Engineering And Technology International Journal
July 2023 | Vol. 5 | No. 2
E-ISSN : 2714-755X
DOI: 10.55642/eatij.v5i02

Engineering And Technology International Journal (EATIJ)

Cloud Computing Implementation In Early Detection of PCOS Disease Application

Mordekhai Gerine Lumangkun*[1], Fetty Try Anggraeny[2]
[1,2]Informatics Study Program, Faculty of Computer Science
Universitas Pembangunan Nasional "Veteran" Jawa Timur
(0623) 18706369
e-mail: *[1]**mordekhaigerinlumangkun@gmail.com**, [2]fettyanggraeny.if@upnjatim.ac.id

Abstract

This project aims at healthcare especially for women. This project is an application-based model to predict the risk of someone getting a syndrome called PCOS, based on their condition and will be deployed to Google Cloud. The risk is divided into two categories, which are low and high. After the model successfully classifies the risk, the application will not show the result to the user explicitly, but the application will give some recommendations or health tips according to the risk that has been classified. The results will differ among patients with certain categories and it is generated by chat GPT by OpenAI based on the model classification and the user data. This article will explain the role of Cloud Computing in the development of an early detection application for Polycystic Ovary Syndrome (PCOS) called PCOS Pal.

Keywords— keywords,Cloud Computing,Google Cloud, API

INTRODUCTION

Polycystic Ovary Syndrome (PCOS) is a collection of symptoms resulting from a disorder in the endocrine system that commonly occurs in reproductive-aged women [1]. This condition is usually seen in women who are obese, characterized by irregular menstrual cycles, chronic anovulation, hyperandrogenism, and even infertility.

Epidemiological data estimates that PCOS affects more than 116 million or about 3.4% of women worldwide [2]. For Indonesia, there is no official data on the national prevalence of PCOS in Indonesia. PCOS can cause serious reproductive health problems. PCOS patients may experience infertility, impaired glucose tolerance, depression, and Obstructive Sleep Apnea (OSA). PCOS has also been linked to an increased risk of endometrial cancer and nonalcoholic fatty liver disease (NAFLD) / nonalcoholic steatohepatitis (NASH) [3]. If not treated seriously, PCOS can also increase the risk of other health problems such as diabetes, weight problems, and heart function disorders.

Many women are unaware that they have symptoms or risk factors that increase their likelihood of developing PCOS. Additionally, in Indonesia, there are still many people who have an inadequate concept of health and illness. For example, they may not want to check their health at healthcare facilities until their condition worsens, and they often ignore symptoms that may lead to the disease [4].

Cloud Computing technology, commonly known as the cloud, is new widely used and no longer unfamiliar. Cloud computing technology can save more expenses compared to building network infrastructure for the short term . The main cost in adopting cloud computing technology lies in the connection and data processing according to the needs [5]. For companies or institutions that already have good network infrastructure and technology in place, the implementation of cloud computing technology can be more optimal and efficient [6].

To make it easier for women, an application called PCOS Pal has been developed, incorporating cloud computing technology in its development process. The application's main feature is to detect early signs of Polycystic Ovary Syndrome (PCOS). By leveraging the capabilities of cloud computing, PCOS Pal can efficiently process and analyze large amounts of data, allowing for accurate identification of potential PCOS symptoms. The integration of cloud computing technology in the development of PCOS Pal not only enhances its computational capabilities but also enables seamless scalability. With cloud infrastructure in place, the application can accommodate a growing user base without compromising performance or incurring significant infrastructure costs.

METHOD

The implementation of Cloud Computing in this project follows a Research and Development (R&D) approach, focusing on developing an Android-based application [7]. Among the various application development methodologies available, the chosen approach for this research is the waterfall method. Figure 1 illustrates the diagram of the waterfall method that will be utilized in the application development process.

Figure 1 Waterfall Method

The waterfall method was selected due to its simplicity and ability to encompass all essential aspects of application development. Given the limited research timeframe of approximately one month, this method is well-suited for designing, developing, and testing the application.

Furthermore, the application development project involves a team of experts with diverse skill sets. The team consists of three individuals specializing in machine learning, two individuals proficient in cloud computing, and one individual with expertise in mobile development.

