Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 10, October-2020

Development of RESTful WebAPI using Token Based OAuth 2.0 Authorization

Pawan Kumar Bhat
Computer Science and Engineering ASRA College of Engineering and Technology
Punjab, India

Rajnish Kansal
Computer Science and Engineering ASRA College of Engineering and Technology
Punjab, India

***Abstract*-Web services are the enhanced form of the web application based programming interfaces (API) or web APIs that are accessed through Hypertext Transfer Protocol (HTTP) to execute on a far off system hosting the requested services to provide services on different devices like laptop, mobile, and others. It is a framework that helps to create and develop HTTP based RESTFUL web services that do not restrict the client- server communication. The open authorization (OAuth) 2.0 industry-standard protocol for authorization allows a user to grant a third-party web site or application access to the user's protected resources, without necessarily revealing their long- term credentials or even their identity. The goal of this research paper was to make a prototype of an Application Programming Interface within the C# language and implement a REST-style of architecture, capable of managing and authenticating different sorts of information, like devices and users, also allow the users to insert, read, update, delete relevant data by open authorization (OAuth) 2.0 protocol to secure ASP.NET Web API using token- based authentication.**

Keywords— REST, WebAPI, ASP.Net, MVC, ORM, JSON, OAuth 2,0 , HTTP protocols, and URI.

I. INTRODUCTION

We live in a world fully linked with technology. Almost everybody is a member of social media like Facebook, Twitter, or LinkedIn and has a public profile on the internet. In 2009,with the boom of smartphones. A large majority of companies and experts in technological marketing agree on the fact that in a window of five years, billions of devices will be connected in the Internet of Things (IoT). The Internet of Things is a concept that refers to the digital interconnection of any everyday object with the internet. These objects acquire more value due to their ability to send and receive data with the user, the manufacturer, and other connected devices. With the aim of being able to connect, the devices require a software component that makes possible its communication and management. The ideal software for this function is an interface that can be implemented in any device, i.e. it is multiplatform, and that allows fast, simple and efficient communication. [1]

Web services are an enhanced form of web application-based programming interfaces (API) or web APIs that are accessed through Hypertext Transfer Protocol (HTTP) to execute on a far off system hosting the requested service to provide services on different devices like laptop, mobile, and others. It is a framework that helps to create and develop HTTP based a RESTFUL service that doesn't restrict the client-server communication. The open authorization (OAuth) 2.0 industry- standard protocol for authorization allows a user to grant a third-party web site or application access to the user's protected resources, without necessarily revealing their long- term credentials or even their identity. The authorization server includes authorization information with the access token and signs the access token. An access token is often reused until it expires [2]. The goal of this thesis was to make a prototype of an Application Programming Interface within the C# language and implement a REST-style of architecture, capable of managing and authenticating different sorts of information, like devices and users, also on allow the users to insert, read, update, delete relevant data by open authorization (OAuth) 2.0 protocol to secure ASP.NET Web API using token-based authentication.

The motivation for this paper is to study, develop, and authorize a Restful WEB API. The goal of this paper is to develop and analyze a WEB API using the REST architectural style. This API will be basic for the future development of different multi-platform applications for the Internet of Things. The WEB API should be able to manage and authenticate different kinds of information like devices and users, as well as to allow the devices to import, export, storage and post-process relevant data.

II. LITERATURE REVIEW

The literature survey includes several topics as background for the proposed Development of Restful Web API using token based Authorization.

K. V. Kanmani et. al. proposed web services that are based on applications programming interfaces (API) or web APIs accessed through Hypertext Transfer Protocol (HTTP) to execute on a remote system hosting the requested services. A RESTful web service is a budding technology, and a lightweight approach that does not restrict the client-server communication. The open authorization (OAuth) 2.0 protocol enables the users to grant third-party application access to their web resources without sharing their login credential data. The Authorization Server includes authorization information with the Access Token and signs the Access Token. An access token can be reused until it expires. An authentication filter is used for business services. This paper presents a secure communication at the message level with minimum overhead and provides a fine grained authenticity using the Jersey framework. This paper presents a literature survey on these various techniques and how each of these techniques has its own benefits and limitations. This paper discusses how REST protocol is performed using open authorization. Compared to the SOAP-based integration approach, REST has many

