EJONS International Journal on Mathematic, Engineering and Natural Sciences | ISSN 2602 - 4136

Article Arrival Date	Article Type	Article Published Date
17.02.2021	REVIEW Article	20.03.2022

Doi Number: http://dx.doi.org/10.38063/ejons.608

RESTFUL WEB SERVİS TESTLERİ HAKKINDA BİR İNCELEME

A REVIEW OF RESTFUL WEB SERVICE TESTS

Dr. Ecem İREN
Computer Programming Program, İzmir Kavram Vocational School, İzmir, Turkey
ORCID ID: 0000-0002-4678-6972

Gizem İREN

Department of Computer Engineering, Ege University, İzmir, Turkey
ORCID ID: 0000-0003-2827-7012

Prof. Dr. Aylin KANTARCI
Department of Computer Engineering, Ege University, İzmir, Turkey
ORCID ID: 0000-0001-7019-1613

ÖZET

Yazılım testi, tasarım, uygulama, bakım ve yönetim gibi birçok teknik ve teknik olmayan alanı kapsayan geniş bir alandır. Test süreçlerinin müşterilerin beklentilerinin karşılanmasında önemli bir rol oynadığını söyleyebiliriz. Geliştirilen yazılımın müşterilerin talep ettiği ihtiyaçları karşılayıp karşılamadığını görmek için yazılım testleri yapılmaktadır. Testler yardımıyla yazılımdaki mevcut hatalar tespit edilip müşteri gereksinimlerine göre düzeltilmektedir. Ek olarak, günümüzde büyük yazılım projelerinde yaygın olarak kullanılan web servisleri uygulamaların veritabanından veri çekebilmesini sağlayan önemli bir bileşen olarak düşünülebilir. Bu nedenle bilgi sistemlerinde kilit bir rol oynayan web servisleri, ürün müşterilere sunulmadan önce detaylı olarak test edilmelidir. Bu çalışmada RESTful web servis testi konusu kapsamlı olarak incelenmiş ve test için kullanılan otomasyon araçları bazı kriterler ele alınarak karşılaştırılmıştır.

Keywords: Yazılım Testi, RESTful Web Servis Testi, Fonksiyonel Test, Performans Testi, Otomasyon Araçları

ABSTRACT

Software testing is a broad field covering many technical and non-technical areas such as design, implementation, maintenance and management. We can say that testing processes play an important role in fulfilling expectations of customers. Software tests are carried out to see if developed software meets the needs demanded by customers. With the help of tests, existing defects in software can be detected and corrected according to customer requirements. Furthermore, web services are widely used in large software projects nowadays and they can be thought as significant components that enable applications to extract data from database. Since it plays a key role in IT systems, testers should attach importance to test web services before product meets with its customers. In this study, we review RESTful web services by giving comprehensive information. Also, automation tools of Restful web services are examined and compared in terms of multiple criteria.

Keywords: Software Test, RESTful Web Service Test, Functional Test, Performance Test, Automation Tools

1. INTRODUCTION

Software testing has become one of the most important topics for many companies around the world today. Software testing is a field in software engineering covering many technical and non-technical areas such as maintenance, process and management, specification, design and implementation. Software testing processes play vital role for the requirements of both institutions and customers to be meet correctly with their expectations. In the digitalizing business world when the costs and potential security risks are considered, software fault detections and fault-oriented solution applications are very important for companies in the context of software testing [1]. With the development of fourth-generation languages (4GL), which accelerates the processes of application projects, the time devoted to testing has increased progressively. Therefore, it is likely that the applications of the testing process will become much more important in computer science in the future [2].

The rapid advancement of technological developments in our age has affected the web world in an inevitable way. When we look at global statistics, the number of internet users has risen by two times compared to one which was 10 years ago [3]. Aside from the increase in Internet users, the number of currently designed websites has reached quite striking numbers. Research shows that interest of users in websites significantly reduced within 3-4 seconds. Potential reasons for this situation consist of poor design, complex structure or lack of information [4] As a result, the subjects of fast, reliable and accurate operation of websites are becoming increasingly important. Considering the flow of information in browsers, it is clear that high priority should be given to the testing processes by information technology (IT) companies. When taking into account costs of time, money and effort, it will not be difficult to understand the importance of software automation tests.

The main purpose of software testing is to measure the quality of the software by focusing on deficiencies in order to optimize the software [5]. In other words, software tests basically check the compatibility of the actual results with the expected results and aim to produce solutions for the errors. Expected results are based on parameters defined in the requirement specification which is previously determined. However, actual results are based on real parameters obtained after the test process in a controlled test environment. Generally, a completed software test must be capable of covering the entire life of a software product [1]. Beside this, web services are widely used in the large software projects nowadays. Applications can communicate each other without knowing how other is designed technically with services. For example; applications can store data in different formats or be implemented in different programming languages. At this point, web services become useful to integrate them by providing a common bridge between applications. Hence, testing of these structures before deployment can be considered as crucial because they play key role in the systems and should work properly. Also, selecting the suitable test automation tool is the key point for the test specialists to accomplish testing processes successfully.

Our article is organized as follows: A brief information is given about the principles of test processes, testing types and web service testing in the second section. In the third section, RESTful web services are introduced. Fourth section includes a case study of a RESTful web service testing with Rest Assured tool. RESTful service-based automation tools are presented

in the fifth section. Results and discussions and conclusions take part in sixth and seventh sections.

