RESTFUL WEB SERVICE IMPLEMENTATION ON UNKLAB INFORMATION SYSTEM USING JSON WEB TOKEN (JWT)

MARAMIS, JEREMIAH
NIM: 105021710023

JURUSAN ILMU KOMPUTER
FAKULTAS TEKNIK INFORMATIKA
UNIVERSITAS KLABAT
MANADO
2020

PREFACE

Praise God Almighty for His grace and guidance that the writer can complete the research with the title:

"RESTful Web Service Implementation on UNKLAB INFORMATION SYSTEM Using JSON Web Token (JWT)"

The researcher also wants to expressed gratitude and respect to:

1. Mr. Jimmy Moedjahedy, S.Kom, MM, as the main advisor and Stenly Adam, MSc, as the co-advisor that helps guide and giving advice to the researcher regarding this research.
2. Mr. Andrew Tanny Liem, Ph.D, as Dean of the Faculty of Computer Science Universitas Klabat.
3. Mr. Green Ferry Mandias, M. Cs, as the Head of Informatics Program Study as well as the instructor of *Research Project II* course.
4. Mrs Lidya Laoh, S.Kom, MM, as the head of panelist and Mr. Andria Wahyudi, S.Kom, M.Eng, as the member of the panelist
5. All the lecturers of the Faculty of Computer Science that give me knowledge throughout all semesters.
6. Mother and Father as well for Brother Dearil that love and supports me throughout my entire life.
7. To my friend Giovanni Tjandra for helping me create the styling of the researcher presentation as well as the poster for this research.

8. To my friends Indri, Habell, Michi and Harold that provide support throughout the semester.

This research is far from perfect. Therefore, any criticisms and suggestions for improving this research are most welcomes and may this research be useful for all the reader.

ABSTRAK

Data mahasiswa yang berstatus publik dan tersimpan di Universitas Klabat (Unklab) saat ini disimpan dalam database lokal dan hanya dapat diakses oleh administrator database. Layanan web RESTful (RESTful web service) bertindak sebagai media yang menyediakan akses ke data dalam database yang terhubung dengan web service tersebut bagi mereka yang memiliki izin, mengakibatkan peningkatan dalam aksesibilitas data. Setiap pengguna yang memenuhi syarat oleh web service dapat menggunakan data Sistem Informasi Unklab untuk mengembangkan aplikasi seluler atau web lain tanpa membuat database serupa lainnya, sehingga menghilangkan proses pembuatan database yang duplikat. JSON Web Token (JWT) juga digunakan dalam web service ini untuk memastikan bahwa orang yang mengakses web service berwenang untuk mengakses data tertentu. Metode agile digunakan untuk mengembangkan aplikasi ini. Kesimpulannya, web service yang dirancang dapat menghasilkan operasi CRUD (Create, Update, Update Delete) terhadap database siswa dan memberikan akses kepada pengguna yang berwenang dengan JWT yang valid.

Kata Kunci : *Data mahasiswa, RESTful Web Service, Unklab Information System, JSON Web Token*

ABSTRACT

Student data that are public and stored on Universitas Klabat (Unklab) currently be stored in a local database and can only be accessed by the database administrator. A RESTful web service acts as a medium that provides access to the data for those who have permission, improving the data's accessibility. Any user eligible by the web service can use the Unklab Information System data to develop a mobile or another web application without making another similar database, thus eliminating the redundant process of making a duplicate database. A JSON Web Token (JWT) is also used in this web service to ensure that the person accessing the web service is authorized to access the specific data. The agile method is used to develop this application. In conclusion, the web service can create, read, update, delete (CRUD) operation regarding the student database and provide it to the user that is authorized with a valid JWT.

Keywords: *Student data, RESTful Web Service, Unklab Information System, JSON Web Token*

TABLE OF CONTENT

PREFACE.....i
ABSTRAK.....iii
ABSTRACT.....iv
TABLE OF CONTENT.....v
LIST OF FIGURES.....vii
LIST OF TABLES.....viii
CHAPTER I INTRODUCTION.....1
1.1 Background Study.....1
1.2 Problem Definition.....3
1.3 Research Purpose.....3
1.4 Benefit of Research.....3
1.5 Scope and Limitation.....4
1.5.1 Scope of Research.....4
1.5.2 Limitation of Research.....4
2.1 Web Service.....5
2.2 Restful Web Service (REST API).....6
2.3 JSON (JavaSript Object Notation).....7
2.4 JWT (JSON Web Token).....8
2.5 Previous Research.....9
2.7 Conceptual Framework.....12
3.1 Research Design.....14
3.2 Research Instrument.....14
3.2.1 Data Type.....14
3.2.2 Data Collection Technique.....15
3.3. Application Development Environment.....15
3.3.1. Hardware.....15
3.3.2. Software.....16
4.1 Use Case Diagram and Scenario.....17
4.2 Sequence Diagram.....23
4.3 ERD (Entity Relationship Diagram).....26

CHAPTER V TESTING.....27
5.1 Logging using an admin account.....27
5.2 Posting or Creating a Student Data.....28
5.3 Getting The Student Data.....28
5.4 Updating Student Data.....29
5.5 Deleting Student Data From The Database.....29
5.6 Logging Out From The Web Server.....30
5.7 Request using a blacklist token or unauthorized token.....30
CHAPTER VI CONCLUSION AND RECOMMENDATION.....32
REFERENCES.....33

LIST OF FIGURES

Figure 2.1: Web Service Architecture.....5
Figure 2.2 Example of JWT Decoded Form.....9
Figure 2.3 Example of JWT Encoded Form.....9
Figure 2.4 Agile Mehodology.....11
Figure 2.5 Conceptual Framework of RESTful Web Service.....12
Figure 4.1 Use Case Diagram.....18
Figure 4.2 How The Web Server Generates a Token.....23
Figure 4.3 Web Server Handling a Request.....24
Figure 4.4 Web Server process client logout.....25
Figure 4.5 ERD Main Database.....26
Figure 5.1 Input Request in The Login Process.....27
Figure 5.2 Example output from the logging request.....28
Figure 5.3 POST Request to /signup and A Response From The Web Server Telling The User Is Successfully Registered.....28
Figure 5.4 Output Student Data With a Raw JSON Form.....29
Figure 5.5 Updating Student “faculty_code” to Another Value.....29
Figure 5.6 Request and A Response From The Web Server Telling The User Has Been Removed.....30
Figure 5.7 A Logout Request to The Web Server As Well Response From The Web Server to The User That The User is Logged Out.....30
Figure 5.8 A Web Server Response To An Unauthorized Token.....31
Figure 5.9 An Output That Tells The User Of The Invalid Token.....31

LIST OF TABLES

Table 4.1 Use Case Scenario User Login.....19
Table 4.2 Use Case Scenario Get Student Data.....19
Table 4.3 Use Case Scenario Post Student Data.....20
Table 4.4 Use Case Scenario Update Student Data.....21
Table 4.5 Use Case Scenario Delete Student Data.....22
Table 4.6 Use Case Scenario Logout.....22

CHAPTER I

INTRODUCTION

1.1 Background Study

In the year academic of 2019 to 2020, 3.298 students are recorded as the student of Universitas Klabat (Unklab). Each student contains data representing their name, faculty, date of birth, blood type and other details data. A database administrator can only access this data with a specific query to retrieve and use it. This process can be time-consuming because many requests to a particular data can only be managed by one person that uses a specific query according to each request. Another issue that occurs is accessibility. A user who wants to create an app with the Unklab Information System database needs to create another database that can represent the database in Unklab Information System, which creates a redundant process. There needs to be a medium where any user can access appropriate data from the Unklab Information System database with ease. A web service can be used as a tool to solve this problem.