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 10, October-2020

advantages such as service is addressable and can be connected to, the interface is consistent, and resources can be cached. Moreover, Restful Web services are the Web service that has a simple description of the document and is easy to release, and provides a platform for the future work on web services. [2]

Femke De Backere et. al. exhibited the design of a RESTful Web Service Communication through Mobile Clients. The aspect of security remains a big issue for RESTful Web services. Many of the current security mechanisms violate the RESTful principles and are, because of their stateful nature, not able to cope with the scalability advantages that REST provides. Basic RESTful security standards are outdated and omit vital security solutions. TLS seems to be a usable standard, nonetheless, the overhead introduced is too large for non-continuous connections, as with mobile interaction. Therefore, a custom security mechanism is proposed, using only a bare minimum of non-RESTful elements. Comparing this implementation with a fully TLS-based solution shows that this method outperforms the latter, both on the aspect of messaging as processing overhead. Because of the genericness of REST, our proposed security mechanism can be adopted by a wide variety of other RESTful Web services. [3]

B. Jaya Kaviya et. Al. proposed a survey on RESTful Web Services Composition. Web services composition has been implemented for combining the various web services as used in many fields. This provides various suggestions that will be more helpful for the users. This can reduce the execution time and the current status of the service which has been utilized by the users can be known easily at a faster rate. The developed web services are available as web data in the specified web repository. The algorithm must be designed efficiently in order to reduce execution and response time. [4]

Marios Argyriou1 et. al. exhibited security flows in the OAuth
2.0 Framework. In this paper, they have presented an overview of the OAuth 2.0 framework and investigated the relevant literature. They described the known vulnerabilities and how they exploit and jeopardize the authentication, authorization, and session integrity properties. Since the OAuth 2.0 framework is widely adopted, this paper confirms and supports the urgent adoption of the proposed mitigation techniques to tackle each known issue. [5]

III. ENTITY FRAMEWORK ORM

The programming language chosen for the development of the API is C#. C# pronounced as C-Sharp is a high-level programming language of .Net Framework. C# is a simple, modern, general-purpose, object-oriented programming language developed by Microsoft within its .NET initiative led by Anders Hejlsberg [6]. In addition to C#, the Web API is built with the Entity Framework. Entity Framework is an open-source ORM framework for .NET applications supported by Microsoft. It enables developers to work with data using objects of domain-specific classes without focusing on the underlying database tables and columns where this data is stored. With the Entity Framework, developers can work at a higher level of abstraction when they deal with data and can create and maintain data-oriented applications with less code compared with traditional applications. Entity Framework is an object-relational mapper (O/RM) that enables .NET developers to work with a database using .NET objects. [7] The figure1 illustrates where the Entity Framework fits into the application. In order to use Entity Framework, the only requirement is to have a minimum required Visual Studio 3.5 or higher installed.

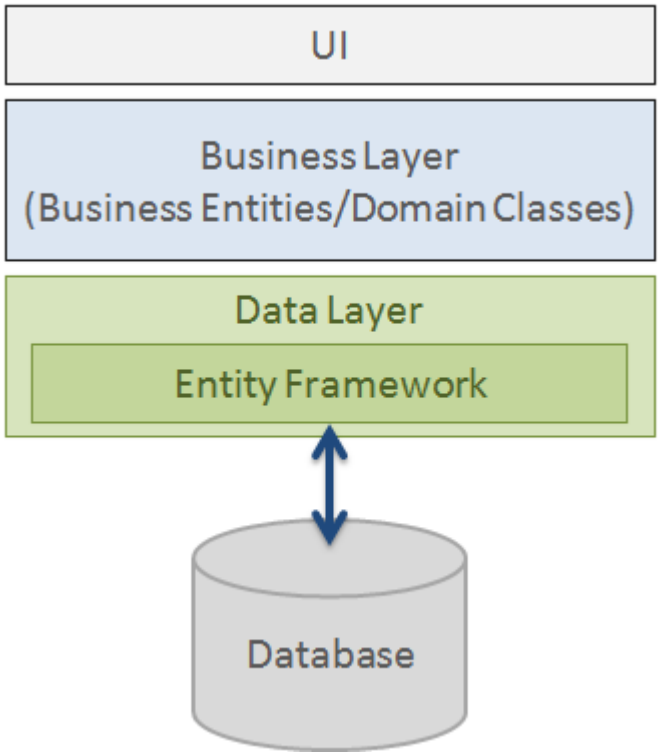

Figure 1. Entity Framework ORM [7]

