View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Pusat Jurnal UIN Ar-Raniry (Universitas Islam Negeri)

INTEGRATED EMAIL MANAGEMENT SYSTEM BASED GOOGLE APPLICATION PROGRAMMING INTERFACE USING OAUTH 2.0 AUTHORIZATION PROTOCOL

Zuhar Musliyana*, Andita Ghaitsa Satira, Mahendar Dwipayana***, Ayu Helinda******

**Department of Information Systems, Ubudiyah Indonesia University, Syiah Kuala, Banda Aceh, Indonesia, zuhar@uui.ac.id*
***Department of Information Systems, Ubudiyah Indonesia University, Syiah Kuala, Banda Aceh, Indonesia, andita.ghaitsa@gmail.com,*
****Department of Information Systems, Ubudiyah Indonesia University, Syiah Kuala, Banda Aceh, Indonesia, mahendar@uui.ac.id*
*****Department of Informatics Engineering, Ubudiyah Indonesia University, Syiah Kuala, Banda Aceh, Indonesia, ayu@uui.ac.id*

Email Correspondence: zuhar@uui.ac.id

Received: November 16, 2019 Accepted: May 13, 2020 Published: June 30, 2020

Abstract : Google Apps is a service provided by Google that allows users to use Google products with their own domain names. Among the products offered by Google Apps are email (Gmail), Docs (Google Drive), and Classroom services. In addition, Google Apps also provides Application Programming Interface (API) services that can be used by developers to take advantage of various features provided by Google. Universitas Ubudiyah Indonesia (UUI) is one of the universities that use Google Apps service for managing student emails. At present, UUI student email management through Google Apps is still not integrated with academic information system data. As a result, UUI must allocate special resources for managing student emails manually. Based on these problems, this study proposes an integration system for UUI student email management using the Google Apps API. This system is designed using PHP programming. The Google Apps API authentication method uses OAuth 2.0. The results of this study indicate that student email management on Google Apps can be done through campus academic information systems. With this system, students can activate email independently without having to be registered manually to the Google Apps page by the campus email managers.
Keywords : Google Apps API, Integration, Email Management, OAuth 2.0

Abstrak : Google Apps adalah sebuah layanan yang disediakan oleh Google yang memungkinkan pengguna dapat menggunakan produk google dengan nama domain sendiri. Di antaranya produk yang disediakan Google Apps yaitu layanan email (Gmail), dokumen (Google Drive), dan Classroom. Selain itu, Google Apps juga menyediakan layanan *Application Programming Interface* (API) yang dapat dimanfaatkan oleh pengembang untuk memanfaatkan berbagai layanan yang disediakan oleh Google. Universitas Ubudiyah Indonesia (UUI) merupakan salah satu universitas yang menggunakan layanan Google Apps untuk pengelolaan email mahasiswa. Saat ini pengelolaan email mahasiswa UUI melalui Google Apps masih belum terintegrasi dengan data sistem informasi akademik. Akibatnya UUI harus mengalokasikan sumber daya khusus untuk mengelola email mahasiswa secara manual. Berdasarkan permasalahan tersebut penelitian ini mengusulkan sistem integrasi

pengelolaan email mahasiswa UUI menggunakan API Google Apps. Sistem ini dirancang menggunakan pemograman PHP. Metode autentikasi API Google Apps menggunakan OAuth 2.0. Hasil penelitian ini menunjukkan pengelolaan email mahasiswa pada Google Apps dapat dilakukan melalui sistem informasi akademik kampus. Dengan adanya sistem ini mahasiswa dapat melakukan aktivasi email secara mandiri tanpa harus didaftarkan secara manual ke halaman Google Apps oleh pengelola email kampus.
Kata kunci : API Google Apps, Integrasi, Pengelolan Email, OAuth 2.0

Introduction

Google Apps is a service that allows users to use various Google products using custom domain names. Google Apps provided some services such as email, classroom, calendar, google drive and site services. In addition, Google Apps also provides Application Programming Interface (API) that can be used by application developers to access various services provided by Google (Treude & Aniche, 2018). API is a collection of commands/functions that can be utilized so that a system can interact with other systems (Lensakom, 2016).

Universitas Ubudiyah Indonesia (UUI) is one of the universities that use Google Apps for student email services. At present, the email management of UUI campus has not been integrated with the campus academic information system so the data of student email users must be manually registered into the Google Apps portal. This situation requires the campus to allocate special time and resources in order to manage student email data. With the increasing number of students, of course, time and resources required to do so will be even greater in the future.