A web service is a medium that describes specific values, functionality that is delivered through the internet protocol in which can be used as a mechanism for another service or application to use [1]. With this functionality, web service can provide a faster process for the end-user with another machine to fetch data and use its data without making another source of data. Web service also has characteristic supporting machines and other machines to communicate in

a network [2]. This means that one type of OS (Operating System) can communicate with other OS types with the same network, which is the internet.

The implementation of web service can be seen in many research. For example, Manuaba [3] implement a web service of the lecturer's assessment information system on Politeknik Negeri Bali. Kurniawan [4] uses a web service for the sales tracking process and sales order process, and Arif [5] implements a web service on filling application of student study plan card.

One issue that comes from this situation is web service handling the authorization of the data access. The web service needs to implement an access control so the database administrator can only do a particular action such as posting data. An access token, JSON Web Token (JWT), is implied to every user with the Unklab Information System database's registration number to handle the authorization issue.

There is four HTTP method that will be used in this RESTful web service which are :

1. GET: Retrieve specific data
2. POST: Add or create new data
3. PUT: Update existing data
4. DELETE: Delete specific data

In this research, the Representational State Transfer (REST) is chosen as the web service architecture. In this case, where the data are on

1.2 Problem Definition

Below are the problems that the researcher needs to solve on this paper:

1. How to develop a web service for Unklab Information System with REST architecture that acts as an endpoint for the end-user to fetch data or manipulate data.
2. How to develop a web service that authorize a user that is requesting to the web service.

1.3 Research Purpose

The purpose of this research is:

1. To develop a web service that can be used as an endpoint for the end-user to access data from Unklab Information System database.
2. To develop a web service that fully functional in terms of giving the right response to the end-user regarding the process of fetching data.
3. To develop a web service that authorized end-user using JWT.

1.4 Benefit of Research

1. **Software Developer:** The web service can be a tool for an application to access data from the Unklab Information System without making an additional database as a seperate data source.
2. **Database Administrator**: Can be used as a way to increase efficiency to organize the Unklab Information System database without engaging with sophisticated queries.

3. **Other Researcher**: As a template for other researchers to build a web service based on a REST architecture on a different information system.

1.5 Scope and Limitation

Divided into two parts. scope and limitation

1.5.1 Scope of Research

1. This web service will be connected to a database that represents the Unklab Information System database.
2. This web service will be host on the internet and have a public ip address that can be accessed by the end-user.

1.5.2 Limitation of Research

1. This web service does not have a front-end implementation.
2. This web service will only be used by student, teacher and staff that are registered on the Unklab Information System.
3. This web service will be used a dummy database that only stored a certain data according to the database administrator requirement.

CHAPTER II
LITERATURE REVIEW

2.1 Web Service

According to W3C, which is the institute that works in standardized web term, web service is a system designed to help machines communicate through a network that is built using HTTP (Hypertext Transfer Protocol) [7].

According to IBM, web service architecture has three fundamental operations that called *publish*, *find*, and *bind*. These operations can be illustrated in the following figure.

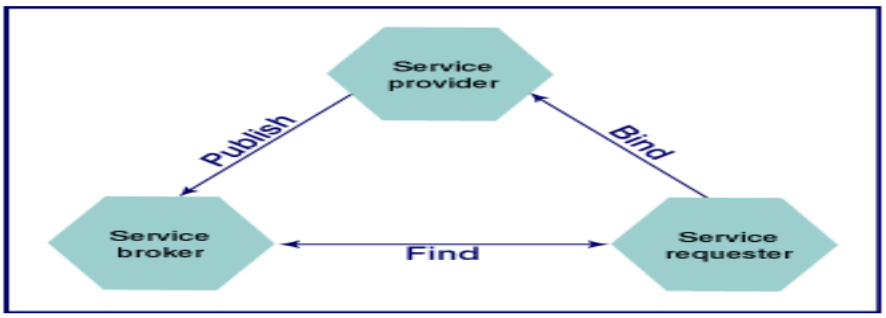

Figure 2.1: Web Service Architecture [8]

The service provider *publish* or serve the web service to the service broker, which is the application that acts as an intermediate. The service requester, which is the client app, then f*ind* the required services using the service broker and *bind* it to the service provider [8].

A web service provides communication between applications that are connected to one network [9]; hence all the Operating System that is connected to the web service can communicate between software.

2.2 Restful Web Service (REST API)

REST stands for Representation State Transfer is an architectural style build for distributed hypermedia systems that are presented by Roy Fielding in the year 2000 from his dissertation [10]. According to Roy, seven constraints imply on a web service:

1. **Starting with NULL style:** A starting point of a REST architecture that is an empty set of constraints.
2. **Client-Server**: There is a separation of concern between the client and the server, thereby simplifying the server components resulting in the improvement of user interface and scalability.
3. **Stateless**: The communication that the client and server have must be stateless, meaning that for each request from the client to the server, the server can only contain all the information from each request without taking advantage of any stored context or prior requests on the server. This constraint emphasizes more properties such as visibility, reliability, and scalability.
4. **Cache**: For improving network efficiency, a cache constraint was added. It means that the data within a response to a request need to be labelled implicitly or explicitly as cacheable or non-cacheable. Improve scalability on the server-side and performance on the client-side.

5. **Uniform Interface:** Information that REST provides must be in a standardized form, meaning it decoupled from the service provider.
6. **Layered System:** Apply a hierarchical layered to the REST architecture so each component cannot view other components outside the layer. It encapsulates legacy component from other components.
7. **Code-On-Demand:** A implemented functionality so REST can be extended by downloading and executing code in runtime. Reducing the number of features that need to be pre-implemented.

Postman to test the API. API (Application Programming Interface) works as an intermediate between two software programs so they can communicate [9]. REST API works in a way that each *resource* can be accessed using a URI (Uniform Resource Identifier) following the REST style [11].

Restful API or Restful web service uses HTTP request methods such as POST, G|ET, PUT, DELETE to implement CRUD (create, read, update, delete) operations for the intended resources using the associated URI [12].

2.3 JSON (JavaSript Object Notation)

JSON is a format data that is lightweight, and it is easy for a human to understand the data as well as for machine to process [13]. JSON can be used on a REST API as the response format.

JSON has two structures. First is the collection of name/value pairs, which in many programming languages known as *an object,* and the second is

an *ordered list of values* which in many programming languages can be achieved with an *array*.