2. LITERATURE REVIEW

There exist studies related with web service testing topic. Hussain et al. compare different service testing tools named as SoapUI, JMeter and Storms in terms of average response time, throughput and data amount per second processed during testing. Results show that while SoapUI performs better than other tools in average response time, JMeter has the greatest throughput and gives good performance in amount of data being transferred in a second [6]. Emektar et al. give information about Postman tool by presenting a case study related with testing of web services in an insurance company [7]. Sualim et al. compare TestComplete, Selenium and Watir under a study. The necessity of user experience for Selenium has been one of the highlighted issues. Arcuri introduces a new automation tool helping practitioners in white-box and black-box testing of RESTful (Representational State Transfer) web services. To compare performance of the tool, experiments are carried out and white-box testing is found to be better at fault detection than black-box testing. It is mentioned that the tool is also the only one which covers white-box test case generation for RESTful services [8]. Kumar et al. analyze web service test tools such as SoapUI Pro, Wcf Storm, Apache JMeter, Wizdl, SOA Cleaner and SOAPSonar Personal by different metrics. SOAPSonar outperforms the rest of other tools in average response time. Similar to the previous study [6], JMeter can process more number of bytes among the tools [9].

3. SOFTWARE TESTING

3.1. Software Test Life Cycle

Software testing process accounts for approximately 50% of the total time in the software development life cycle [10]. From a budget point of view, the test process cost covers approximately 40% of the budget [11]. The time of the test process depends on the algorithm used, programming language, how many lines of code, function points, external and internal interfaces. In general, software tests should be developed in the form of structures which should be included in the life cycle and touch the system at every stage. In other words, software testing should not be a separate stage in system development, but should be applicable in design development and maintenance stages [12]. The software test life cycle can be summarized as in Figure 1.

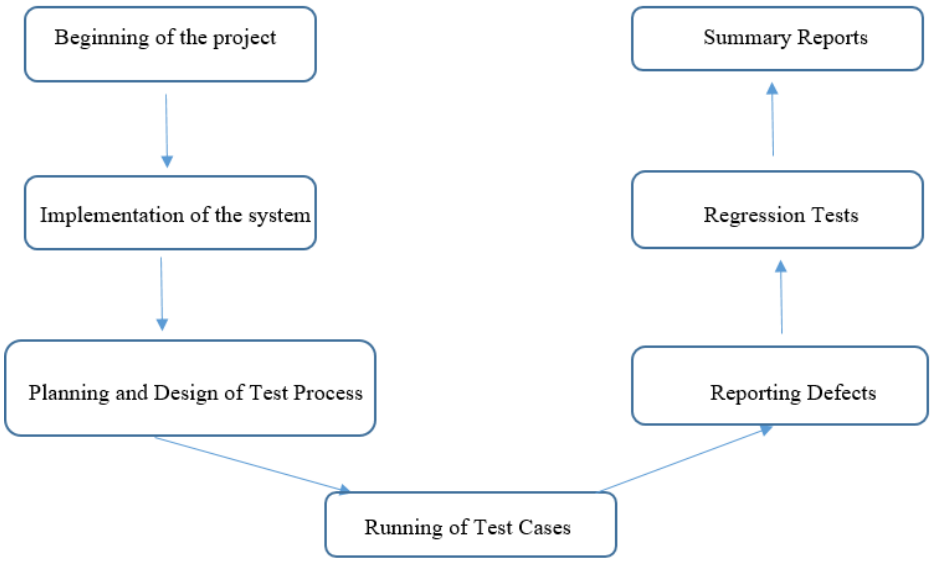

Figure 1. Software testing life cycle activities [13]

The process that starts with the beginning of the project continues in the form of defining the test plan, creating and realizing test scenarios, reporting and analysis activities. Planning stage includes high-level test plans, planning of quality targets, reporting procedures, determination of acceptance criteria and includes activities such as planning the timing of the project testing process [14]. The analysis process includes steps such as functional verification according to business requirements, writing test scenarios, developing test cycles with matrices and timelines, if available, defining test scenarios that can be automated, determining the procedures for backup, restore and verification documents. The priority issue is very important in software testing. It can be said that it is impossible to perform a thorough test. Test cases are prioritized depending on importance level. Test results should be independent of the test device so that the testing process and results can be considered consistent and unbiased, reproducible. There is an information flow process based on the activities in the life cycle. The information flow of the test process is summarized in Figure 2.

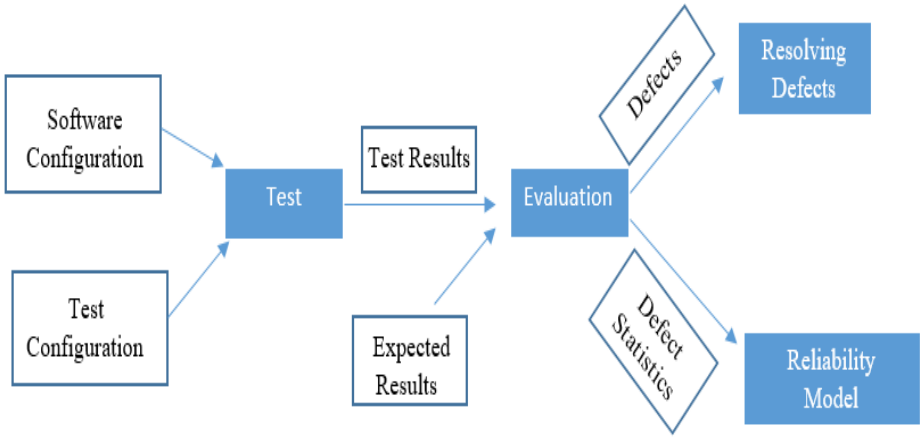

Figure 2. Information flow of test process [12]