Based on these problems, this study proposes a design for UUI student email management integration system using the Google Apps API. This system development uses PHP programming and MySQL database (database used in UUI academic information systems). PHP is the most popular web-based programming language in use today (W3Techs, 2019). PHP is included in the server-side scripting programming language that can be used for web-based application development (Munadi, Musliyana, Arif, Azmi, & Syahrial, 2016; Zuhar Musliyana, Arif, & Munadi, 2016). As well as MySQL which is an open-source and also has a large user community (Perkasa & Setiawan, 2018; Tongkaw & Tongkaw, 2017). The Google Apps API authentication method used in this study is OAuth 2.0(Xie, Huang, Yang, & Yang, 2016)

The results of this study indicate that the proposed system can handle the process of managing student emails integrated with UUI academic information system data. The system designed can integrate email services from Google Apps with the campus academic information system so students can activate email independently without having to be manually registered to the Google Apps page by the campus email managers.

Theoretical Basis

This section will explain some related theoretical bases from similar studies that have been carried out and also from other related literature regarding problems and solutions proposed in the study.

Google API Services

Google is a top information technology company. It is also one of the successful company with its innovations through the Loon project for spreading internet networks to remote areas a few years ago (Zulfan & Chaidir, 2016). In the digital field, Google provides many services, including API provided by Google Framework. Currently, Google provides various APIs that can be used by users/application developers to be able to use Google services through third-party applications such as email services, Google Maps, Google Classroom, and Google Docs (Hairah & Budiman, 2017; Treude & Aniche, 2018).

API is a collection of program/function codes that serve as a connector between applications, both the same and different platforms (Hradil & Sklenák, 2017; Lensakom, 2016). API services can be used by writing certain syntax/commands through programs provided by providers to be able to access certain services provided. In more detail, the concept of API can be seen in figure 1.

Figure 1. The Concept of API

The Protocol of OAuth2

OAuth is the industry-standard protocol that focuses on providing third-party application authorization services to access resources stored on servers (Xie et al., 2016). This protocol continues to be developed starting from the first version until the version which is widely used now, i.e. OAuth2. OAuth server works by authenticating and authorizing (Ferry, Raw, & Curran, 2015; Jung & Jung, 2017). If both processes are successful, the client will be given a token code by the OAuth server. Then the client can access the data source provided by the Resource Server using token access via an API.

Resource Server in the OAuth2 protocol consists of several parts, namely Authentication Components, Consent Components, and Token Management (Febrian, 2017).

1. Authentication Component is the component that is responsible for providing the login page to the user.

2. Consent Component is the component to provide approval information for third-party applications to access information from users.
3. Token Management is the component that provides a function to check/validate token code that has been provided by the OAuth Provider to users.

The working of OAuth2 can be easily understood through the Grant Type scheme. This scheme explains the mechanisms of access provision to resources on a server as shown in figure 2 (Febrian, 2017)

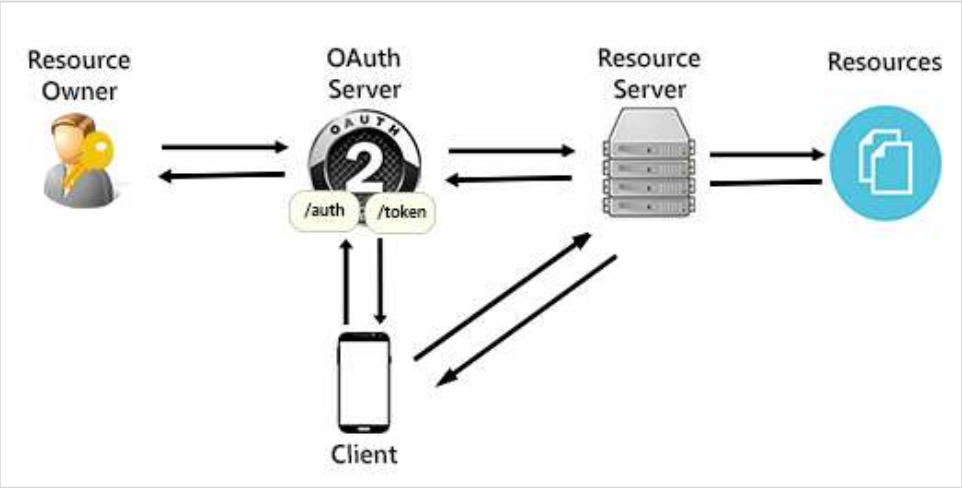

Figure 2. Grant Type Scheme on OAuth2

In Figure 2, there are several stages in the mechanism of OAuth2 Grant Type, namely:

