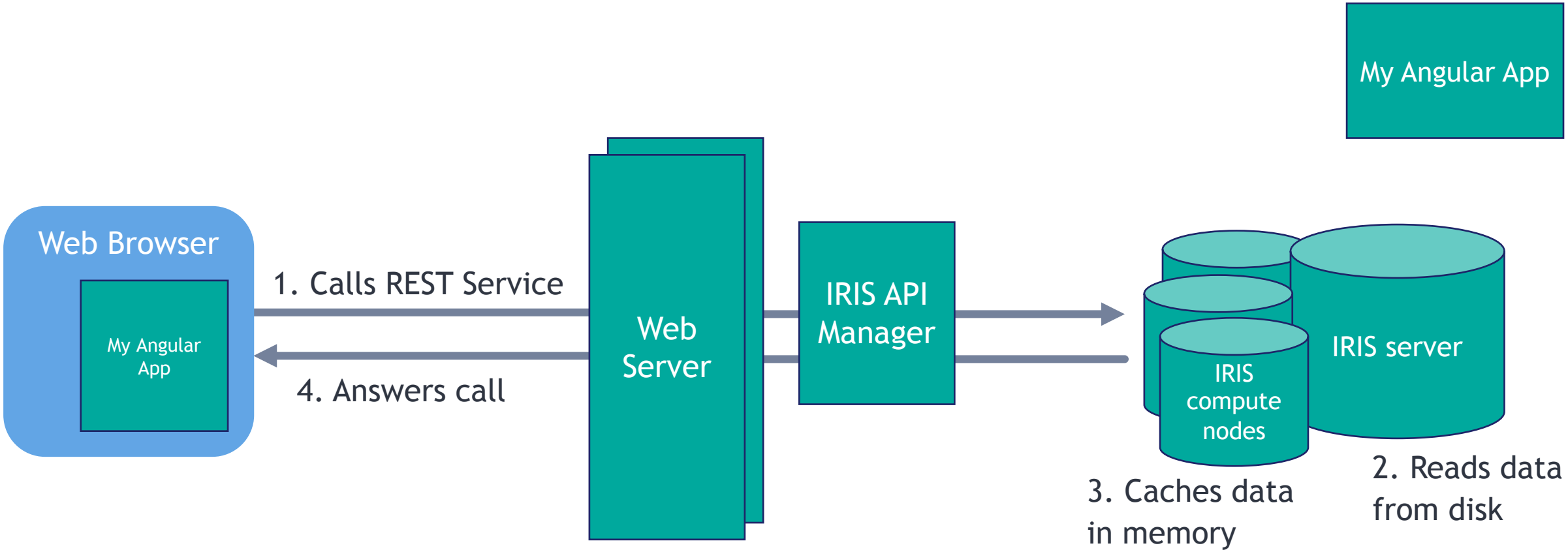
 © InterSystems Corporation. All rights reserved. |

Data Caching on the Server (2)



 © InterSystems Corporation. All rights reserved. |

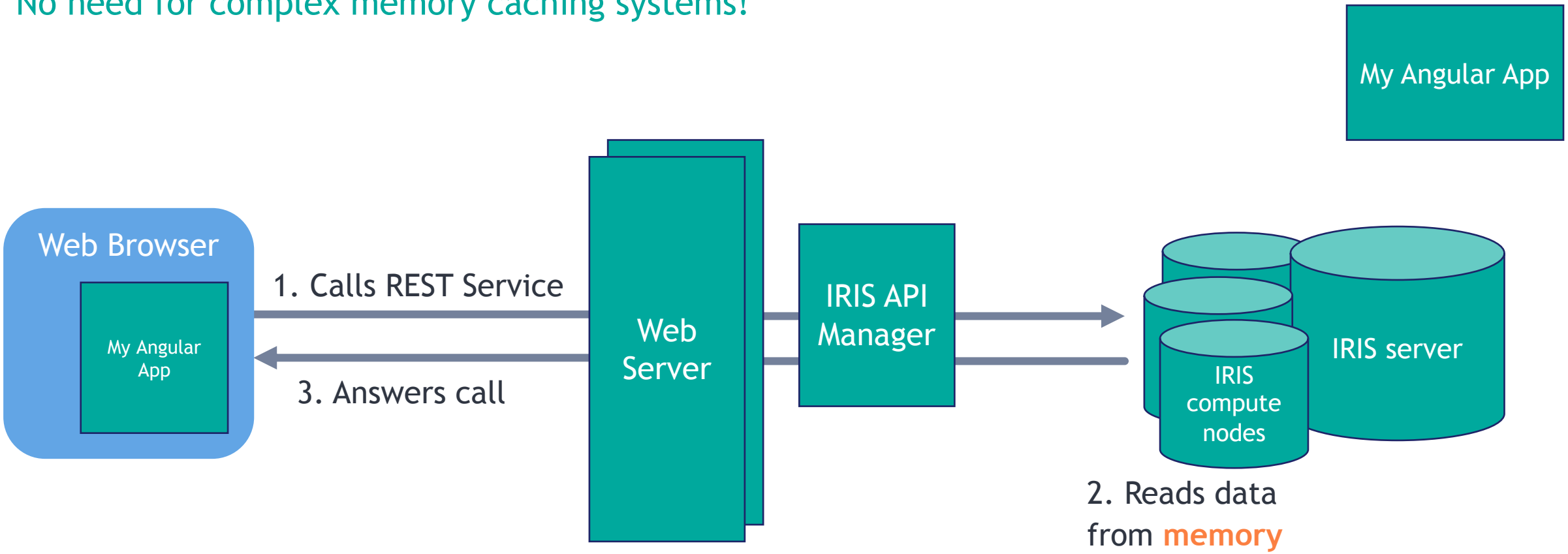
Data Caching on Compute Nodes (ECP)