RESULT AND DISCUSSION

I. The Involvement of Cloud Computing in Application Development

Cloud computing has transformed the Information Technology (IT) industry by hosting applications and providing resources (CPU and storage) as services on-demand over the Internet [8]. With the presence of Cloud Computing, developers no longer need to worry about servers and can be more focused on application development.

Cloud Computing plays a significant role in application development. Here are some key roles of cloud computing in application development:

1. Scalability. Cloud Computing allows applications to scale seamlessly based on demand. Developers can easily provision additional computing resources, such as processing power and storage, to accommodate increased user loads without the need for significant infrastructure changes. This ensures that applications can handle varying workloads efficiently [9].
2. Cost Efficiency. Cloud Computing offers cost advantages by eliminating the need for upfront investments in hardware and infrastructure. Developers can leverage cloud services on a pay-as-you-go model (only pay for the resources needed) [10].
3. Accessibility and Availability. Cloud-based applications can be accessed from anywhere and at any time with an internet connection [11].
4. Collaboration and Agile Development. Cloud Computing fosters collaborative development by enabling teams to work together on shared resources. Developers can

leverage cloud-based collaboration tools and version control systems. This promotes efficient collaboration, faster development cycles, and improved productivity [12].

5. Database and Storage Services. Cloud platforms offer scalable and reliable database and storage services [13].
6. Continuous Integration/Continous Deployment. Cloud Computing supports DevOps practices by providing tools and services for continuous integration, continuous delivery, and automated deployment pipelines [14].

II. Implementation of Cloud Computing in Application

In this case, the implementation of Cloud Computing utilizes services from Google Cloud to deploy the backend RESTful API that will be used by the PCOS Pal Android application. The implementation based on the method is as follows:

A. Requirements

In this stage, the developers gather the resources that will be used to solve the problem. In this scenario, the application requires an API for executing logic operations and CRUD operations on the database. Additionally, the developer aims to integrate the application with Chat GPT to generate health tips based on the classification outcomes of a machine learning model developed by the team. The API will be deployed using services from Google Cloud. Moreover, the developer intends to improve productivity by leveraging a CI/CD pipeline. The following are the requirements needed for this project:

a. Google Cloud Platform (The main service to deploy application)
b. Express.js (JavaScript Framework for backend development)
c. App Engine (Google Cloud service to deploy the RESTful API)
d. Cloud SQL (Google Cloud service to store structured data)
e. Cloud Source Repository (Google Cloud service to connect GitHub repo to Google Cloud)
f. Cloud Build (Google Cloud service to make CI/CD pipeline)
g. Cloud Secret Manager (to store the environment variables)
h. Chat GPT API Key (Integration to Chat GPT)

B. Design

During this stage, the developers will review the specifications for the cloud infrastructure to be constructed and proceed with designing the cloud infrastructure accordingly. Here are the result:

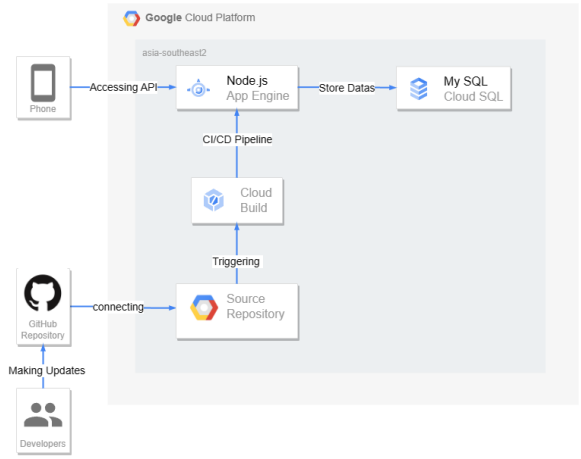

Figure 2 Cloud Infrastructure

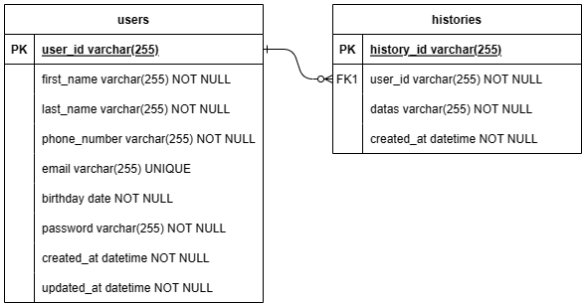

Figure 3 Database Table Design

The developers have designed a database and created cloud infrastructure for the project. The primary reason for using App Engine is because it is easy to deploy, eliminating the need for complex configurations to deploy an application. Additionally, App Engine helps us save costs and offers scalability, autoscaling, built-in services, and integration with various Google Cloud services. In addition, the developer also utilizes the Cloud SQL to store data. In this scenario, the developer uses the MySQL database driver.

For the database design in Figure 3, there are only two tables: the "users" table and the "histories" table. The "users" table will store user account data, including fields such as first name, last name, phone number, email, birthday, password, creation date, and update date. On the other hand, the "histories" table will store the history of PCOS detection, which includes health tips generated by Chat GPT based on the classification results from the machine learning model. Additionally, the "histories" table has a foreign key "user_id", which establishes a relationship between the two tables. The relationship between these two tables is on-to-many. It's important to note that the table design provided here is currently basic and will be further developed in accordance with the evolving features of the PCOS Pal application.

From Figure 2, it can be concluded that the flow of the cloud infrastructure is as follows:

a. From the user's standpoint, the users will utilize API endpoints to perform operations. These API endpoints will be processed in the backend, which will be deployed on App Engine. If there are any interactions with the database during these operations, App Engine will communicate with Cloud SQL to store the data.
b. From the developer's perspective, The developer utilizes a CI/CD pipeline through Cloud Source Repository and Cloud Build. This facilitates the development of process as the GitHub repository is integrated (CI) with Google Cloud via Cloud Source Repository then through Cloud Build, enabling seamless Continuous Deployment (CD) following a well-automated and controller Continuous Integration (CI) process for App Engine.

C. Development

During this stage, the developer initiates the development of the backend RESTful API, which will be utilized by the Android application. Also during this stage, the API will be deployed to Google Cloud using App Engine, and the developer will leverage a CI/CD pipeline through Cloud Source Repository. For the backend development, Node.js is used. Node.js is chosen because it is lightweight and open-source, which makes it easier to build applications. Also, Node.js is designed to build scalable network applications.

To support the development process, Express.js is used. Express.js is a backend framework which is built on top of Node.js, provides a simple and flexible way to build APIs. Express.js has routing feature, making it easy to define routes for handling different HTTP requests such as GET, POST, PUT, DELETE. Furthermore, Express.js has a robust middleware system that allows developers to extend the functionality of their applications. Also Express.js has a large and active community, which has led to the development of numerous third-party middleware, plugins, and extensions. This ecosystem provides developers with a wide range of options and tools to enhance their applications and improve productivity. Since Node.js is an open-source platform, several libraries such as bcrypt, jsonwebtoken, mysql2, sequelize, nanoid,

openai, nodemon, and body-parser are utilized during the development phase. These libraries enhance the functionality and capabilities of the application.

Overall, the developers have developed 9 RESTful API endpoints and deployed to Google Cloud using App Engine. Here are the API endpoints results.

Table 1 API Endpoints

Method	Endpoint	Function
[GET]	/api/v1	Homepage
[POST]	/api/v1/register	Register user account
[POST]	/api/v1/login	Login user account
[GET]	/api/v1/auth/user	Show user data
[POST]	/api/v1/auth/calculate-bmi	Calculate bmi
[POST]	/api/v1/auth/pcos	Create user answer (survey
[POST]	/api/v1/auth/pcos/result	Generate tips and trick with Chat GPT
[POST]	/api/v1/auth/pcos/history	Store generated tips and trick to database

During this stage, the developer also deploys the RESTful API to Google Cloud by referring to the previously created Cloud Infrastructure, as depicted in Figure 2. Here are the results.