Another tool that is used in this project is Microsoft SQL SERVER. Microsoft SQL Server is a relational database management system developed by Microsoft. As a database server, it is a software product with the primary function of storing and retrieving data as requested by other software applications which may run either on the same computer or on another computer across a network. Microsoft markets at least a dozen different editions of Microsoft SQL Server, aimed at different audiences and for workloads ranging from small single-machine applications to large Internet-facing applications with many concurrent users. Microsoft SQL Server is built on top of SQL, a standardized programming language that database administrators (DBAs) and other IT professionals use to manage databases and query the data they contain. SQL Server is tied to Transact-SQL (T-SQL), an implementation of SQL from Microsoft that adds a set of proprietary programming extensions to the standard language [8].

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 10, October-2020

IV. RESULT & CONCLUSION

The database of the project has been built in the Microsoft SQL Server and database design and structure are explained here with the help of screenshots. Here we have designed two tables namely Login and Product in the database. For the authorization part, we used the Login table with username and password as fields in it. For the details part, we have used the Product table to store, retrieve, edit, and delete data in it. For the token generation, we have also designed the stored procedure to generate the token id used for the authorization purpose by the Web API. The database table is shown in figure 2.

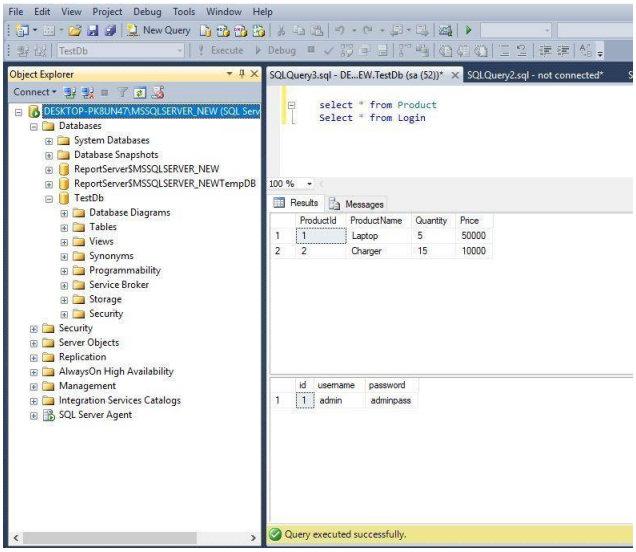

Figure 2. Database table

In the Web API, we created the following action methods in the controller class.

- GetAllProducts()
- GetProduct()
- InsertProduct()
- UpdateProduct()
- DeleteProduct()

In the GetAllProduct() action method, an authorized user can view all the products stored, gets them populated from the database using the data access layer. To test the Web API we have used an extension tool available in the Mozilla Firefox browser called RESTED, as an add on to test the Web API. It is easy to use so lets us work as effectively as possible. It features all the most commonly used HTTP methods, setting headers, saving requests to local storage, and more [9].

Some of the neat features of this extension are:

- URL templating: If we have a parameter that is shared across several rest calls, we can template the string. Example - rest-service.com/user/{{userId}}. We can then set userId in a separate panel, and it will be resolved across all your different requests.
- Collections: Save requests to local storage for easy reuse later.
- Basic auth and custom headers: Easily set basic auth and custom headers that are passed alongside the request.
- Imports: Import data from HAR (HTTP Archive), or from a Postman collection
- Modern design: Based on modern technology and minimalist web design, this extension is anything but dated. It also features changeable themes.[10]

The Firefox rested tool extension is shown in figure 3.