1. Authorization Code
   At this stage, the client/user will send an authorization code request via the '/auth' endpoint. The purpose of this process is to get token code access
2. *Get the Token*
   At this stage, client/user requests to get access to the token code which can later be used to access resource. This process is done by making requests via endpoints/tokens using the POST method via HTTP basic. Data sent via HTTP basic at this stage are username and password of the client (third party application) and also the authorization code parameter (which is obtained at the first stage). Furthermore. OAuth server will check the validity of the data sent by the client, if it is a valid client, then will get access from the OAuth server.
3. *Use the Token to Access a Resource*
   At this stage, the client/user accesses the resource using the token that was provided at the previous stage. Access is done through the endpoint address using GET or POST methods, according to the service provided by Resource Server. In the next process, the Resource Server will check the validity of the token sent by the client by sending it to the OAuth server.

OAuth server will check the token code and send the results to the resource server. Furthermore, if the results are valid, the resource server will provide access to the information needed client.

PHP

PHP is a programming language that can be used to design a dynamic website (W3Techs, 2019). This programming language requires a server to compile a program so that it is categorized into server-side scripting programming language (Z. Musliyana, Dwipayana, Helinda, & Maizi, 2018). PHP PHP is the programming language designed for web development and can also be used as a general programming language (Munadi et al., 2016; Zuhar Musliyana et al., 2016). This language is managed by The PHP Group and developed by Rasmus Lerdorf.

MySQL Database

MySQL is a type of database that falls into the Relational Database Management System (RDBMS) category that can be used to manage databases using Structured Query Language (SQL) (Perkasa & Setiawan, 2018; Tongkaw & Tongkaw, 2017). The characteristic of this database that becomes the main concept in the database is that the process of selecting or inputting data can be done easily.

Methodology

This section will explain the system design scheme and the system design stages which consist of designing Data Flow Diagrams (DFD) and Entity Relationship Diagrams (ERD). The next stage is the implementation and testing of the system.

System Design Scheme

The system design scheme proposed in this study can be seen in Figure 3.

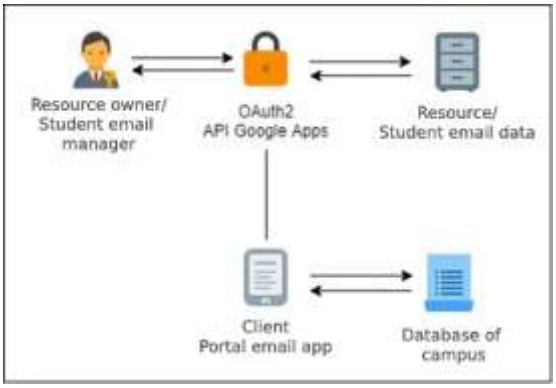

Figure 3. System design scheme

In figure 3 the proposed system scheme consists of email portal application (client) which is in charge of requesting access to resources (student

email data) on google apps through API with OAuth2 authentication using the credential account of student email manager. This process occurs in accordance with the OAuth2 grant type flow mechanism described in the previous section, i,e the request to obtain authorization code, token and the use of the token to access resources on google apps.

After successfully passing OAuth2 authorization and authentication, then the client can make a request for student email data management by sending a request to Google API endpoint that has been provided. When conducting the activation/email creation process via API, the client will check students' identities in the academic database for validation purposes. After the process is successful, the client will save the data in the database of email portal application.

Data Flow Diagram (DFD)

In this study, DFD design is designed to make system modelling based on data flow. This design can be seen in Figure 4.

Figure 4. A Level 0 DFD

In Figure 4 the process of email management consists of two main entities, i.e. the entities of manager and students. Both of these entities can be directly involved in the process of email management but have their respective functions and duties. DFD in Figure 4 can be described in more detail in level one DFD that can be seen in Figure 5.