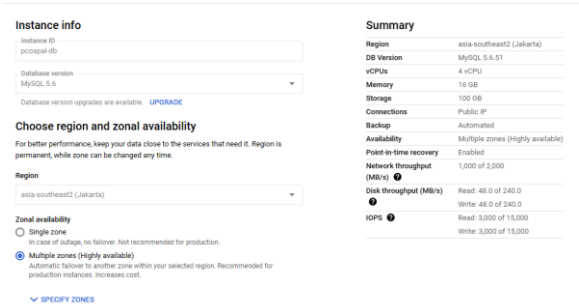

Figure 4 Cloud SQL Instance Configuration

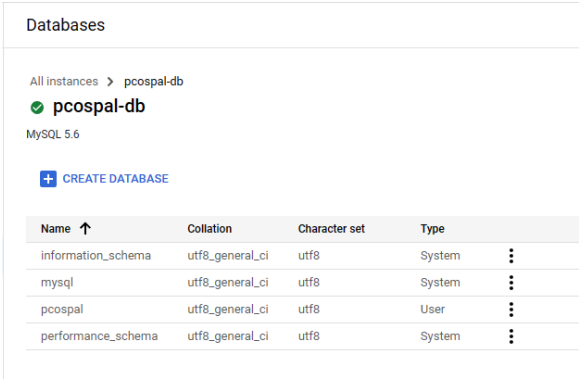

Figure 5 MySQL Database

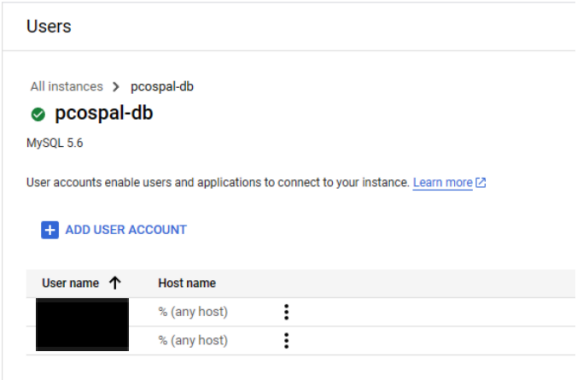

Figure 6 MySQL User Account

The developer has created a Cloud SQL instance, as seen in Figure 4, named "pcospal-db", using MySQL as the database driver with version 5.6. The instance is created in the asia-southeast2 region (Jakarta) with multiple zones configuration. Since it is still in the development stage and for testing purposes in the future, the desired configuration is to use 4 vCPUs and 16GB of memory to optimize cost. In Figure 5, the developer creates a database named "pcospal" within the "pcospal-db" instance. Additionally, the developer creates an user account for the "pcospal-db" instance to enable connectivity and usage with the API.

To deploy the created RESTful API, the developer utilizes Google Cloud's service called App Engine. The deployment process itself is straightforward and does not require many additional configurations. Here are the results of the deployment.