2.4 JWT (JSON Web Token)

JWT encodes the data that is a JSON object by representing it as a string and secure it by digitally signed using algorithms such as HMAC (Hash-based Message Authentication Code) [14].

JSON Web Token contains three components that are separated by dots (.), namely [15]:

1. **Header:** Mainly contain two parts. The type of the token which is JWT and the signing algorithm that is being used. Example HMAC SHA256 or RSA.
2. **Payload:** The second part of the token that contains claims. These claims are a statement about an entity (usually the user) and additional data.
3. **Signature**: Contain the part of the encoded header and the encoded payload and a secret key that the server provides and hash it with the HMAC SHA256 algorithm.

JSON Web Token is used to ensure that the data that goes from the client to the server has not tampered by other users or the client itself.

Figure 2.2 shows an example of JWT in a decoded form that contains the three components or parts mentioned above.

```
//Header = Contain the type of the token and signing algorithm
{
"alg": "HS256",
"typ": "JWT"
}
//Payload = Contain the actual data
{
"sub": "1234567890",
"name": "John Doe",
"admin": true
}
//Signature = Hash function
HMACSHA256(
base64UrlEncode(header) + "." +
base64UrlEncode(payload),
"secret" //this secret can be any string
)
```

Figure 2.2 Example of JWT Decoded Form

Figure 2.3 shows an example of JWT formed from the hash algorithm or the encoded form of JWT.

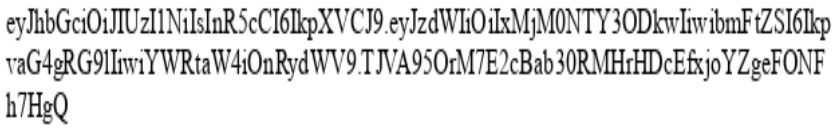

Figure 2.3 Example of JWT Encoded Form

The hashed encoded string from Figure 2.2 will be used by the web server to tell if the data that is in JWT is tampered or not and will be an authorized tool to check if the user is authorized to request the data.

2.5 Previous Research

Muhammad Haekal and Eliyani research [16] about "Token-Based Authentication Using JSON Web Token on SIKASIR RESTful Web Service" conclude that a JSON web token can be implemented in SIKASIR web service. .

A similar research by Muhammad Perkasa and Eko Setiawan propose A REST API web service built for monitoring civilian data with the usage of an access token [17] concludes that the web service improve the time of handling the process of registering civilian data.

A conducted research by Albertengo et al. [18] about “On the performance of web services, google cloud messaging and firebase cloud messaging” explain how REST outperform other web service such as google cloud messaging and firebase cloud messaging in terms of data efficiency, delay and power consumption.

Research by Aldy, Alam, and Muhammad stated that JWT is a form of stateless authentication, meaning that the JWT is not stored on the server but rather in the client [19].

2.6 Theoritical Framework

The researcher chooses an agile method for the theoretical framework that is going to apply in this research. Agile methodology is chosen because it can provide immediate feedback from the stakeholder regarding every iteration on the application implementation. As stated in the Agile Manifesto [20], the agile methodology focuses more on the program's result and people collaboration inside the project than comprehensive documentation of the program itself. The steps that required in this methodology are presented as follows:

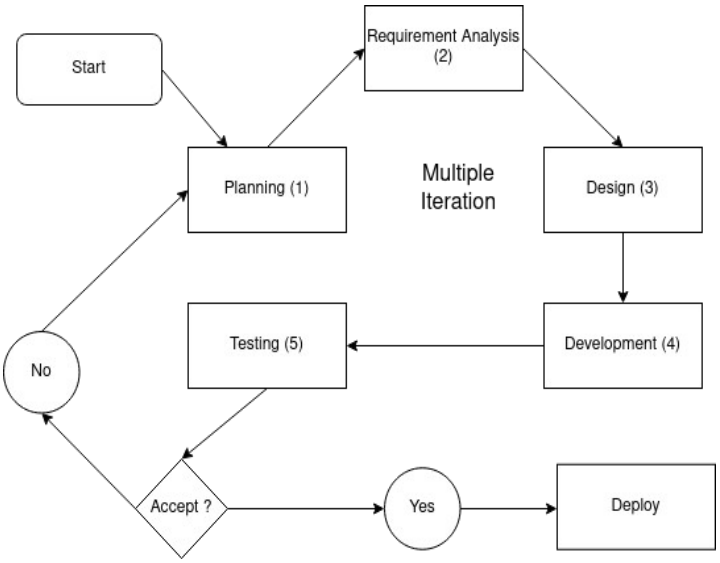

Figure 2.4 Agile Mehodology

Below is the explanation regarding the steps that occur in Figure 2.3:

1. Planning: Organizing what should be accomplished within a set of time by using a specific framework. This step required the researcher to gather requirements from the stakeholder, mainly the database administrator, to provide the proposed software's requirement.
2. Requirement Analysis: This step required the researcher to list the priority of the requirement that has been gathered from the previous step. Organize which requirement is a priority and which is a secondary feature to the proposed software.
3. Design: This step required the researcher to build the architectural design based on the already analyzed requirement.
4. Development: This step required the researcher to code the actual software using the design made from the previous step.

5. Testing: Test the existing software that has been made by the researcher. If the stakeholder accepts the current software, then deploy the software alongside Unklab Information System. If not, then the software development cycle repeats for the system until the stakeholder accepts the software. Every iteration that the software goes has incremental progress in terms of features.

2.7 Conceptual Framework

Below is the provided details regarding the conceptual framework of the Restful web service that will be implemented in the UNKLAB information system.

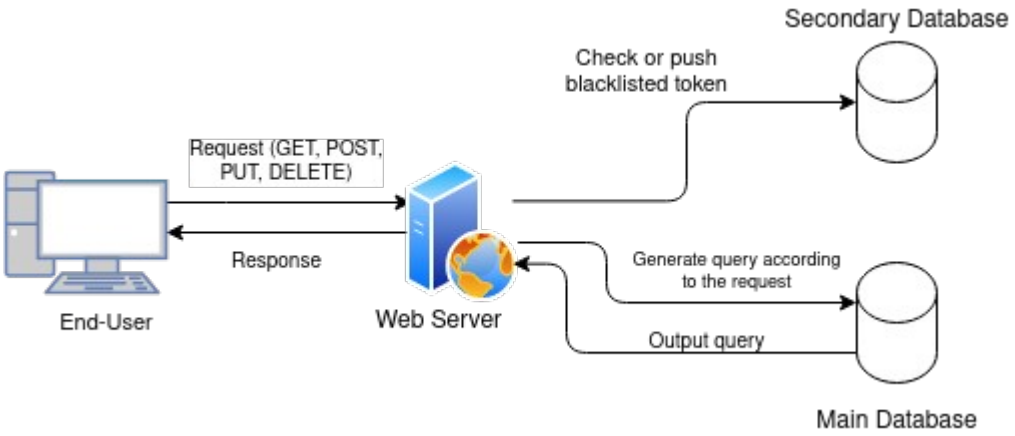

Figure 2.5 Conceptual Framework of RESTful Web Service

Below is the explanation regarding the steps that occur on Figure 2.4**:**

1. **End-User**: The client that wants to access the web server and give a request HTTP method such as GET, POST, PUT, DELETE to the server. It can be the Database Administrator or other person that registered to the Unklab Information System Database. The user

passed a JWT if the URI that the user access need a JWT authorization.