The testing process, which starts under the guidance of software and test configurations, goes into an evaluation process in which the actual results and expected results are compared. As a result of the evaluation, investigations are carried out for the relevant corrections to solve the defects. Error rate data calculated from defect statistics is entered as a parameter to the reliability model and then reliability level is estimated in the model [2].

Different types of test procedures and applications are widely used in software testing. Analyses can be divided into two categories: static and dynamic analyses [2]. Static analysis ensures software quality without any application and includes activities performed to identify and predict. These activities include code checking, program analysis and model checking. Dynamic analysis includes activities to determine software quality under real or simulated conditions of real data through an application. Activities such as the use of test procedures, automation activities, analysis of test scripts through the application are within the scope of dynamic analysis.

Unit test includes applications related to the verification of the smallest units in software processes. In software development processes, unit tests are defined at the design stage and can be performed for more than one unit at the same time. It is performed by software developer. Although it is the first stage of the dynamic testing process, it is the most important stage of this process in terms of finding and correcting errors early. In this test, special functions or code modules (functions, data structures, objects, etc.) are tested.

Integration Testing is a test to ensure whether different components of an application work together in harmony. These types of tests are used especially in the testing of client / server applications and distributed systems.

Regression Testing is performed after the necessary changes and fixations are made in the application. This ensures that the problems identified in previous tests are resolved and new errors will not occur.

Performance Testing is often used synonymously with "load test". It can also be defined as the functionality test of a system under severe conditions such as unexpected heavy loads, too much increase of certain actions and demands, very intense numerical operations, very complex inquiries, etc. Tests which are used to determine at what point the system response will reduce or be unresponsive can be served as an example of performance testing [1, 13, 15, 16].

Security Testing is a testing type ensuring security requirements as confidentiality, integrity, availability, authentication, authorization in the application. Confidentiality means that information is hidden from unauthorized party. Integrity guarantees sensitive information is not distorted through any alteration maliciously or accidentally. Availability assures that authorized side can reach the resources in case of any request. Operations as verifying the identity of parties and validating the data falls into the scope of the authentication. Lastly, giving the users to some privileges for accessing a particular data is related with authorization topic. Furthermore, possible vulnerabilities can be exploited in the security testing by acting as a hacker in the system.

Black Box (Functional) Testing is also known as functional test in which details of data elements and programmatic structure, coding technique of the software are not known and accessible to the user. In other words, the system being tested is perceived as a black box. Testing is operated from the user interfaces to only control that whether functionalities of the software

White Box Testing is a testing of both the source code and of the project. Such tests depend on knowledge on the internal logic of the application code. Statements in software code, flow controls, conditions and etc. elements are tested. [2, 15-17].

2.2. Comparison of Manual and Automation Testing

In manual test, a tester runs the application and tries a number of scenarios on the user interfaces (UI) using different inputs to test whether the application exhibits the expected behaviour and it meets the business and technical requirements [22]. Manual tests require physical effort and

time because test results should be recorded by the tester. Also, it is possible to make a mistake when test cases become complex. However; automation testing is faster and more efficient than manual testing. It is quite advantageous over manual testing for long-term and repetitive tests. Also, reporting processes are carried out easily with reporting options of selected automation tool. Automation testing is reliable since test cases are run through test scripts and this leads to more accurate results. On the other hand, creating an automation test system by writing test scripts could take a long time. As long as the change requests are demanded by customers, the system should be refreshed. Therefore, it requires a constant maintenance [23]. Automation testing can be a good choice for the testing of large projects.

2.3. Key Points of Web Service Testing

When we talk about web services, security, functional and performance tests gain importance in this context. Because web services are challenged by the factors listed below:

- There is no guarantee that trustworthiness is ensured.
- Malicious users can attack to the web services to steal sensitive information by configuring different parameters.
- Designing input parameters of the clients in an unsuitable way and this situation leads sending invalid data to the service and returning some errors to the client. Moreover, service parameters may be written in wrong orders and again this prevents clients to reach the data successfully.
- In some cases, there could be high load of requests to the web services and this causes to decrease in the performance of them.

Above, first and second challenges are related with security testing. Web services should be tested to assure that application programming interface (API) is safe from external threats and dangers. Also, services should be checked and validated in terms of authorization subject while accessing to the resources. Functional testing handles the third challenge by controlling specific methods of web services and ensuring the API works correctly in the system. On the other hand, performance testing is dealing with the fourth challenge and tests are generally designed in three scenarios as testing the service in regular, maximum and overloaded traffics [8, 24].

3. OVERVIEW of RESTFUL WEB SERVICES

Web services can be considered as important components that enable applications to extract data from the database and transfer it to other applications. The user makes a HTTP request to the service with the application. It processes the service request and connects to the database and retrieves the crucial data. It then sends the relevant data back to the user application. Generally, they are categorized as simple object access protocol (SOAP) and RESTful web services. RESTful services can be easier to learn and use. While SOAP only use extensible markup language (XML) data type, RESTful can transfer different smaller data formats such as JavaScript object notation (JSON) or HTML. Hence, it works efficiently over the SOAP architecture. Some methods used to process through the RESTful web services are shown in Figure 3.

