Σημασιολογικός και Κοινωνικός Ιστός

Διάλεξη 07 – REST API

Γεώργιος Δημητρακόπουλος
dimitrakopoulos@ionio.gr

Τμήμα Πληροφορικής
Ιόνιο Πανεπιστήμιο

REST API

- API : *application programming interface*
- REST: *representational state transfer*

- A REST or RESTfull API is an API that conforms to the design principles of the REST.

- 6 principles

REST principles

1. **Uniform interface**. All API requests for the same resource should look the same, no matter where the request comes from. The REST API should ensure that the same piece of data, such as the name or email address of a user, belongs to only one uniform resource identifier (URI). Resources shouldn’t be too large but should contain every piece of information that the client might need.

2. **Client-server decoupling**. In REST API design, client and server applications must be completely independent of each other. The only information the client application should know is the URI of the requested resource; it can't interact with the server application in any other ways. Similarly, a server application shouldn't modify the client application other than passing it to the requested data via HTTP.

3. **Statelessness**. REST APIs are stateless, meaning that each request needs to include all the information necessary for processing it. In other words, REST APIs do not require any server-side sessions. Server applications aren’t allowed to store any data related to a client request.

4. **Cacheability**. When possible, resources should be cacheable on the client or server side. Server responses also need to contain information about whether caching is allowed for the delivered resource. The goal is to improve performance on the client side, while increasing scalability on the server side.

5. **Layered system architecture**. In REST APIs, the calls and responses go through different layers. As a rule of thumb, don’t assume that the client and server applications connect directly to each other. There may be a number of different intermediaries in the communication loop. REST APIs need to be designed so that neither the client nor the server can tell whether it communicates with the end application or an intermediary.

6. **Code on demand (optional)**. REST APIs usually send static resources, but in certain cases, responses can also contain executable code (such as Java applets). In these cases, the code should only run on-demand.

REST API

- REST APIs communicate via HTTP requests to perform standard database functions like creating, reading, updating, and deleting records (also known as CRUD) within a resource.