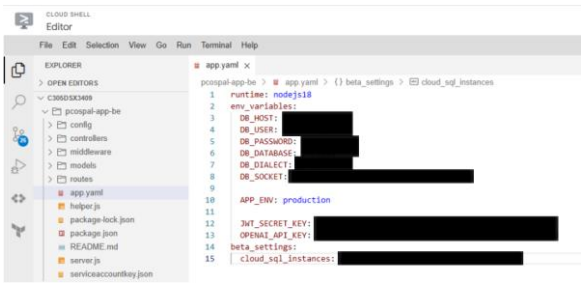

Figure 7 app.yaml Configuration

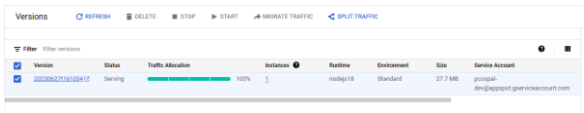

Figure 8 Deployed API

Before deploying the API to the App Engine, the first step is to clone the backend RESTful API from the GitHub repository into Google Cloud. Then, the developer needs to create an app.yaml file, as shown in Figure 7. The app.yaml file is a configuration file used to deploy applications to Google App Engine. Its main function is to manage the environment settings, scaling, and resources required by the application on App Engine. In this case, the developer sets the runtime to nodejs18 since the API is based on Node.js version 18. Additionally, the developer defines environment variables within the app.yaml file. The "DB_HOST", "DB_USER", "DB_PASSWORD", "DB_DATABASE", and "DB_DIALECT" settings are related to the MySQL database used by the application. The "DB_SOCKET" represents the database socket for connecting to Google Cloud SQL. "APP_ENV" determines the application environment, which in this case is set to "production". The "JWT_SECRET_KEY" is a secret key used for encrypting and decrypting JSON Web Tokens. The "OPENAI_API_KEY" is the API key obtained from OpenAI to use Chat GPT. Furthermore, there is a "beta_settings" section that enables the use of Google Cloud SQL. The "cloud_sql_instances" specifies the Google Cloud SQL instance to be used by the application.

After configuring the app.yaml file, the next step is to open the Cloud Shell terminal on the Google Cloud Platform and enter the following command: "gcloud app deploy". This command is used to deploy the API based on the configurations defined in the app.yaml file. Upon successful execution of the "gcloud app deploy" command, the App Engine instance and version will be automatically created. This can be observed in Figure 8.

To set up the CI/CD pipeline, the developer connects the GitHub repository via Cloud Source Repository. Then to make the automated deployment, the developer needs to set up Cloud Build and make the trigger. Here are the results.

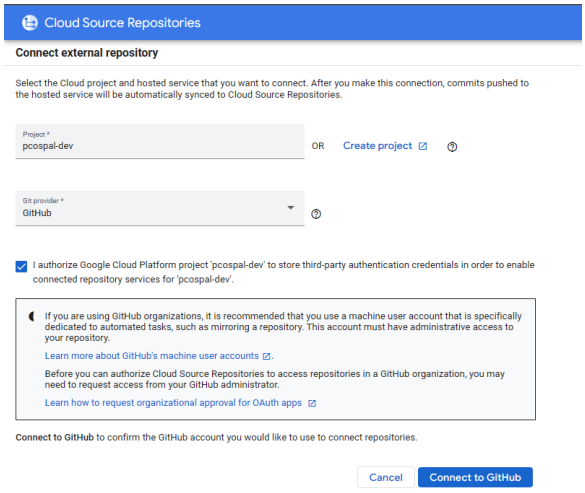

Figure 9 Connecting to GitHub Repository

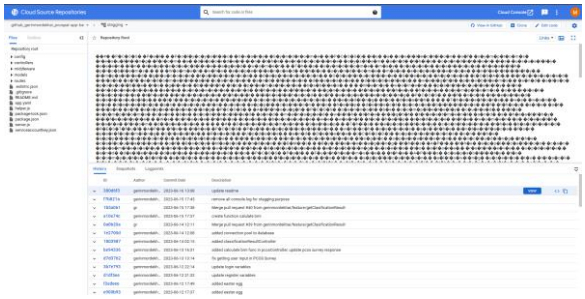

Figure 10 Connected Cloud Source Repository

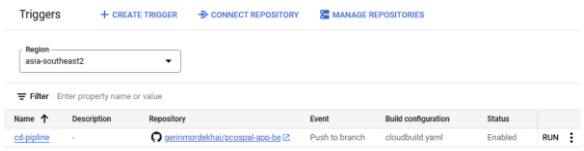

Figure 11. Cloud Build Trigger

To automate cloud build, it is necessary to create a file called cloudbuild.yaml and make some changes to the app.yaml file. The cloudbuild.yaml file is a configuration file used in the Google Cloud Build service. Its function is to define the steps that will be executed during the build and deployment process of an application or project on the Google Cloud Platform. In this case, is to define the steps that will be executed during the build from the GitHub repository and deployment to App Engine. Here are the result.