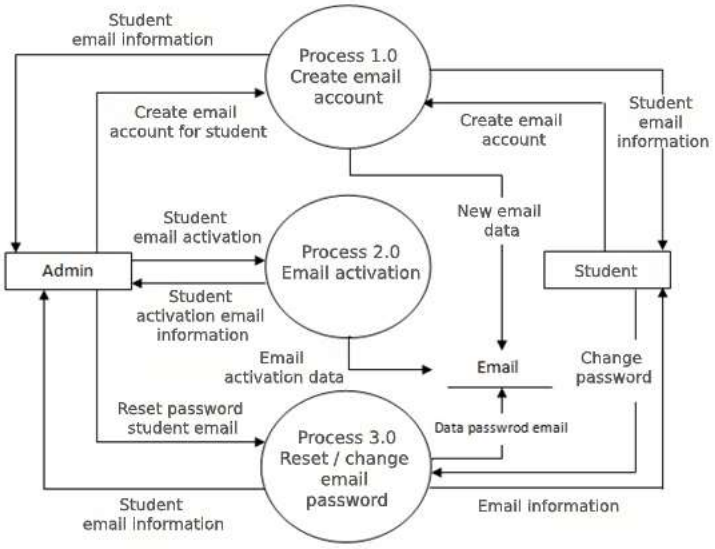

Figure 5. A level 1 DFD

In Figure 5, email management proposed in this study consists of three main processes, namely creating an email, activating email and changing the email password. All three processes involve both entities, manager and students. But the difference in the data flow is that student entity can only carry out activities to manage student-related data. Unlike the admin entity that can manage email data for all student data.

Entity Relationship Diagram (ERD)

ERD is modelling designed to describe the relationship of data in the system proposed in this study. This ERD diagram consists of three main entities, namely manager, students and email. Relationships between these three entities are defined in the related form, as shown in Figure 6.

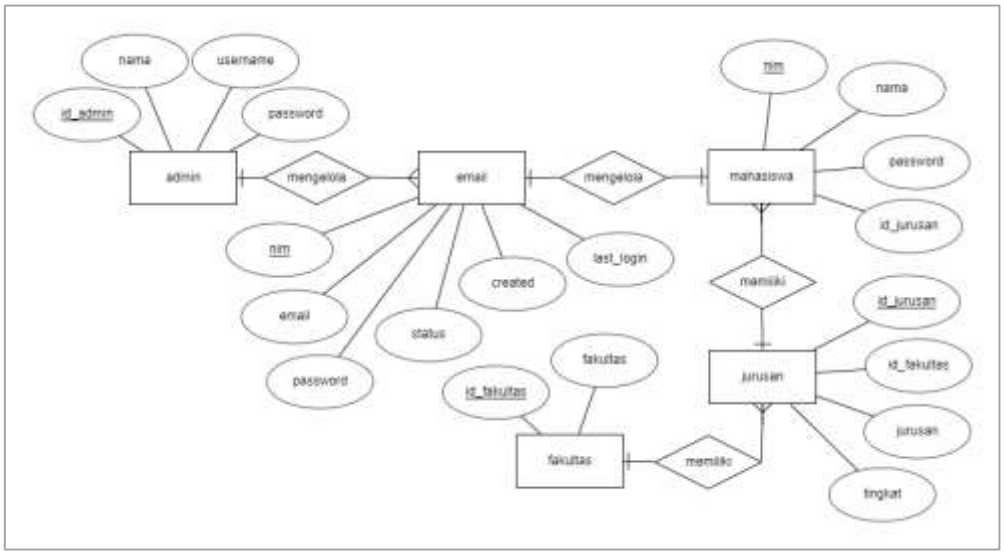

Figure 6. ERD of integrated email management using Google API

In Figure 6, the entities of students, faculty, and department are those whose data sources come from the database of UUI campus academic information system. The entities of department and students have a one-to-many relationship where each department has many students. Meanwhile, the entities of faculty and department also have a one-to-many relationship where each faculty can have several departments in it. While the entities of email and manager are additional ones that are added to store the system manager data and student email account information data which will be synchronized to google apps using API.

System Testing Mechanism

At this stage, several related tests will be conducted to evaluate the integration mechanism of Google API proposed in this study, including:

1. The test of Google API

2. The test of data integration of academic information system with the proposed system which includes the student email management by the manager as well as independent email management by students.

Results and Discussion

The Google API test was carried out to ensure all functions provided in API can run properly to minimize errors when implemented on the system.