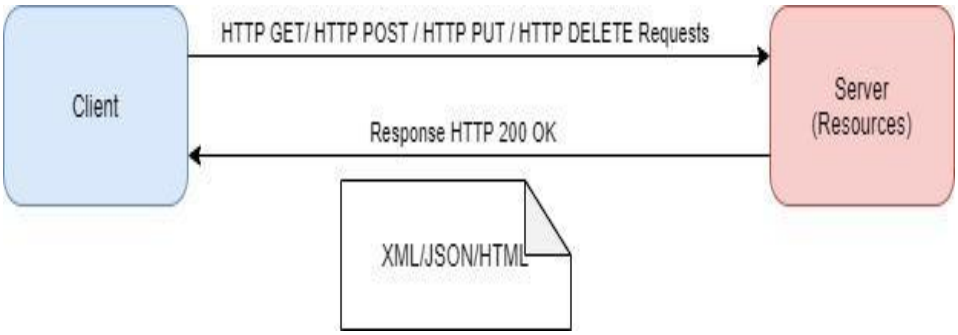

Figure 3. Architecture of RESTful Web Services

- **GET:** It is service method performing the reading process from the database into the application.
- **POST:** Service method inserting an item to the database.
- **DELETE:** Service method deleting an item from the database.
- **PUT**: Service method updating an item in the database.

RESTful APIs are built on four basic principles. These principles are described below:

Client – Server: This item is considered to be the first fundamental constraint of the REST architecture. According to this restriction, an application should be modelled as client - server. In other words, the application must have two basic components in the application such as front side (user interface) and back side (database). Client and server components can develop independently. This can be described as a benefit of this architecture. In addition; one server can serve multiple clients. On the other hand, clients having different technologies can communicate with the same server.

Stateless: According to this restriction, a server should not store the information of a client within itself. Each request sent by the client is not associated with previous requests made by the same client. Therefore, session information of the client is not available on the server side. When the client makes a request to the server, it carries the request information to the server each time. There is no need for the server to manage resources. The disadvantage of the structure is that adding the information needed by the server to each request increases network traffic and the verification of incoming requests puts an extra load on the server.

Cache: The responses sent by the server must contain some information about whether they will be cached on the client side. This is usually done with the header information sent in the reply. This restriction increases client throughput for cacheable responses. The client does not make repeated requests to the server and checks its cache for the corresponding response. This mechanism saves the bandwidth of the network and the processing power of the client.

Uniform Interface: This constraint allows each component of the service to develop independently and simplifies the service architecture. There are some principles that guide the uniform interface constraint.

- **Resource Based:** In the RESTful service structure, resources are defined using URIs in requests. Resources are conceptually different from representations sent back to the client. For example; although the server does not send the database to the client, it uses HTML, XML or JSON data formats representing database records in its responses.
- **Self-Explaining Messages:** Each message contains sufficient information on how to handle it. As an example, which parser to use is specified by the Internet Media type (MIME type).

- **Layered System:** According to this constraint of RESTful architecture, the system application should be in a layered structure. Each layer generally abstracts a certain function of the system. Layered software design is also a widely used technique outside of REST. This restriction reduces the overall complexity of the different components in the system.
- **Code on Demand:** This item can be considered as an optional restriction of the RESTful architecture. According to this restriction, a client can expand its functionality by downloading a certain code from the server. On the other hand, if the server sends code snippets to the client, it may bring security problems [25-27].

4. A CASE STUDY OF AUTOMATION OF A WEB SERVICE

In this section, a case study is examined by running different test cases of a web service with Rest Assured test tool to show code-based automation. Test cases includes checking the service methods like GET and PUT. Methods of the service are defined at first and then test cases are written in Java programming language. The scenarios are categorized under two operations listed below:

- Getting the users according to age criteria
- Inserting a new user

The getting of users via the web service according to age criteria is done as follows. Firstly, service method is created as shown in Figure 4.

```
@GET
@Produces(MediaType.APPLICATION_JSON)
@Path("/age/{age}")
public List<UserModel> start(@PathParam("age") int userAge) throws Exception {

    wso = new WebServiceOperations();
    businessUtil = new BusinessUtil();

    ResultSet myRs = wso.queryAllUsersByAge(userAge);
    businessUtil.addUser(myRs);
    return businessUtil.getUserList();
}
```

Figure 4. Definition of web service method filtering users by age

Different notations at the beginning of the service methods are also can be used. @Produces notation indicates format of the data to be returned by the method. In order to send the results to the web page in JSON data type, APPLICATION_JSON type should be selected as the media type. Finally, the @GET and @POST notations should be put specifically to indicate whether the method is a GET or POST method. The service method is tested in Figure 4. Since the age parameter will be used for this process, this parameter must be given in the method. In this context, appropriate parameter setting has been made with @PathParam notation. It has been checked that whether all users who are 25 years old are retrieved with the following web service method successfully. The related method has been tested in Figure 5.

```
public class TEST_GET_USERS_BY_AGE {
        @Test
        public void getAllUsersByAge()
        {
            //writing base URL
            RestAssured.baseURI = "http://localhost:8080/ServiceProject/wel

            RequestSpecification httpRequest = RestAssured.given();

            Response response = httpRequest.request(Method.GET,"/age/25");

            String responseBody = response.getBody().asString();
            System.out.println("Response body is "+responseBody);

            int statusCode = response.getStatusCode();
            System.out.println("Status code is "+statusCode);
            Assert.assertEquals(statusCode, 200);

            String statusLine = response.getStatusLine();
            System.out.println("Status line is "+statusLine);
            Assert.assertEquals(statusLine, "HTTP/1.1 200 ");
```