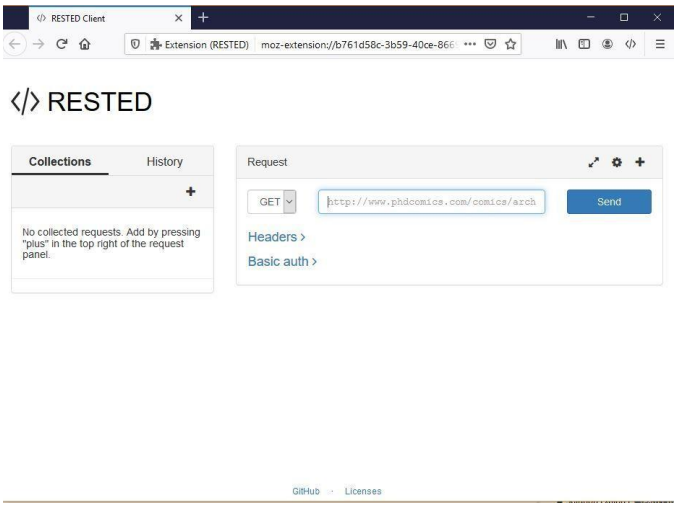

Figure 3. Firefox Rested Tool Extension

We pass the URL, of the Web API to the Rested tool without passing any credentials and select the request method like GET, after clicking the Send Request button available in the tool; we get 401 unauthorized access message as shown in figure 4.

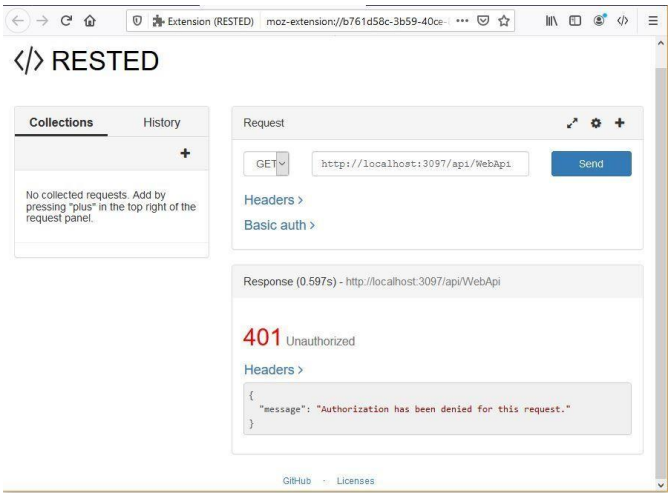

Figure 4. 401 Unauthorized message

In order to get access to the Web API we will provide the system user authorization to get access token and

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 10, October-2020

then use that access token as a header in the REST Web API. The authorized message is shown in figure 5.

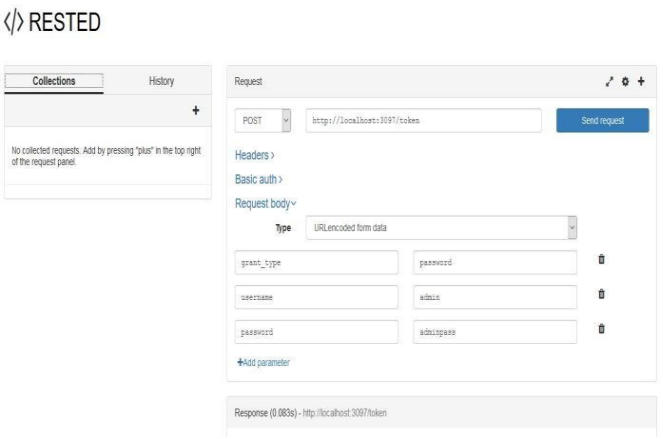

Figure 5. Authorized message

After user authorization is successful, then only user can view the details. In the GetProduct() method, an authorized user can filter the product records based on the product available to him/her. The product details are shown in figure 6.

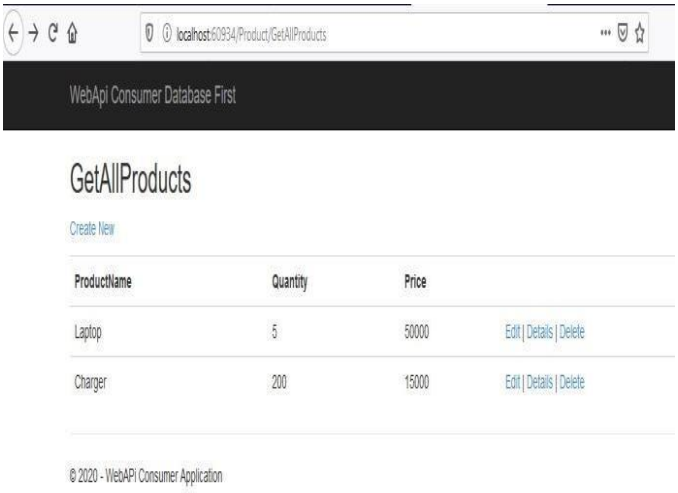

Figure 6. Product details

In the InsertProduct () method, an authorized user can Insert a new product into the database. Create and update methods are shown in figures 7 and 8 respectively.