2. **Web Server**: Media that the business logic process occurs. It checks if the end-user is authorized to access the intended HTTP method for specific URI. The web server generate a JWT from a client login process, check if the JWT that is passed are in the secondary database or blacklisted, generate query according to the client request and give response to the user whether with JSON format or string.
3. **Main Database:** A MySQL database that store data regarding Unklab Information System.
4. **Secondary Database**: Store the blacklisted token that are pushed from the web server when the client log out from the web server. Redis is used as the secondary database because it is faster than the SQL database and the way Redis stored data by key-value [21] is suitable to store the blacklist token because it only has one attribute.

CHAPTER III
RESEARCH METHODOLOGY

3.1 Research Design

As mentioned in chapter 2 in this research, a software engineering method will be used, which is the Agile Model. The implementation phase in order is Planning, Requirement Analysis, Design, Development and Testing.

3.2 Research Instrument

This research will be using two types of a research instrument which are data type and data collection technique.

3.2.1 Data Type

The researcher uses two types of data which are primary data and secondary data. Below is the description regarding each data:

1. Primary Data

Primary Data is obtained directly through an interview with the database administrator of UNKLAB regarding the data that are going to be used from the database of UNKLAB Information System.

2. Secondary Data

Secondary data is obtained through other references such as research papers, books, journals and web articles to determine what and how the web service should be implemented regarding this research.

3.2.2 Data Collection Technique

The researcher collects the data by discussing what data should be used on this research with UNKLAB database administrator. The data that is chosen by the database administrator will be used on the implementation of this research.

The researcher also uses another reference to find what web service should be implemented on the Unklab Information System and how to implement the web service.

3.3. Application Development Environment

To design and make the application, the researcher uses hardware and software as the primary tools for developing the application. Below is the list of hardware and software specification that the researcher use:

3.3.1. Hardware

Below are the list of hardware that support this research:

1. Laptop ACER E1-472G, and below are the following specifications:
2. CPU: Intel Core i4-4200u.
3. GPU: NVIDIA GeForce 820M.
4. STORAGE: 500GB HDD.
5. RAM: 2 X 8GB DDR3.

3.3.2. Software

Below are the list of software that support this research:

1. Linux Manjaro as the chosen Operating System for code development.
2. MySQL Workbench to help integrate the API with the main database .
3. JavaScript as the chosen programming language.
4. Postman to test the API.

CHAPTER IV
SYSTEM DESIGN

In this chapter the researcher use UML (Unified Modelling Language) to visualize the flow of the application in this research. There are two types of UML that are used in this research namely *use case diagram* with *scenario* and *sequence diagram.*An ERD (Entity Relationsip Diagram) is also used to visualize the database design implementation.

4.1 Use Case Diagram and Scenario

Use case diagram is used to show interactions between end-user and the web service.

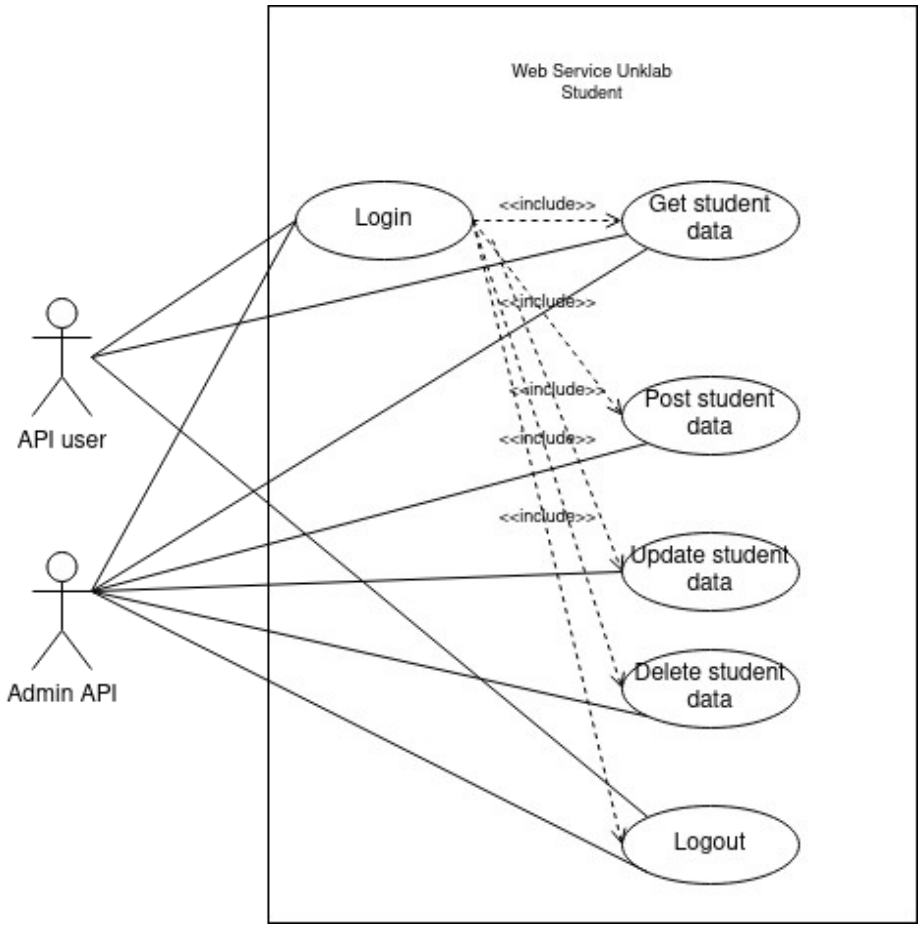

Figure 4.1 Use Case Diagram

Figure 4.1 shows what actions can be done to the web server by the actors. Every use case on Figure 4.1 will be explain by a *use case scenario*.

Table 4.1 Use Case Scenario User Login

Title	Login.
Description	User log in to the web server.
Actors	API user (student and lecturers of Unklab), Admin API.
Precondition	Actors need to have an email and password to log in to the web server.
Basic Flow	1. User log in to the web server by inserting email and password to the web server. 2. The web server checks if the email and password are registered in the main database of Unklab Information System.
Post Condition	If the user is registered on the main database of Unklab Information System then the web server will output a token to the user. If its not registered then no token is provided to the user.

Table 4.2 Use Case Scenario Get Student Data

Title	Get student data.
Description	User retrieve student data from the web server
Actors	API user (student and lecturers of Unklab), Admin API.
Precondition	User needs to log in to the web server and retrieve the token from the log in process.
Basic Flow	1. Request with a GET http method to */student* url. 2. Input the token that is retrieve from the login process alongside this request.
Post Condition	If the token is valid then the web server output the student data. If its not valid (tampered or unauthorized), then the web server will tell the user that the token is invalid.