Figure 5. Testing the web service method associated with age criteria

First of all, the link of the web service to be tested is introduced in the program with the baseURI property of the RestAssured class. Then, in order to make an HTTP request manually to the web service, the given method of the RestAssured class is applied. This method returns a variable from RequestSpecification class type. This variable is the HTTP request variable and calls the *getAllUsersByAge* method of the web service class. When this method runs, it returns a service response. If the service method works successfully, the status code of the response becomes HTTP 200. The test process started with controlling of the status code. With the help of the equality method of the Assert class, it has been checked whether the response status code is equal to 200. Subsequently, the response status line is tested. The response status line informs the user about the HTTP Protocol version, status code and status message. If the service method worked well, the response status line is expected to be "HTTP / 1.1 200 OK". This control is also made with the equality method of the Assert class. Response body is defined as the data requested by the user from the web service. This data is shown in the JSON data format type. TestNG [28] tool has been applied to run the tests. It is an open-source automated test framework and allows software developers to write more flexible and powerful test cases. Test is operated with the TestNG framework producing results given in Figure 6 and it is seen that the service method works properly.


```
[RemoteTestNG] detected TestNG version 7.0.0
Response body is [{"ID":1,"age":25,"name":"gizem","surname":"iren"},{"ID":4,"age":25,
Status code is 200
Status line is HTTP/1.1 200
PASSED: getAllUsersByAge

===============================================
    Default test
    Tests run: 1, Failures: 0, Skips: 0
===============================================


===============================================
Default suite
Total tests run: 1, Passes: 1, Failures: 0, Skips: 0
===============================================
```

Figure 6. Definition of web service method for all users

Beside getting the records, we also examined the insertion and deletion scenarios in the study. Insertion is done with the method implemented as in Figure 7. Service URL of insertion operation is stated again with @Path notation. Parameters of the user to be inserted is represented with @PathParam. This method uses the database method of *insertUser*.

```
@POST
@Produces(MediaType.APPLICATION_JSON)
@Path("/post/{name}/{surname}/{age}")
public List<UserModel> start (@PathParam("name") String name,@PathParam("surname")
String surname, @PathParam("age") int age) throws Exception{

      wso = new WebServiceOperations();
      businessUtil = new BusinessUtil();

      ResultSet myRs = wso.insertUser(name,surname,age);
      businessUtil.addUser(myRs);

      return businessUtil.getUserList();
}
```

Figure 7. Definition of web service method inserting user to the database

Test script of insertion is implemented and user parameters are given with predefined *put* method of JSONObject class. Insertion request is made with *Post* method by stating the valid link of the web service method. At the end of the script, returned status code is compared with the expected one. Generally, status code becomes 201 when the insertion is accomplished properly (Figure 8).

```
public class TEST_INSERT_USER {

RestAssured.baseURI
="http://localhost:8080/ServiceProject/webService/getUserService";
        RequestSpecification request = RestAssured.given();

        JSONObject requestParams = new JSONObject();
        requestParams.put("Name","Ali");
        requestParams.put("Surname","Güven");
        requestParams.put("Age",20);

        request.body(requestParams.toJSONString());
        Response response = request.post("/post");
        String responseBody = response.getBody().asString();
        System.out.println("Response body is " +responseBody);

        int statusCode = response.getStatusCode();
        Assert.assertEquals(statusCode, "201");
}
```

Figure 8. Testing the web service method associated with insertion

The tests run successfully and the results are in Figure 9.

```
<terminated> TEST_INSERT [TEST_NG] C:Program Files\Java\jre1.8.0_161\bin\javaw.exe
[RemoteTestNG] detected TestNG version 7.0.0
Status Code is 201
PASSED: insertUser

===============================================
    Default test
    Tests run: 1, Failures: 0, Skips: 0
===============================================


===============================================
Default suite
Total tests run: 1, Passes: 1, Failures: 0, Skips: 0
===============================================
```

Figure 9. Test results of web service method associated with insertion

5. RESTFUL WEB SERVICE BASED AUTOMATION TOOLS

In this section RESTful web service automation tools are explained and compared in Table 1 seen in Appendix. Some metrics are taken into account while evaluating as listed below:

- Does the tool require a code knowledge?
- Does the tool support built-in parallel execution of test cases?
- Does the tool have cost free version?
- Does the tool support multiple languages to use in automation scripting?

• Does the tool support multiple platform testing such as web or mobile application with built - in methods?
• Is the tool used in various operating systems?
• Is the tool designed with reporting options?
• Is the tool capability of functional, performance and security testing?
•Does the tool allow users to run the automated tests in continuous integration environment?