The app.yaml file:

```
runtime: nodejs18

env_variables:
  DB_HOST:                    <db-host-value>
  DB_USER:                    <db-user-value>
  DB_PASSWORD:                <db-password-value>
  DB_DATABASE:                <db-database-value>
  DB_DIALECT:                 <db-dialect-value>
  DB_SOCKET:                  <db-socket-value>

  APP_ENV:                    <app-env-value>

  JWT_SECRET_KEY:             <jwt-secret-key-value>
  OPENAI_API_KEY:             <openai-api-key-value>

beta_settings:
  cloud_sql_instances: <cloud-sql-instances-value>
```

The cloudbuild.yaml file:

```
steps:
- name: 'gcr.io/google.com/cloudsdktool/cloud-sdk'
  entrypoint: 'bash'
```

```
  args:
    - '-c'
    - |
      sed -i "s/<db-host-value>/$$DB_HOST/" app.yaml
      sed -i "s/<db-user-value>/$$DB_USER/" app.yaml
      sed -i "s/<db-password-value>/$$DB_PASSWORD/" app.yaml
      sed -i "s/<db-database-value>/$$DB_DATABASE/" app.yaml
      sed -i "s/<db-dialect-value>/$$DB_DIALECT/" app.yaml
      sed -i "s/<db-socket-value>/$$DB_SOCKET/" app.yaml
      sed -i "s/<app-env-value>/$$APP_ENV/" app.yaml
      sed -i "s/<jwt-secret-key-value>/$$JWT_SECRET_KEY/" app.yaml
      sed -i "s/<openai-api-key-value>/$$OPENAI_API_KEY/" app.yaml
      sed -i "s/<cloud-sql-instances-value>/$$cloud_sql_instances/" app.yaml
  secretEnv:
    - DB_HOST
    - DB_USER
    - DB_PASSWORD
    - DB_DATABASE
    - DB_DIALECT
    - DB_SOCKET
    - APP_ENV
    - JWT_SECRET_KEY
    - OPENAI_API_KEY
    - cloud_sql_instances
- name: 'gcr.io/google.com/cloudsdktool/cloud-sdk'
  entrypoint: 'bash'
  args: ['-c', 'gcloud config set app/cloud_build_timeout 1600 && gcloud app deploy']
availableSecrets:
  secretManager:
    - versionName: projects/531807919327/secrets/DB_HOST/versions/1
      env: 'DB_HOST'
    - versionName: projects/531807919327/secrets/DB_USER/versions/1
      env: 'DB_USER'
    - versionName: projects/531807919327/secrets/DB_PASSWORD/versions/1
      env: 'DB_PASSWORD'
    - versionName: projects/531807919327/secrets/DB_DATABASE/versions/1
      env: 'DB_DATABASE'
    - versionName: projects/531807919327/secrets/DB_DIALECT/versions/1
      env: 'DB_DIALECT'
    - versionName: projects/531807919327/secrets/DB_SOCKET/versions/1
      env: 'DB_SOCKET'
    - versionName: projects/531807919327/secrets/APP_ENV/versions/1
      env: 'APP_ENV'
    - versionName: projects/531807919327/secrets/JWT_SECRET_KEY/versions/1
      env: 'JWT_SECRET_KEY'
    - versionName: projects/531807919327/secrets/OPENAI_API_KEY/versions/1
      env: 'OPENAI_API_KEY'
    - versionName: projects/531807919327/secrets/cloud_sql_instances/versions/1
      env: 'cloud_sql_instances'
timeout: '1600s'
logsBucket: pcospal_cloud_build_logs
```

From the cloudbuild.yaml file above, there are two steps and some other configuration defined. Step one, the "name" specifies the docket image to be used for the build step, in this case, it's the Cloud SDK image from Google. The "entrypoint" sets the entrypoint for the docker container to "bash". The "args" contains the command to be executed inside the container. Hare, it uses the "|" symbol followed by a multi-line string to run multiple "sed" commands. These "sed" commands are used to replace placeholder values in the "app.yaml" file with environment variables values. The environment variables values are referenced using "$$" to escape the variable in the YAML file.

Step two, the "name" specifies the docker image to be used for this step, which is again the Cloud SDK image. The "entrypoint" sets the entrypoint for the docker container to "bash". The "args" contains the command to be executed inside the container. It uses "-c" to run a shell command and sets the "app/cloud_builld_timeout" configuration value using "gcloud config set". Finally, it deploys the application using "gcloud app deploy"

The "availableSecrets" section lists the secrets that are stored in Secret Manager and their corresponding environment variables. Each secret is defined with a "versionName" that includes the project ID, secret name, and version number. The "env" field specifies the environment variable name to which the secret value will be assigned. The "timeout" sets the maximum amount of time (1600 seconds) for the build and deployment steps to complete before timing out. The "logsBucket" specifies the Google Cloud Storage Bucket where the build logs will be stored. Here are the successful CI/CD pipeline integration result.