Table 4.3 Use Case Scenario Post Student Data

Title	Post student data.
Description	User create new student data.
Actors	Admin API.
Precondition	User needs to log in to the web server and retrieve the token from the log in process.
Basic Flow	1. Request with a POST http method to */signup* url. 2. User input the token that is retrieve from the login process alongside this request. 3. User input the student data to the web server
Post Condition	If the token is valid then the web server will registered the student data to the main database. If the student email or registration number is already in the database then the web server will tell the user that the database already have the student data. If the token is invalid, an output from the web server to the user telling the token is invalid is produced.

Table 4.4 Use Case Scenario Update Student Data

Title	Update student data.
Description	User update or change student data from the web server
Actors	Admin API.
Precondition	User needs to log in to the web server and retrieve the token from the log in process.
Basic Flow	1. Request with a PUT http method to */student* url. 2. Input the token that is retrieve from the login process alongside this request. 3. User input the student record that want to be updated on the request body parameter. 4. User input “initial_stud_reg_num” a value in the request body parameter as an index for the web server to update the student record based on the value of “initial_stud_reg_num”
Post Condition	If the token is valid and the request is valid, then the web server updates the student data. If there is no student with the registration number of “initial_stud_reg_num” from the request, then the web server tells the user that there is no user in the main database. If the token is invalid, an output from the web server to the user telling the token is invalid is produced.

Table 4.5 Use Case Scenario Delete Student Data

Title	Delete student data.
Description	User delete student data from the web server
Actors	Admin API.
Precondition	User needs to log in to the web server and retrieve the token from the log in process.
Basic Flow	1. Request a DELETE http method to */student* url. 2. User input the token that is retrieve from the login process alongside this request. 3. User Input “student_reg_number” on the request body parameter.
Post Condition	If the token is valid and the request is valid, then the web server delete the student that match the value of “student_reg_number” in request body parameter from the main database. If there are no registration number that match the “student_reg_number” value in request body parameter, then the web server tells the user that the student is not existed. If the token is invalid, then the web server tells the user that the token is invalid or unauthorized to be used.

Table 4.6 Use Case Scenario Logout

Title	Logout.
Description	User log out from the web server.
Actors	API user (student and lecturers of Unklab) ,Admin API.
Precondition	User needs to log in to the web server and retrieve the token from the log in process.
Basic Flow	1. Request a POST http method to */logout* url. 2. User input the token that is retrieve from the login process alongside this request.
Post Condition	The web server insert the token to the secondary database to be blacklisted.

4.2 Sequence Diagram

The Sequence diagram is used to explain the process of how the application in this research works.

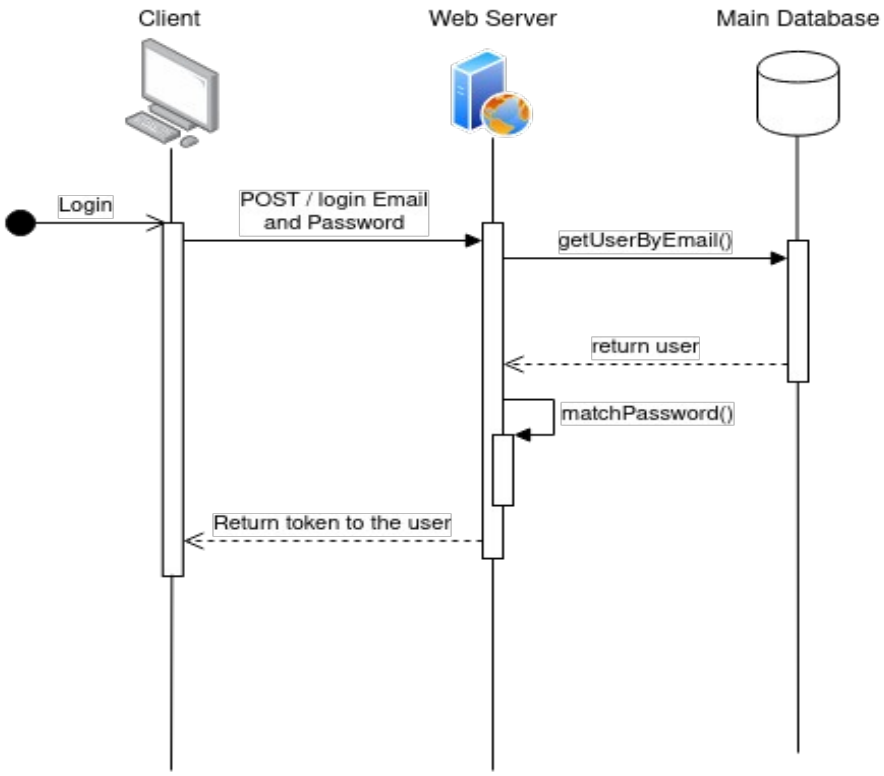

Figure 4.2 How The Web Server Generates a Token

Figure 4.2 explains how the client can get a token from a web server. First, the client logs to the web server using a mobile device or PC by entering the email and the password. Then the web server processes the email, and it finds the corresponding user according to the email. The database returns a user according to the email. Then it checks the password if the password match with the corresponding password that has been given by the database. If it matches, then

the web server returns a token to the client that logs in. If the user's email and password are not matched with the database, then no token is provided.

