SESSION (REST API)

Need for Data Interchange/Exchange between different Applications

Different Scenarios :

I) Loan Application Software needs CIBIL Score/Credit Score of the customers before sanctioning the loan .

i.e. Needs data of the Customers from other Banks , finance lenders , Credit Card Companies etc.

II) Shopping Cart Application needs Payment Data from Payment Gateways like Razor, HDFC etc. which are third party applications

III) Need data from Social Networking sites like Twitter , Facebook or Youtube to be included in your website

IV) Microservices which are 1 Application's different modules/parts spread over different servers at different locations need to transfer data from 1 module at 1 location/server to another module at other location/server

V) Dot Net Application , Java Application , PHP Application etc needs to be merged/integrated for 1 Customer . This requires data transfer/interchange from Dot Net Application to Java/PHP Applications & vice versa.

Solutions Available

1) Data in JSON (Javascript Array) format used in REST APIs (for us Controller or Java Class Methods which return data).

2) Data in XML format used in SOAP based Webservices(for us Controller or Java Class Methods which return data).

3) Other formats like EDI etc...

JSON is the most popular Data Interchange format currently .

Why JSON is preferred over XML for Data Interchange ?

1) Controller method :

```
public int sum(int i , int j)
   {
```

```
                return (i+j);

        }
```

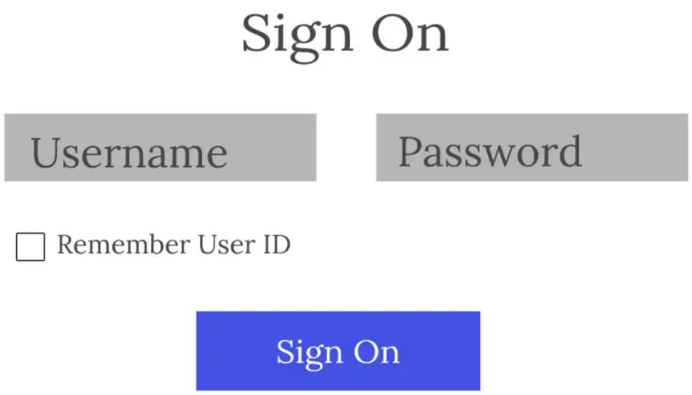
XML output :

```
<Sum>
  <result>10</result>
</Sum>
```

REST API output :

```
{
  "result":10
}
```

2) Login Form



JSON for a login page

```
{
    "username": “Shubham”,
   "password": “xyz@123#”
}
```

Login form in XML

```
<UserLogin>
  <Username>Shubham</Username>
  <Password>“xyz@123#”</Password>
</UserLogin>
```

JSON Basics

JSON stands for JavaScript Object Notation

JSON Syntax Rules

Data is in key/value pairs

Data is separated by commas

Curly braces hold objects

Square brackets hold arrays

eg:

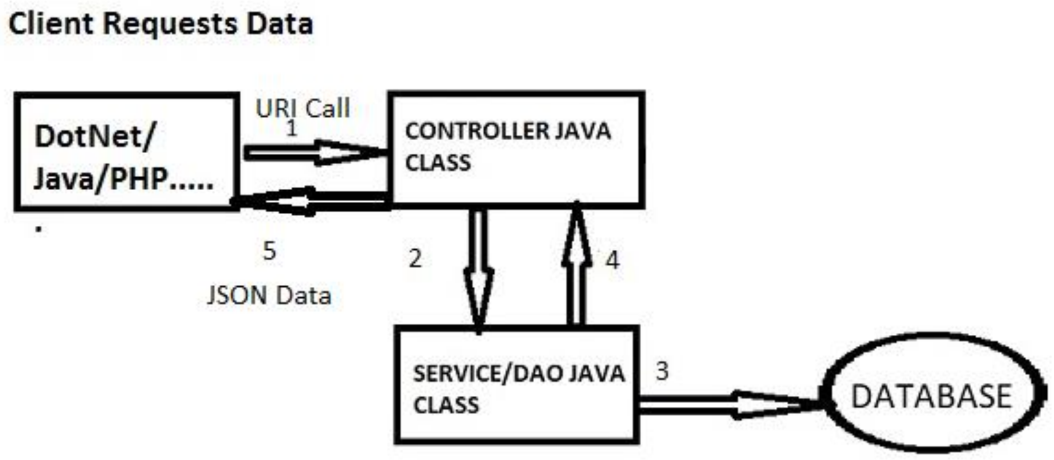
```
{
   "username": “Shubham”,
   "password": “xyz@123#”,
    "age":30,
     "employee":{"name":"John", "age":30, "city":"New York"},
     "employees":["John", "Anna", "Peter"],
      "sale":true,
      "middlename":null
}
```