The test carried out is by adding/creating new email via the Google API with the commands as in codes of a source below.

```
$client=getClient();
$service=new Google_Service_Directory($client);
$userInstance=new Google_Service_Directory_User();
$nameInstance=new Google_Service_Directory_UserName();
$nameInstance->setGivenName('Tes 41');
$nameInstance->setFamilyName('API');
$userInstance->setName($nameInstance);
$userInstance->setHashFunction("MD5");
$userInstance->setPrimaryEmail('tes41_api@student.uui.ac.id');
$userInstance->setPassword(hash("md5","tes41_api"));
try{
        $createUserResult=$service->users->insert($userInstance);
        var_dump($createUserResult);
}
catch(Google_IO_Exception $gioe){
        echo"Error in connection: ".$gioe->getMessage();
}
catch(Google_Service_Exception $gse){
        echo"User already exists: ".$gse->getMessage();
}
```

Figure 7. Codes of Source Commands to create email account via the API

The above commands have the function to add/create a new email account on Google Apps with the email address of tes41_api@student.uui.ac.id and the password of tes41_api. In the above commands, it can also be seen that Google Apps uses the MD5 algorithm to secure email account password. The results of the command execution can be seen in Figure 8.

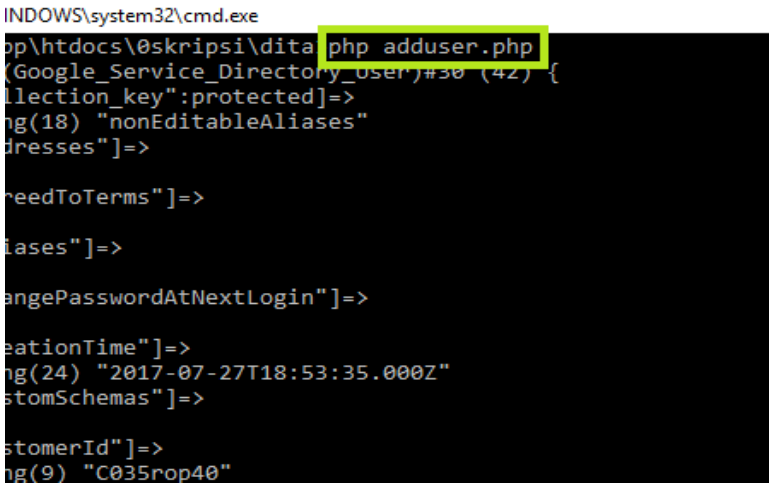

Figure 8. Creating a new email via Google API

Figure 8 above shows the output of the process of adding a new email account that has been successfully carried out via API.

Integration Results of Google API with academic information system data

The integration of Google API and the system was done to facilitate the manager and students in managing email through the Graphical User Interface (GUI). The results of system design with the integration of Google API and academic information system data can be explained as follows.

a. Create new email

The part of this function is used to create new student email that will later be sent to Google Apps via API. The interface for making student email via this system can be seen in Figure 9.

Figure 9. The interface for the process of creating a new student email

Figure 9 is an interface for the process of creating a student email. In figure 8, the student email address that will be created is tes12_api@student.uui.ac.id. After the data is saved and the process is successful, then tes12_api@student.uui.ac.id can be used as shown in Figure 10.

Figure 10. Student email

b. Edit and delete email

This facility provides a function to change student email passwords and also delete email data as shown in Figure 11.

Figure 11. The edit, delete and search functions on student email

c. Email activation (student level)

Through this service, students can register email independently via a link for email as shown in Figure 12.

Figure 12. The interface of student email activation

d. Change email password (student level)

This facility can be used by students to change their e-mail login password at any time. This interface of this function can be seen in Figure 13 .

Figure 13. The interface for changing student email passwords

System security discussion

Although the proposed system uses the OAuth2 protocol which is widely used today, the security issue is the important thing that needs to be studied. Currently, there are similar studies that discuss safety in OAuth2, including security against phishing (Xie et al., 2016). OAuth2 is a protocol that uses an authentication and authorization process to be able to access data resources.

As in the case study discussed in this study, to be able to manage student email through the campus academic data portal application (third party apps), an email account that has a role/ right to access an email manager on google apps is needed. This account is used to send an access request to student email data resources on google apps via the OAuth2 protocol.