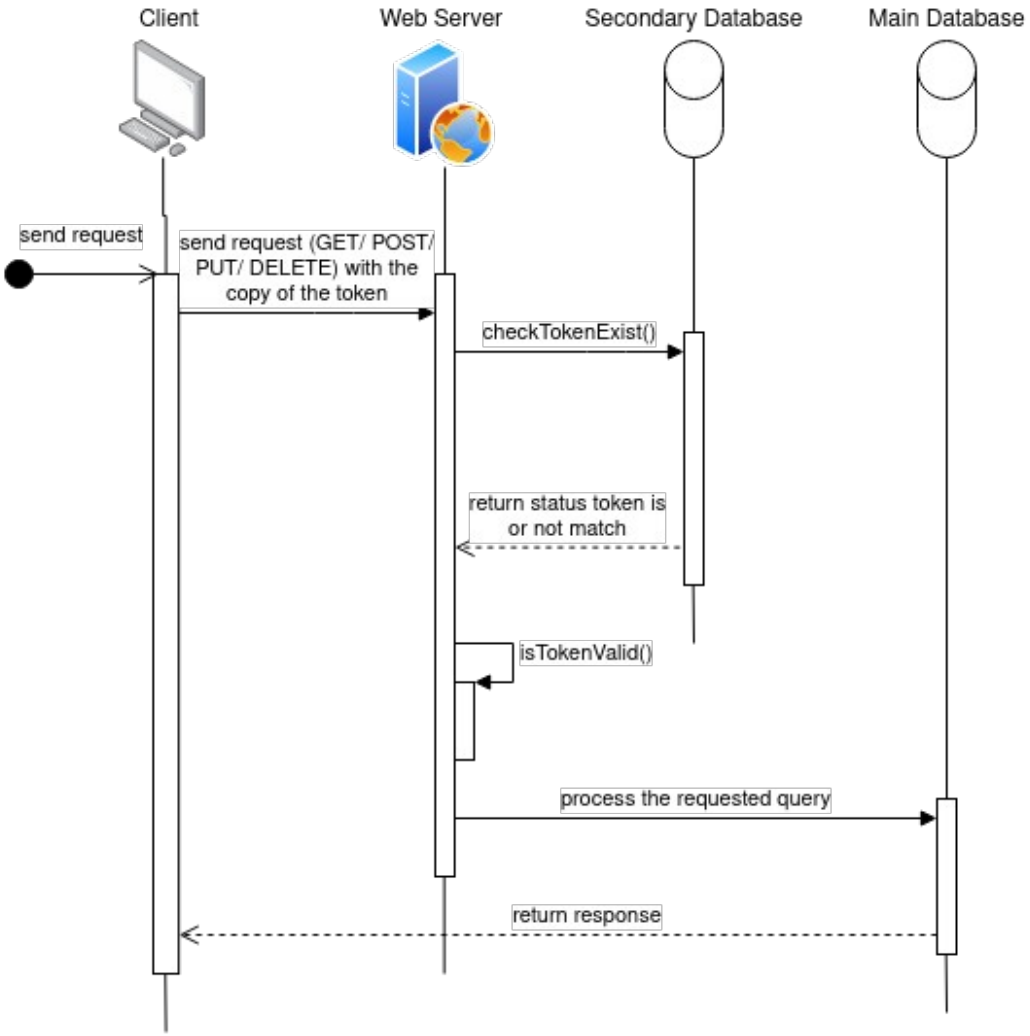

Figure 4.3 Web Server Handling a Request

Figure 4.3 explains how the server is handling the request from a particular client. First, the client requests the web server such as GET, POST, PUT or DELETE with a token obtained from the logging process. Then the token is checked by the web server if it matches the token that is in the secondary database, which is the database that stores all the blacklisted token. After that, the

web server will validate the user token if it does not tampered or does not match any of the tokens in the secondary database. If the token is tampered or match any of the tokens in the secondary database, then the token becomes invalid. The web server processes the requested query from the user and returns a response according to the request.

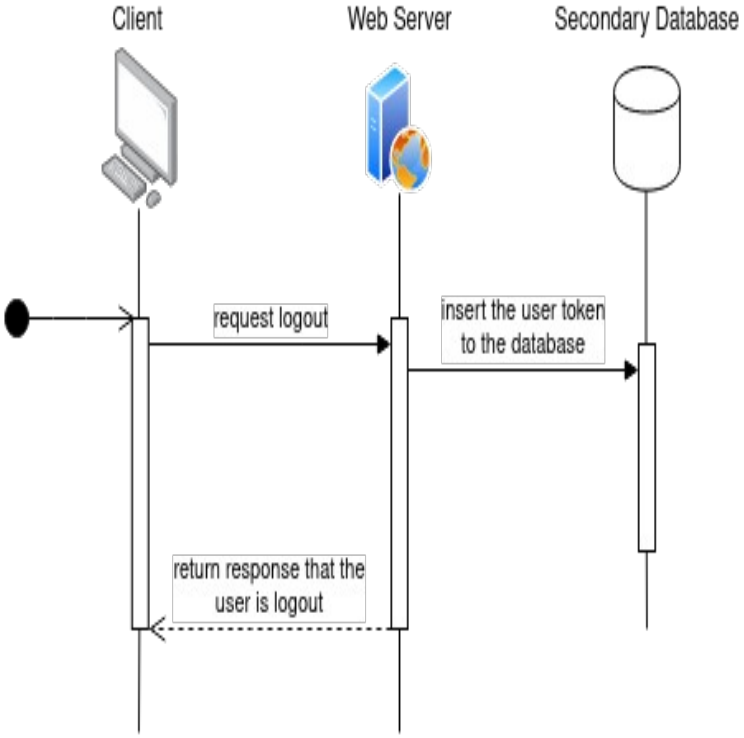

Figure 4.4 Web Server process client logout

Figure. 4.4 explains how the server handles a logout request from a client or user. When the user wants to logout from the server, the server takes the user token, and the server inserts the token to the secondary database. The secondary database will be used to blacklist all the token from the user logout from the server to prevent anyone from using the same token to access the server.

4.3 ERD (Entity Relationship Diagram)

Below is the ERD that is used to show the tables implementation regarding the main database that is going to be used in this research.

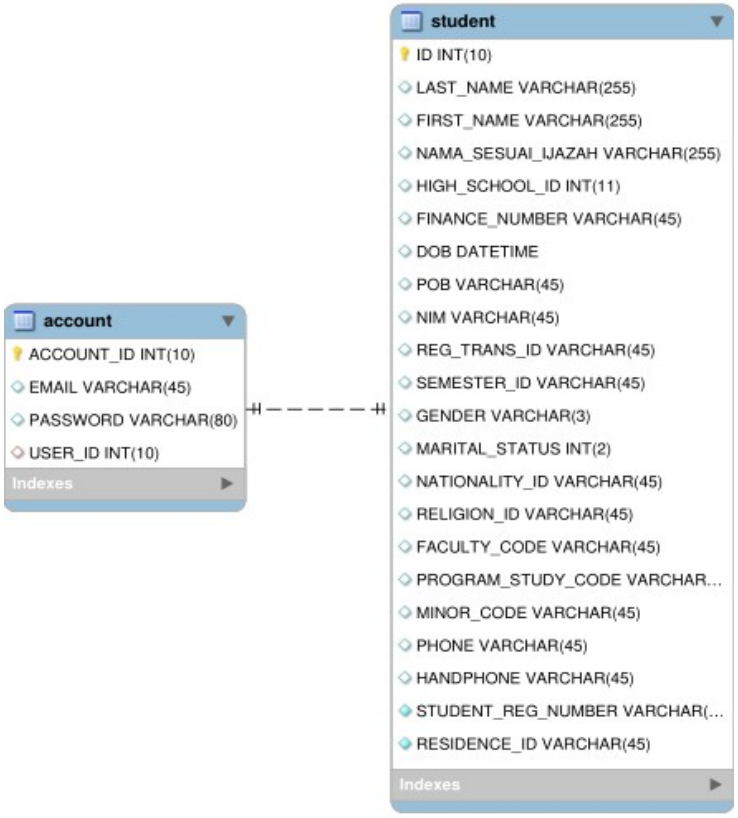

Figure 4.5 ERD Main Database

CHAPTER V

TESTING

Testing the web server will be done using an application called POSTMAN. This phase will be consisting of testing various cases on actors toward the web service. The web service will be using a “dummy” database representing the actual database the web service will be using.

5.1 Logging using an admin account

Figure 5.1 shows a user that is admin (depends on the email and password used) logs in to the web server.