When the tools are compared overall, Katalon Studio, JMeter, RedwoodHQ, ReadyAPI, Parasoft/SoaTest have more features than others. However, if we make an evaluation in terms of number of test types supported by the tool, SoapUI, JMeter, KarateDSL, ReadyAPI and SoapSonar supports functional, performance and security testing together. Therefore, they can be preferred for end-to-end testing of large-scale projects.

5.1. SoapUI Open Source

SoapUI Open Source is a tool for testing REST, SOAP, GraphQL applications. This tool can be used for functional, performance and security testing. However, testing of web and mobile UI cannot be utilized on it. Test scripts are written with Groovy or Javascript programming language. SoapUI Pro version allows testers to develop and analyze complex test cases. This version unfortunately doesn't support continuous integration and data driven testing with external sources and reporting options. But users can display test logs to compare results manually [29, 30].

5.2. Postman

Postman can be used in two different form such as desktop application or web version. It is easy to learn and no code knowledge is required. Tool has a good user interface and works based on setting API parameters for testing purposes. Testers can develop and store their test cases meaning that it supplies reusing of the test commands [29, 31].

5.3. Rest-Assured

Rest-Assured provides writing test cases only in Java programming language and it is an open-source tool. It has numerous built-in methods for testing process and helps creating test scenarios with behavior driven development testing syntax by including some keywords as "given/then/when" [29, 32].

5.4. Katalon Studio

Katalon Studio is designed for testing various platforms such as API, web, desktop and mobile applications. It is found to be easy to use for beginners and also has scripting options for automation. Moreover, it is possible to operate manual and automation tests. Beside them, data driven testing, powerful recording and built-in code assisting are some of the functionalities of the tool [29, 33].

5.5. JMeter

JMeter is a tool widely used for performance testing by the companies. It allows multithreading meaning handling of multiple requests of multiple users to the web service and dynamic HTML reporting, recording and replaying test results. JMeter has a user-friendly interface and command line options. It has an advantage of being an open-source tool and is preferred in load testing area [29, 34, 35].

5.6. RedwoodHQ

It is an open-source automation tool which can be integrated with other tools like Selenium, Appium, Silk etc. Users can create keyword or action driven test cases through a web interface. It has a capability of executing tests in parallel or multithreaded forms on the multiple machines. Furthermore, user has a chance of comparing test results of all executed test cases [35, 36].

5.7. Unirest

Unirest can be chosen highly by test specialists since it is known for supporting various programming languages such as Java, Node, Ruby, Python and Objective C. A detailed documentation of the tool exists for the explanation of REST methods [37].

5.8. HttpMaster

HttpMaster is a tool used for both development and testing purposes. It is implemented with many features like observing API responses, supporting custom API requests, data uploading, requesting particular items for executing and monitoring them, defining a request chain to reuse data (response body, header etc.) from the previous request, parameterizing input data with various values for the multiple web service calls [38].

5.9. RestSharp

RestSharp is very popular Rest API test tool which has a high download number as over 32 million. Developers can create APIs in .Net programming language and test them easily. It supports serialization and deserialization of JSON, XML formats, synchronous and asynchronous HTTP calls, attaching objects to body of the requests in JSON or XML, uploading and downloading files. However, it can only work with .Net applications [39].

5.10. Karate DSL

It combines API test, performance and UI automations into a single unit. Multithreaded parallel executions can be run with the tool. Behaviour driven development based on Cucumber can be written in plain text form. It is also good for testing GraphQL structures. It has a built-in reporting mechanism and switching configuration between different environments [40, 41].

5.11. VRest NG

VRest NG is a tool providing automation testing and automated recording with scriptless assertions and response validators which makes the validation process easier. Difference of actual and expected test results can be seen with the help of reporting system. Test data can be written in .csv files hence multiple test scenario can be executed in a short time. It integrates test process with Excel sheet easily. It allows scenario-based flow while writing test cases [42].

5.12. ReadyAPI

ReadyAPI is designed as a platform which have three modules allowing teams to create and execute functional, security and performance tests. It was also professional version of SoapUI. It helps creating complex assertions without writing any code. ReadyAPI has a detailed dashboard displaying results and analysing them. It can generate reports in the form of HTML, comma seperated values (CSV) or JUnit. Tool can be integrated with other continuous integration tools as Jenkins, Git, Docker, Azure DevOps and TeamCity. Parallel execution of tests is another feature of the tool [43].

5.13. PyRestTest

PyRestTest is based on Python language and tests are described in YAML Ain't Markup Language (YAML) or JSON files. It returns exit codes in case of failure of a test. Code knowledge is not required to benefit from the tool. However, it can only work on Mac and Linux operation systems. PyRestTest could be applied for smoke tests which are carried out to control the main functions of a product after deployment to any environment (test, production, etc.) [35, 44].

5.14. Testerum

Testerum allows users to test web applications and Rest APIs. It has a friendly user interface that everybody can write and run their automated test scripts without any technical knowledge about programming. Therefore, it has an easy learning curve. Additionally, it helps to create test documentation explaining some features of test cases such as acceptance criteria, functional requirements etc. Documentation summarizes the steps of the testing process. Test scripts are formed with behaviour driven development and test results could be tracked with strong analytics tool [45].

5.15. SoapSonar

SoapSonar is built as a framework covering functional, performance and security testing. For this reason, it enables an end-to-end testing of a web service with its various testing types. Not only it can accomplish test process quickly but also users can deploy SoapSonar easily and the tool requires no script knowledge. Besides, it has for message and protocol layer identity standards allowing for cloud testing [46, 47, 48].