The issue of phishing attacks on OAuth2 occurs on the client-side, namely the method of getting a user an email password that has a role as an account manager on google apps with a certain method. After getting an account, the attacker can send a request to the OAuth2 protocol to get student email data information. Therefore, in this case, third-party apps must be able to provide additional security methods to client's OAuth2 email account, for example, through encryption or other methods. It also needs studies through further research to examine the security issues in the OAuth2 protocol.

Conclusions

Below are some conclusions from this research were the system proposed in this study can be integrated with academic information system data so that students can manage their email accounts independently. And then with this system, student email management can be done quickly and easily without having to manually register email accounts into Google Apps portal so that the process of managing student email becomes more efficient.

References

Febrian, T. (2017). Memahami OAuth 2.0 (API Security). Retrieved December 13, 2019, from UNIKOM Codelabs website: https://medium.com/codelabs-unikom/memahami-oauth-2-0-api-security-9376bc3a307b

Ferry, E., Raw, J. O., & Curran, K. (2015). Security evaluation of the OAuth 2.0 framework. *Information and Computer Security*.

https://doi.org/10.1108/ICS-12-2013-0089

Hairah, U., & Budiman, E. (2017). Pemanfaatan Google Maps Api Dalam Pengembangan Media Informasi Pasar Malam Di Kota Samarinda. *ILKOM Jurnal Ilmiah*. https://doi.org/10.33096/ilkom.v9i1.104.9-16

Hradil, J., & Sklenák, V. (2017). Practical Implementation of 10 Rules for Writing REST APIs. *Journal of Systems Integration*. https://doi.org/10.20470/jsi.v8i1.290

Jung, S. W., & Jung, S. (2017). Personal OAuth authorization server and push OAuth for Internet of Things. *International Journal of Distributed Sensor Networks*. https://doi.org/10.1177/1550147717712627

Lensakom. (2016). Mengenal Konsep API pada Pemrograman. Retrieved November 16, 2019, from Lensakom website: https://www.lensakom.com/mengenal-konsep-api-pada-pemrograman/

Munadi, R., Musliyana, Z., Arif, T., Azmi, A., & Syahrial, S. (2016). Kombinasi Waktu Sinkronisasi dan Nilai Salt untuk Peningkatan Keamanan pada Single Sign-On. *Jurnal Nasional Teknik Elektro Dan Teknologi Informasi (JNTETI)*, *5*. https://doi.org/10.22146/jnteti.v5i3.257

Musliyana, Z., Dwipayana, M., Helinda, A., & Maizi, Z. (2018). Improvement of Data Exchange Security on HTTP using Client-side Encryption. *Journal of Physics: Conference Series*, *1019*(1). https://doi.org/10.1088/1742-6596/1019/1/012073

Musliyana, Zuhar, Arif, T. Y., & Munadi, R. (2016). Peningkatan Sistem Keamanan Autentikasi Single Sign on (SSO) Menggunakan Algoritma AES Dan One-Time Password Studi Kasus: SSO Universitas Ubudiyah Indonesia. *Jurnal Rekayasa Elektrika*, *12*(1), 21–29.

Perkasa, M. I., & Setiawan, E. B. (2018). Pembangunan Web Service Data Masyarakat Menggunakan REST API dengan Access Token. *Jurnal ULTIMA Computing*. https://doi.org/10.31937/sk.v10i1.838

Tongkaw, S., & Tongkaw, A. (2017). A comparison of database performance of MariaDB and MySQL with OLTP workload. *ICOS 2016 - 2016 IEEE Conference on Open Systems*. https://doi.org/10.1109/ICOS.2016.7881999

Treude, C., & Aniche, M. (2018). Where does Google find API documentation? *Proceedings - International Conference on Software Engineering*. https://doi.org/10.1145/3194793.3194796

W3Techs. (2019). Usage of server-side programming languages for websites. Retrieved November 16, 2019, from W3Techs website: https://w3techs.com/technologies/overview/programming_language

Xie, M., Huang, W., Yang, L., & Yang, Y. (2016). VOAuth: A solution to protect OAuth against phishing. *Computers in Industry*, *82*, 151–159. https://doi.org/10.1016/j.compind.2016.06.001

Zulfan, D. T., & Chaidir, M. (2016). Google Loon Sebagai Solusi Terkini Koneksivitas Internet di Daerah Pedalaman dan Terpencil Google Loon Sebagai Solusi Terkini Koneksivitas Internet Di Daerah Pedalaman. In *Elkawnie: Journal of Islamic Science and Technology*.