 © InterSystems Corporation. All rights reserved. |

Data Caching on Compute Nodes (ECP)

No need for complex memory caching systems!



 © InterSystems Corporation. All rights reserved. |

In summary…

1. Design your service around its resources.
2. Your service should be stateless.
3. Use request forwarding to modularize your service.
   - Maintain multiple versions of the interface.
4. Don’t repeat yourself.
5. For handling errors, use exceptions.
6. User a Web Server and configure it with HTTPS!
7. Use IRIS API Manager for security and monitoring of your REST APIs!
8. Allow clients to cache REST response data if appropriate.
9. Use IRIS native memory caching

 © InterSystems Corporation. All rights reserved. |

Questions?

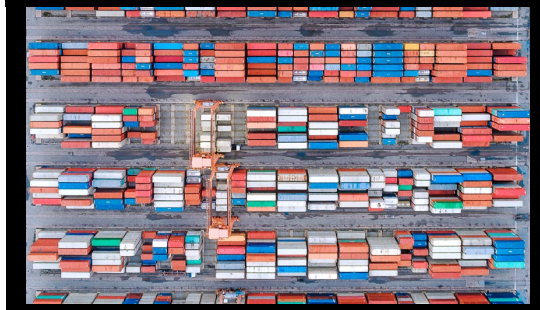
 © InterSystems Corporation. All rights reserved. |

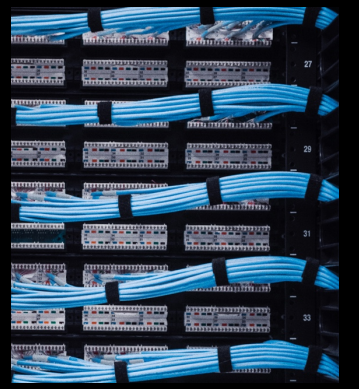
Thank you!

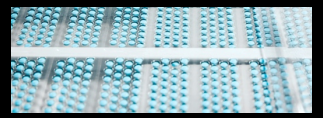
InterSystems®
Health | Business | Government

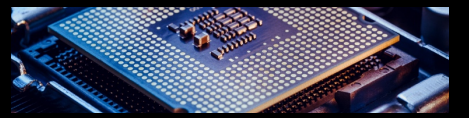
Amir Samary

Manager, Solution Architecture









JSON

```
{
    "customer": "Amir",
    "startLocation": "Brazil",
    "destination": "Boston",
    "startDate": "2018-02-01",
    "endDate": "2018-02-10"
}
```

 © InterSystems Corporation. All rights reserved. |

Dynamic Objects — Object to JSON

```
Set obj = {
            "customer": "Amir",
            "startLocation": "Brazil",
            "destination": "Boston",
            "startDate": "2018-02-01",
            "endDate": "2018-02-10"
          }
```

```
Write obj.%ToJSON()
```

 © InterSystems Corporation. All rights reserved. |

Dynamic Objects — JSON to Object

```
##class(%DynamicObject).%FromJSON(jsonStream)
```

 © InterSystems Corporation. All rights reserved. |