5.16. Parasoft Soatest

Parasoft Soatest is a strong test tool providing database testing, user interfaces of mobile and web applications, functional, security, load of APIs. It supports a great number of data format (XML, Web Services Description Language (WSDL), JSON, etc.), sources (CSV, Excel, Structured Query Language (SQL), etc.) and databases (Oracle, SQL Server, DB2, MySQL, etc.). Tool gives the opportunity of creating codeless test stories and has comprehensive reporting options [49].

6. CONCLUSION

Testing of web services are quitely important since they provide critical operations in information systems. To stress this importance firstly software testing processes are introduced generally and then key points of web service testing are mentioned in this study. In addition to them, test automation of a web service is explained in detail with Rest Assured tool. Usage of automation tools facilitates testing of web services with their special capabilities. These tools are compared in terms of different metrics and finally it is recognized that Katalon Studio, JMeter, RedwoodHQ, ReadyAPI, Parasoft/SoaTest are more powerful by having more capabilities. On the other hand, SoapUI, JMeter, KarateDSL, ReadyAPI and SoapSonar include more testing types. Of course, selection of test tool will vary according to the project, company and technical knowledge of the tester. The main contribution of this study is that it explains automation of a RESTful service comprehensively and can be helpful for selecting the best web service tool for special testing scenarios and environments according to the criterias mentioned in the fifth section of the study.

REFERENCES

1. Saini, G., K. Rai. An Analysis on Objectives, Importance and Types of Software Testing. – International Journal of Computer Science and Mobile Computing, Vol. 2, 2013, No. 9, pp. 18-23.

2. Luo, L. Software Testing Techniques. – Technology Maturation and Research Strategy Class Report for 17-939A, Institute for Software Research International Carnegie Mellon University, Pittsburgh, 2001.

3.Statista. Number of internet users worldwide from 2005 to 3019. https://www.statista.com/statistics/273018/number-of-internet-users-worldwide, 2019.

4. Quan, D. M. Implementing test automation with Selenium WebDriver. – Bachelor's Thesis, Lathi University of Applied Sciences, Degree Programme in Business Information Technology, 72 p., 2018.

5. Mahajan, P. Different Types of Testing in Software Testing. – International Research Journal of Engineering and Technology (IRJET), Vol. 3, 2016, No. 4, pp. 1661-1664.

6. Hussain, S., Z. Wang, I. K. Toure, A. Diop. Web Service Testing Tools: A Comparative Study. – International Journal of Computer Science Issues, Vol. 10, 2013, no. 1-3, pp. 641-647.

7. Emektar, M., Y. Altınsoy, U. Erdem. Sigortacılık Web Servislerinde Test ve Test Otomasyonu Yaklaşımı. – In: Proc. Of 12th National Software Engineering Symposium, İstanbul, Turkey, 2018, pp.1-8.

8. Arcuri, A. Automated Black and White-Box Testing of RESTful APIs with EvoMaster – IEEE Software, Vol. 38, 2021, no. 3, pp. 72-78.

9. Kumar, R., A. J. Singh. A Comparative Study and Analysis of Web Service Testing Tools – International Journal of Computer Science and Mobile Computing, Vol. 4, 2015, no. 1, pp. 433-442.

10. Devi, J., K. Bhatia, R. Sharma. A Study on Functioning of Selenium Automation Testing Structure. – International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 7, 2017, No. 5, pp. 855-862.

11. Tuteja, M., G. Dubey. A Research Study on importance of Testing and Quality Assurance in Software Development Life Cycle (SDLC) Models. – International Journal of Soft Computing and Engineering (IJSCE), Vol. 2, 2012, no. 3, pp. 251-257.

12. Chaurasia, N., K. Bahl. Analytical Study on Manual vs. Automation Testing Using Selenium. International Journal of Innovative Science, Engineering & Technology, Vol. 5, 2018, no. 10, pp. 40-43.

13. Ghuman, S.S. Software Testing Techniques. – International Journal of Computer Science and Mobile Computing, Vol. 3, 2014, no. 10, pp. 988-993.

14.Chintakyala, P. Beginners Guide to Software Testing. https://www.softwaretestingclass.com/wp-content/uploads/2016/06/Beginner-Guide-To-Software-Testing.pdf, 2013.

15. Babbar, H. Software Testing: Techniques and Test Cases. – International Journal of Resarch in Computer Applications and Robotics, Vol. 5, 2013, no. 3, pp. 44-53.

16. Hooda, I., R. S. Chhillar. Software Test Process, Testing Types and Techniques. – International Journal of Computer Applications, Vol. 111, 2015, no. 13, pp. 10-14.

17. Felderer, M., M. Büchler, M. Johns, A. D. Brucker, R. Breu, A. Pretschne. Security Testing: A Survey – Advances in Computers, Vol. 101, Elsevier, 2016, pp. 1-43.

18. Büyükyumukoğlu, G., E. Ersoy, M. Ş. Özdemir, S. Bağrıyanık, A. Karahoca. Challenges and Lessons Learned in Test Automation: Experiences from Telecommunications Industry. – In: Proc. of 11th Turkish National Software Engineering Symposium, Alanya, Turkey, 2017, pp. 78-88.