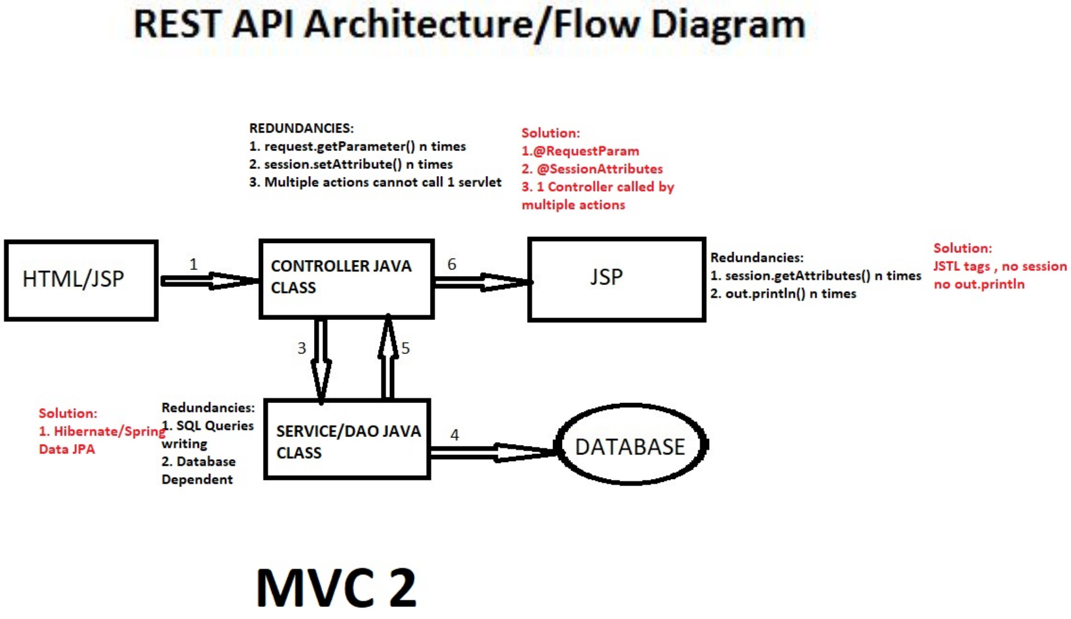
JSON Values

In JSON, values must be one of the following data types:

1. a string
2. a number
3. an object
4. an array
5. a boolean
6. null

REST API Architecture/Flow Diagram




REST API Architecture/Flow Diagram
REDUNDANCIES:
1. request.getParameter() n times
2. session.setAttribute() n times
3. Multiple actions cannot call 1 servlet
Solution:
1.@RequestParam
2. @SessionAttributes
3. 1 Controller called by multiple actions
HTML/JSP
1
CONTROLLER JAVA CLASS
6
JSP
Redundancies:
1. session.getAttributes() n times
2. out.println() n times
Solution:
JSTL tags , no session no out.println
3
5
Solution:
1. Hibernate/Spring Data JPA
Redundancies:
1. SQL Queries writing
2. Database Dependent
SERVICE/DAO JAVA CLASS
4
DATABASE
MVC 2

For Coding, NO JSPS/views required , only methods required in REST API

Program

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.servlet.ModelAndView;

import com.examples.beans.Employee;
import com.examples.repository.EmployeeRepo;

@RestController
public class EmployeeRestController {
@Autowired
Employee empbean;

@Autowired
EmployeeRepo empRepo;


@RequestMapping("rest")
public String displayIndex()
{
System.out.println("Index called");
return "index";
}

@GetMapping("restemp")
public String displayEmployeeForm()
{
System.out.println("EmployeeForm called");
return "employee";
}

@PostMapping("restemp")
public Iterable<Employee> processEmployeeData(@RequestBody Employee empBean)
{
System.out.println("processEmployeeData called");

empRepo.save(empBean);

Iterable<Employee> empItr = empRepo.findAll();
//Optional<Employee> optEmp= empRepo.findById("");

return empItr ;

}
```

}

REST API Annotations :

@RESTController : Use at Controller class level

@ResponseBody : Use at method level to return value as response

@Request Body : Use at method parameter level like @ModelAttribute

@PathVariable : Use at method parameter level like @RequestParam

SESSION 21 ASSIGNMENT

Write a Program to create Bank Controller class(Use @RESTController) , interface EmployeeRepo extends CrudRepository<Bank, Integer>,Bank Java Bean Class

- Write 3 methods with get Request in Bank Controller class.
- 1st method getBank mapped to url="bank" return String "Bank"
- 2nd method getBank mapped to url="bankbean" return Bean "bankBeanObj" populated with hard coded data.
- 3rd method getBankList mapped to url="banklist" return list of Bank Beans "bankBeanlst" populated with hard coded data.
- Generate json objects for above methods
- Bank Bean Java class should have fields like Name,Address,MobileNo

  Write CRUD Operations as below :

- Now create methods saveBank , updateBank , deleteBank, getBankById , getAllBanks
- Use Post,Put,Delete,Get mappings for above methods .
- Use @PathVariable, @RequestBody wherever required .
- Call above methods through PostMan Tool.