Figure 5.1 Input Request in The Login Process

Figure 5.2 is an example output of a token produced from the request shown in Figure 5.1.

```
{
  "message": "Token successfully created",
  "accessToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
     eyJpZCI6ImFkbWluQHVua2xhYi5hYy5pZCIsImlhdCI6MTYwMDg3MTM1MywiZXhwIjoxNjAwODcyMjUzfQ.
     ByIDP8vJnk8kYENPA8C5Am_ZzQzcjJX6ubtmpXEcngE"
}
```


Figure 5.2 Example output from the logging request

5.2 Posting or Creating a Student Data

Figure 5.3 shows a POST request to the web service to create student data and the output, which is the web server's response with a valid token.

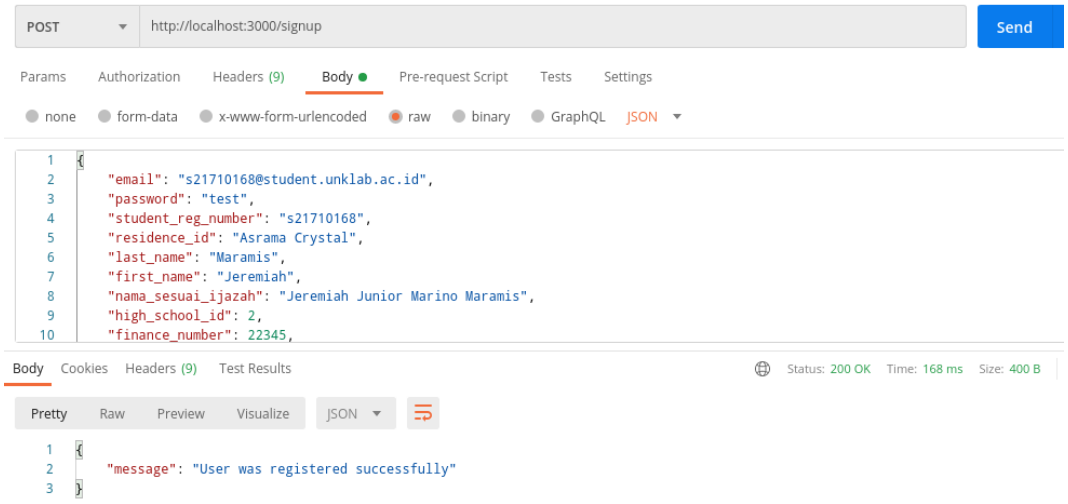

Figure 5.3 POST Request to /signup and A Response From The Web Server Telling The User Is Successfully Registered

5.3 Getting The Student Data

Figure 5.4 is the output data registered from Figure 5.3 when the token provided is authorized to access the url.

[{"ID":1,"LAST_NAME":"Maramis","FIRST_NAME":"Jeremiah","NAMA_SESUAI_IJAZAH":"Jeremiah Junior Marino Maramis","HIGH_SCHOOL_ID":2,"FINANCE_NUMBER":"22345","DOB":"2000-03-29T00:00:00.000Z","POB":"Manado","NIM":"105021710023","REG_TRANS_ID":"","SEMESTER_ID":"2017","GENDER":"L","MARITAL_STATUS":1,"NATIONALITY_ID":null,"RELIGION_ID":"Kristen","FACULTY_CODE":"FIK","PROGRAM_STUDY_CODE":"Informathics","MINOR_CODE":"","PHONE":"","HANDPHONE":"1234567","STUDENT_REG_NUMBER":"s21710168","RESIDENCE_ID":"outsider"}]

Figure 5.4 Output Student Data With a Raw JSON Form

5.4 Updating Student Data

Figure 5.5 is an example of updating the student attribute, which is "faculty_code" to another value using the "initial_stud_reg" request as a guide for the database to know which data should be updated.

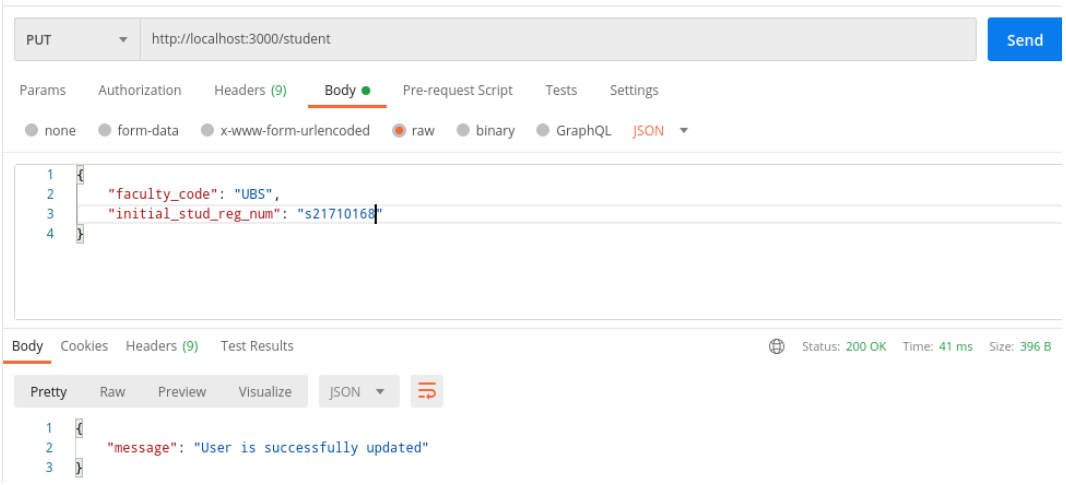

Figure 5.5 Updating Student "faculty_code" to Another Value

5.5 Deleting Student Data From The Database.

Figure. 5.6 shows a user from the database with the student_reg_number of "s21710168" will be deleted.

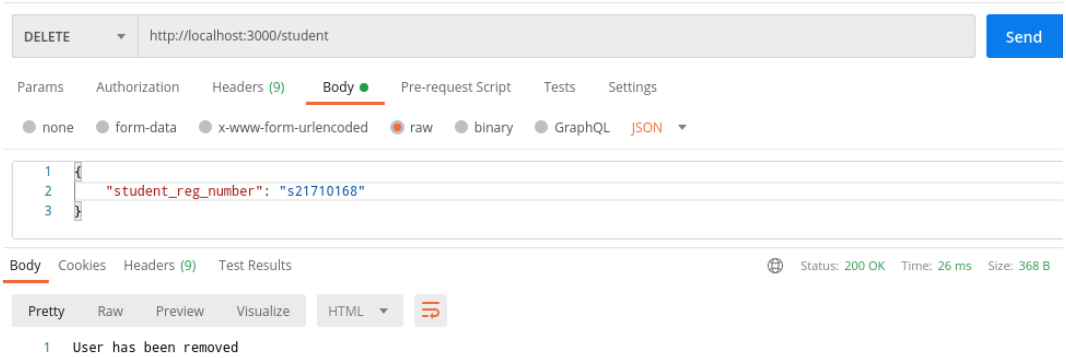

Figure 5.6 Request and A Response From The Web Server Telling The User Has Been Removed

5.6 Logging Out From The Web Server

Figure 5.7 shows a user logout from the web service and the web service response to the user.