19. Madan, S., A. Kakkar, Web Uygulamaları için Çerçeve Olarak Test Otomasyonu. – Indian Journal of Computer Science and Engineering (IJCSE), Vol. 8, 2017, no. 3, pp. 479-486.

20. Kuday, G., Yazılım Mühendisliği Yöntemleriyle Yazılım Test Süreci. – Master Thesis, Haliç Üniversitesi Fen Bilimleri Enstitüsü, 119 p., 2015.

21. Lariya, S., S. Shrivastava, S. Nemai. Automation Testing using Selenium Web Driver & Behavior Driven Development (BDD) – International Journal for Scientific Research & Development (IJSRD), Vol. 6, 2018, no. 3, pp. 2014-2017.

22. Emektar, M., Y. Altınsoy, U. Erdem. Sigortacılık Web Servislerinde Test ve Test Otomasyonu Yaklaşımı. – In: Proc. Of 12th National Software Engineering Symposium, İstanbul, Turkey, 2018, pp.1-8.

23. Yener, H. A., F. Baştürk, B. Meçik. General Mobile Productivity Increase with Software Development and Test Automation: General Mobile. – Karamanoglu Mehmetbey University Journal of Engineering and Natural Sciences, Vol. 1, 2019. no. 1, pp. 172-196.

24.Cigniti. Web Services Test Automation: Framework, challenges & benefits. https://www.cigniti.com/blog/web-services-test-automation-framework-challenges-and-benefits/, 2021.

25. Sharma, L. What is Rest?. https://www.toolsqa.com/rest-assured/what-is-rest/, 2017.

26. Soni, A., V. Ranga. API Features Individualizing of Web Services: REST and SOAP. – International Journal of Innovative Technology and Exploring Engineering (IJITEE), Vol. 8, 2019, no. 9, pp. 664-671.

27. Halili, F., E. Ramadani. Web Services: A Comparison of Soap and Rest Services. – Modern Applied Science, Vol. 12, 2018, no. 3, pp. 175-183.

28. TestNG. https://testng.org/doc/.

29. Software Testing Help. Top 15+ Most Popular Web Service Testing Tools In 2021. https://www.softwaretestinghelp.com/web-services-testing-tools/, 2021.

30. SoapUI. https://www.soapui.org/.

31. Postman. https://www.postman.com/.

32. Rest Assured. https://rest-assured.io/.

33. Katalon. https://www.katalon.com/.

34. Apache Jmeter. https://jmeter.apache.org/.

35. Srivastava, V., Top 25+ API Testing Tools. https://nordicapis.com/top-25-api-testing-tools/, 2020.

36. RedwoodHQ. https://redwoodhq.com/.

37. Unirest-Java. http://kong.github.io/unirest-java/.

38. HTTPMaster. https://www.httpmaster.net/.

39. RestSharp. https://restsharp.dev/.

40. Karate DSL. https://github.com/karatelabs/karate.

41.Testbytes. 21 Best API Testing Tools That are insanely good. https://www.testbytes.net/blog/api-testing-tools/, 2020.

42. vRest NG. https://vrest.io/.

43. ReadyAPI. https://smartbear.com/product/ready-api/overview/.

44. PyRestTest. https://github.com/svanoort/pyresttest.

45. Testerum. https://testerum.com/.

46. SoapSonar. https://www.crosschecknet.com/products/soapsonar/.

47.Crosscheck Networks SOAPSonar Reviews. https://www.itcentralstation.com/products/crosscheck-networks-soapsonar-reviews.

48.The most powerful Web Service testing tool SOAPSonar download tutorial. https://blog.katastros.com/a?ID=01100-94bd3f32-6b11-441e-a679-bb4516b674ad

49. Parasoft SoaTest. https://www.parasoft.com/products/parasoft-soatest/

50. Shahin, M., M. A. Babar, S., L. Zhu, Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices. – IEEE Access, 2017, pp. 1-32.

APPENDIX - 1

Table 1. Features of RESTful Web Services Testing Tools

	Code Knowledge Requirement	Built –in Parallel Execution of Test Cases	Cost Free Version	User Interface	Support Multiple Language (Scripting)	Support Multiple Operating Systems	Support Multiple Platform Tesing	Reporting	Performance Testing	Functional Testing	Security Testing	Continuos Integration
SoapUI Open Source		✓	✓	✓	✓	✓			✓	✓	✓	
Postman			✓	✓	✓	✓		✓		✓	✓	✓
RestAssured	✓		✓			✓				✓		✓
KatalonStudio		✓	✓	✓	✓	✓	✓	✓	✓	✓		✓
JMeter			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
RedwoodHQ	✓	✓	✓	✓	✓	✓	✓	✓		✓		✓
UniRest	✓		✓		✓	✓			✓	✓		
HttpMaster		✓	✓	✓			✓	✓	✓	✓		
RestSharp	✓		✓			✓				✓		✓
KarateDSL		✓	✓			✓	✓	✓	✓	✓		✓
vRestNG		✓	✓	✓		✓	✓	✓		✓		✓
ReadyAPI		✓		✓		✓		✓	✓	✓		✓
PyRestTest		✓	✓			✓		✓		✓		✓
Testerum		✓	✓	✓		✓	✓	✓		✓		✓
SoapSonar		✓	✓	✓			✓	✓	✓	✓	✓	
Parasoft/SoaTest		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