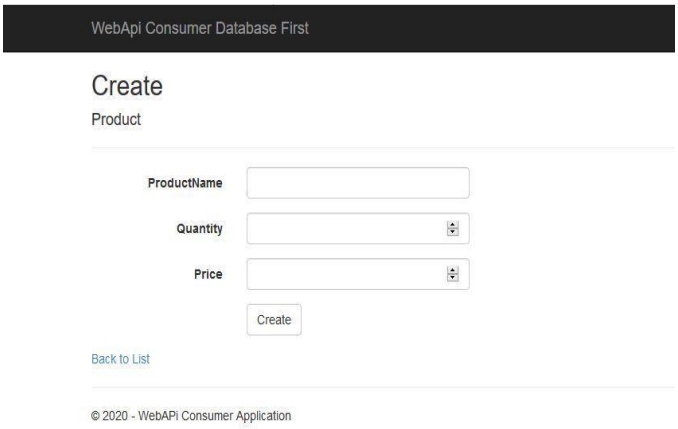

Figure 7. Create method

In the UpdateProduct () method, an authorized user can update an existing product in the database.

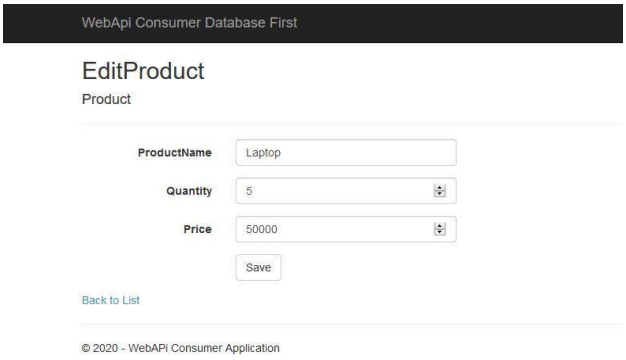

Figure 8. Update method

In the DeleteProduct () method, an authorized user can delete an existing product in the database.

ACKNOWLEDGEMENT

I extend my thanks to my coordinator & Head, CSE for his patient guidance and useful critiques of this work. My grateful thanks are also extended to the Management, for their continuous encouragement. Finally, I wish to thank my parents and wife for their continuous support and encouragement throughout my study.

REFERENCES

[1] https://www.abiresearch.com/press/more-than-30-billion-devices-will- wirelessly-conne
[2] K.V.Kanmani and P.S.Smitha."Survey on Restful web services using open authorization (Oauth)" IOSR Journal of Computer Engineering, vol. 15, pp. 53-56, 2013.
[3] Femke De Backere, Brecht Hanssens, Ruben Heynssens, Rein Houthooft, Alexander Zuliani "Design of a Security Mechanism for RESTful Web Service Communication through Mobile Clients" IEEE Network Operations and Management Symposium (NOMS), 2014.
[4] B. Jaya Kaviya, G. Selvakumar "A Survey on RESTful Web Services Composition", International Journal of Advanced Research in Computer Science and Software Engineering, pp98-100, 2015.
[5] Marios Argyriou1, Nicola Dragoni1,2, and Angelo Spognardi, "Security Flows in OAuth 2.0 Framework: A Case Study, Springer International Publishing, pp. 396–406,2017.
[6] https://www.tutorialspoint.com/csharp/csharp_discussion.htm
[7] https://www.entityframeworktutorial.net/what-is-entityframework.aspx
[8] https://en.wikipedia.org/wiki/Microsoft_SQL_Server
[9] https://github.com/RESTEDClient/RESTED
[10] https://chrome.google.com/webstore/detail/rested/eelcnbccaccipfolokglf hhmapdchbf