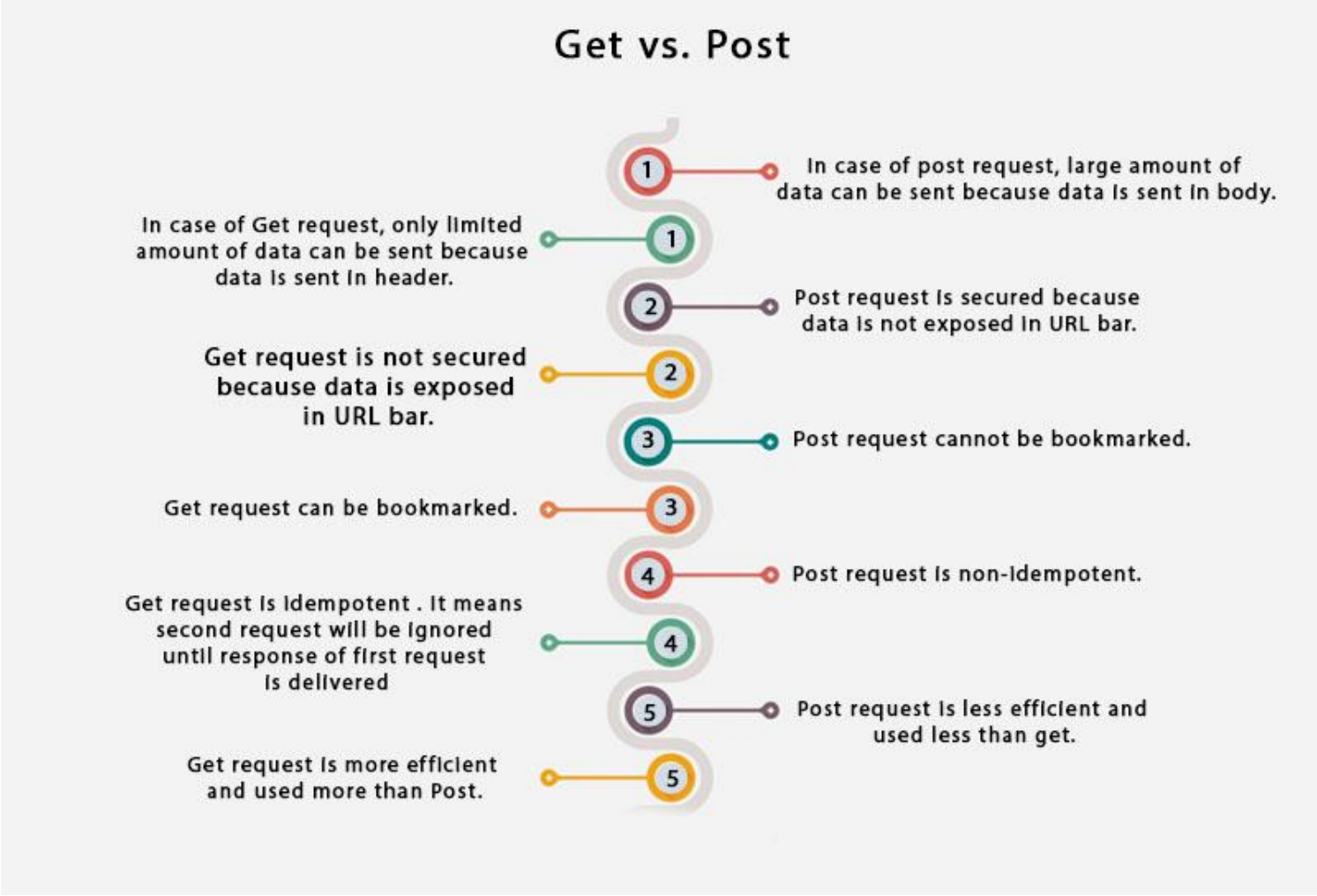
- The state of a resource at any particular instant, or timestamp, is known as the resource representation. This information can be delivered to a client in virtually any format including JavaScript Object Notation (JSON), HTML, XLT, Python, PHP, or plain text. J

REST API – μορφή URL

- https://server.com/api/users/create
- https://server.com/api/users/list_all
- https://server.com/api/users/get/1
- https://server.com/api/users/edit/1
- https://server.com/api/users/delete/1

GET, POST requests

<form action="test.php" method="**POST|GET**">



REST API

- Μπορεί να γίνει με GET, POST
- Συνήθως χρησιμοποιουνται και άλλα είδη request: PUT, DELETE

POST	CREATE
GET	READ
PUT	UPDATE
DELETE	DELETE

Πρακτικό κομμάτι:

- Μπορεί να χρησιμοποιηθεί οποιαδήποτε γλώσσα (JAVA, PHP, Python κλπ)
- Εδώ:
- Backend: node js
- Frontent: javascript
- Test: με εφαρμογές όπως POSTMAN (postman.com)

Π.χ. Nodejs server

```
create_customer = function(customer){
        return new Promise(function(resolve, reject){
                var query = "INSERT INTO `customer` (`Postal_code`, `Contact_Number`, `Address`, `Email`) VALUES ('"
+ customer.postal_code + "', '" + customer.contact_number + "', '" + customer. address + "', '" + customer.email + "')";

                con.query(query, function (err, result) {
                        if (err) {
                                return reject("Error");
                        }else{
                                resolve({"insertId":result.insertId});
                        }
                });
        });
}


app.post('/customers', function (req, res) {
        data = req.body;
        create_customer(data).then(function(result){
          res.send(result);
  });
})
```

Ορίζεται τι κάνει η εφαρμογή όταν δεχτεί post request στο end-point /customers

Δοκιμή back-end με POSTMAN

- Αποστολή δεδομένων (JSON μορφή) με POST request στο localhost:8081/customers