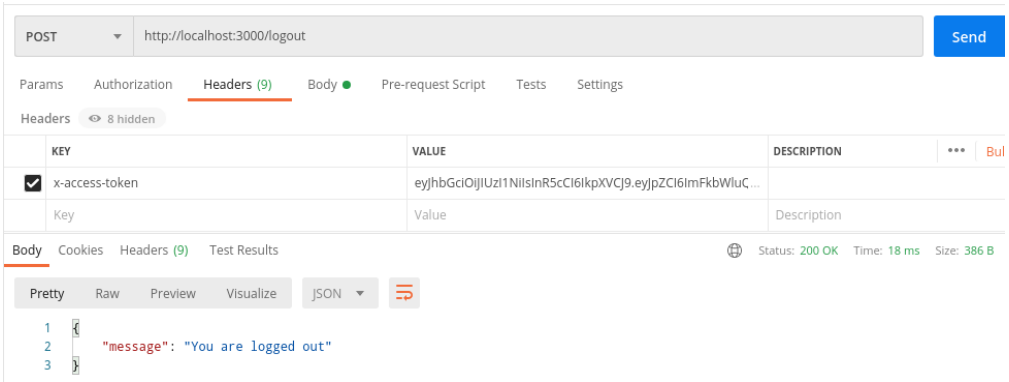

Figure 5.7 A Logout Request to The Web Server As Well Response From The Web Server to The User That The User is Logged Out

5.7 Request using a blacklist token or unauthorized token

Figure 5.8 is an example when an unauthorized token is used alongside a client request to the web service that needs a token authorization process.

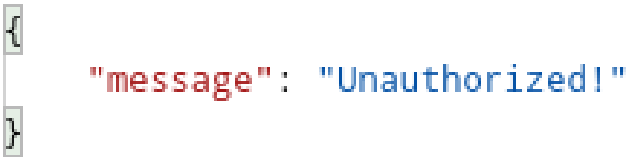

Figure 5.8 A Web Server Response To An Unauthorized Token

Figure 5.9 shows the output from the web server to the user using a blacklisted token alongside a request.

Figure 5.9 An Output That Tells The User Of The Invalid Token

CHAPTER VI

CONCLUSION AND RECOMMENDATION

The web service built on this research provides an endpoint for any user registered in the Unklab information system to access the student data provided in the web service. The JWT that is implemented on the web service handles the user's authorization to access the web service. CRUD operation is also implemented in the web service for the user authorized to use the operation.

A front–end interface is recommended to be implemented onto this web service to improve the ease of use of this web service to the user.

REFERENCES

[1] A. Bosworth, “Developing Web services,” in *Proceedings 17th International Conference on Data Engineering*, Heidelberg, Germany, 2001, pp. 477–481, doi: 10.1109/ICDE.2001.914861.

[2] I. A. Elia, N. Laranjeiro, and M. Vieira, “Understanding Interoperability Issues of Web Service Frameworks,” in *2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, Atlanta, GA, USA, Jun. 2014, pp. 323–330, doi: 10.1109/DSN.2014.40.

[3] I. B. P. Manuaba and E. Rudiastini, “API REST Web service and backend system Of Lecturer’s Assessment Information System on Politeknik Negeri Bali,” *J. Phys. Conf. Ser.*, vol. 953, p. 012069, Jan. 2018, doi: 10.1088/1742-6596/953/1/012069.

[4] E. Kurniawan, “IMPLEMENTASI REST WEB SERVICE UNTUK SALES ORDER DAN SALES TRACKING BERBASIS MOBILE,” vol. 07, no. 01, p. 12.

[5] A. Laksito, “Web-Service Implementation on Filling Application of Student Study Plan Card,” 2006.

[6] F. Halili and E. Ramadani, “Web Services: A Comparison of Soap and Rest Services,” *Mod. Appl. Sci.*, vol. 12, no. 3, p. 175, Feb. 2018, doi: 10.5539/mas.v12n3p175.

[7] “Web Services Architecture.” https://www.w3.org/TR/2004/NOTE-ws-arch-20040211/#relwwwrest (accessed Apr. 16, 2020).

[8] “Web Services architecture overview,” Sep. 06, 2000. http://www.ibm.com/developerworks/library/w-ovr/index.html (accessed Mar. 17, 2020).

[9] “API Features Individualizing of Web Services: REST and SOAP,” *Int. J. Innov. Technol. Explor. Eng.*, vol. 8, no. 9S, pp. 664–671, Aug. 2019, doi: 10.35940/ijitee.I1107.0789S19.

[10] R. T. Fielding, “in Information and Computer Science,” p. 180, 2000.

[11] M. Melnichuk, Yu. Kornienko, and O. Boytsova, “WEB-SERVICE. RESTFUL ARCHITECTURE,” *Autom. Technol. Bus. Process.*, vol. 10, no. 1, Apr. 2018, doi: 10.15673/atbp.v10i1.876.

[12] R. Sinha, M. Khatkar, and S. C. Gupta, “Design & Development of a REST based Web Service Platform for Applications Integration on Cloud,” vol. 1, no. 7, p. 5.

[13] “JSON.” https://www.json.org/json-en.html (accessed Mar. 20, 2020).

[14] J. Bradley, N. Sakimura, and M. B. Jones, “JSON Web Token (JWT).” https://tools.ietf.org/html/rfc7519 (accessed Mar. 22, 2020).

[15] auth0.com, “JWT.IO - JSON Web Tokens Introduction.” http://jwt.io/ (accessed Mar. 22, 2020).

[16] M. Haekal and Eliyani, “Token-based authentication using JSON Web Token on SIKASIR RESTful Web Service,” in *2016 International*

Conference on Informatics and Computing (ICIC), Mataram, Indonesia, 2016, pp. 175–179, doi: 10.1109/IAC.2016.7905711.
[17] M. I. Perkasa and E. B. Setiawan, "Pembangunan Web Service Data Masyarakat Menggunakan REST API dengan Access Token," *J. ULTIMA Comput.*, vol. 10, no. 1, pp. 19–26, Jul. 2018, doi: 10.31937/sk.v10i1.838.
[18] G. Albertengo, F. G. Debele, W. Hassan, and D. Stramandino, "On the performance of web services, google cloud messaging and firebase cloud messaging," *Digit. Commun. Netw.*, vol. 6, no. 1, pp. 31–37, Feb. 2020, doi: 10.1016/j.dcan.2019.02.002.
[19] A. P. Aldya, A. Rahmatulloh, and M. N. Arifin, "Stateless Authentication with JSON Web Tokens using RSA-512 Algorithm," *J. INFOTEL*, vol. 11, no. 2, p. 36, Jun. 2019, doi: 10.20895/infotel.v11i2.427.
[20] "Manifesto for Agile Software Development." https://agilemanifesto.org/ (accessed May 14, 2020).
[21] R. Čerešňák and M. Kvet, "Comparison of query performance in relational a non-relation databases," *Transp. Res. Procedia*, vol. 40, pp. 170–177, 2019, doi: 10.1016/j.trpro.2019.07.027.