### 1. SOAP Vs REST

### 1. Protocol vs. Architectural Style

* SOAP (Simple Object Access Protocol) is a protocol with strict rules and standards. It defines a specific set of message formats, and communication relies on XML-based messages and a formal structure.
* REST (Representational State Transfer) is an architectural style rather than a protocol. It provides guidelines and principles for building web services, but it does not enforce any specific message format, leaving flexibility in communication methods.

**2. Message Format**

* SOAP uses XML as its message format, which requires strict adherence to a specific structure. The messages are wrapped in an envelope that contains both the header and body elements.
* REST supports multiple message formats like XML, JSON, HTML, and plain text. JSON is commonly used in RESTful APIs, making it lighter and easier to parse.

**3. Communication**

* SOAP typically uses HTTP, SMTP, TCP, or other protocols for communication. However, it is tightly coupled with XML messaging and tends to have a higher overhead due to its verbosity and strict message standards.
* REST relies on HTTP/HTTPS and uses standard HTTP methods such as GET, POST, PUT, DELETE, and PATCH to perform CRUD (Create, Read, Update, Delete) operations on resources.

**4. Statefulness**

* SOAP is generally stateful, meaning it can maintain a session or track the status between different requests, typically through the use of mechanisms like WS-Security or SOAP headers.
* REST is stateless, meaning each request from a client to a server is treated as independent. The server does not store any session information, and every request must contain all necessary data.

**5. Error Handling**

* SOAP has built-in error handling through the use of standard SOAP fault messages, which provide detailed information about the error and help with debugging.
* REST uses HTTP status codes (e.g., 404 for not found, 500 for internal server error) for error handling. While it can be less descriptive than SOAP's fault messages, it is simpler and often more efficient.

### 2. Create API documentation for the

1. [https://katalon-demo-cura.herokuapp.com/](https://katalon-demo-cura.herokuapp.com/" \t "_blank) -

Registration:

**Objective -** This API is used to login to the katalon-demo-cura.herokuapp.com application with the valid and invalid user.

**URL** - [https://katalon-demo-cura.herokuapp.com/profile.php#login](https://katalon-demo-cura.herokuapp.com/profile.php%23login)

**Login API** - POST

## Login API - POST

**URL** - <https://katalon-demo-cura.herokuapp.com/authenticate.php>

**HTTP Metho**d - POST

**Body/ Payload** -

**username: John Doe**

**password: ThisIsNotAPassword**

**Headers**

Content-Type: text/html; charset=UTF-8

**Auth:  NO**

**Response - Validation**

1. <https://app.vwo.com/> - Login API

**Objective -** This API is used to login to the vwo.com application with the valid and invalid user.

**URL** - <https://app.vwo.com/login>

**Login API** - POST

## Login API - POST

**URL** - <https://app.vwo.com/login>

**HTTP Metho**d - POST

**Body/ Payload** -

{username: "mdashwaqamer@gmail.com", password: "RcQ4WqQdd@67jFK", remember: false,…}

password: "RcQ4WqQdd@67jFK"

recaptcha\_response\_field: ""

remember: false

username: "mdashwaqamer@gmail.com"

**Headers:** application/json;charset=UTF-8

**Auth:  NO**

**Response - Validation**

1. Ref - Opencart Login and Registration

**Registration**:

**Objective -** This API is used to Register to the awesomeqa.com application by providing valid details of new user.

**URL:** <https://awesomeqa.com/ui/index.php?route=account/register>

**Login API:** POST

**Body/ Payload**

customer\_group\_id: 1

firstname: Ashwaq Amer

lastname: Mohammed

email: mdashwaqamer@gmail.com

telephone: 9032243416

password: DtE@a4ML22F@@r2

confirm: DtE@a4ML22F@@r2

newsletter: 0

agree: 1

**Header:** text/html; charset=UTF-8

**Auth:  NO**

**Response - Validation**

**Login:**

**Registration**:

**Objective -** This API is used to login to the awesomeqa.com application with the valid and invalid user

**URL:** <https://awesomeqa.com/ui/index.php?route=account/login>

**Login API:** POST

**Body/ Payload**

email: mdashwaqamer@gmail.com

password: DtE@a4ML22F@@r2

**Header:** text/html; charset=UTF-8

**Auth:  NO**

**Response - Validation**