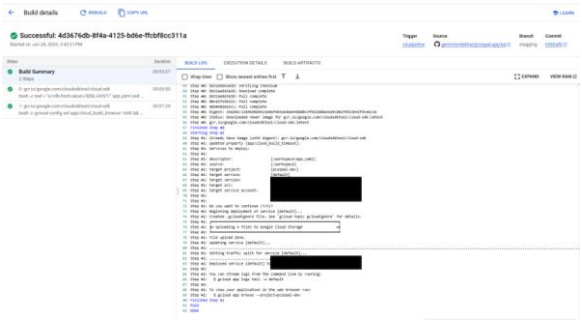

Figure 12 Successful CI/CD Pipeline

D. Testing

Once the development stage is completed, the API will proceed to the testing phase to ensure its functionality, performance and accuracy. The Postman application will be utilized as the testing tool. The API's URL for testing can be obtained from the App Engine, while the API endpoints to be tested can be found in Table 1.

E. Maintenance

After the testing phase is completed, the API is ready to enter the production stage. The integration of a CI/CD pipeline by the developer makes the maintenance phase will become much easier. In case of future changes or the addition of new features, the developer only needs to focus on the code and doesn't have to worry about the deployment process again, as it has been automated through the CI/CD pipeline.

CONCLUSION

Cloud Computing is a new computing and is the future technology that makes it easy for everyone with a variety of slick services. Cloud computing provides the foundation for a cost-effective and successful IT transformation. Cloud computing emerges as a brand new and powerful computing model that ever more companies are willing to adopt in order to improve collaboration. Advantages such as increased IT infrastructure flexibility, computational power, the opportunity to use an existing infrastructure on a pay-per-uses basis, and disaster recovery. In cases like this, cloud computing is the optimal choice for small business or businesses looking to transition to virtualization. Cloud computing simplifies and reduces maintenance efforts since its virtual nature eliminates the need for purchasing and maintaining physical servers, which can be costly.

REFERENCES

[1] Rasquin Leon LI, Anastasopoulou C, Mayrin JV., 2022, Polycystic Ovarian Disease, https://www.ncbi.nlm.nih.gov/books/NBK459251/, retrieved 28 June 2023.

[2] Singh S, Pal N, Shubham S, Sarma DK, Verma V, Marotta F, Kumar M., 2023, Polycystic Ovary Syndrome: Etiology, Current Management, and Future Therapeutics, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9964744/, retrieved 28 June 2023.

[3] Doycheva I, Ehrmann DA., 2022 Nonalcoholic fatty liver disease and obstructive sleep apnea in women with polycystic ovary syndrome, https://pubmed.ncbi.nlm.nih.gov/35512974/, retrieved 28 June 2023.

[4] Elviethasari J., 2020, Knowledge of General Practitioners about Polycystic Ovarian Syndrome at the Primary Health Care in Surabaya, Indonesia, No.8, Vol.70.

[5] Golightly L, Chang V, Xu QA, Gao X, Liu BS., 2022 Adoption of cloud computing as innovation in the organization. *International Journal of Engineering Business Management*, https://doi.org/10.1177/18479790221093992.

[6] Sean Marston, Zhi Li, Subhajyoti Bandyopadhyay, Juheng Zhang, Anand Ghalsasi, 2011, Cloud computing — The business perspective, *Decision Support Systems*, No.1, Vol.51, 176-189, https://doi.org/10.1016/j.dss.2010.12.006.

[7] I Putu W. Ariawan, Wayan Sugandini, I Made Ardana, I Made S. D. Arta, 2022, Design of Formative-Summative Evaluation Model Based on Tri Pramana-Weighted Product, *Emerging Science Journal*, No.6, Vol.6, https://doi.org/10.28991/ESJ-2022-06-06-016.

[8] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, Matei Zaharia, 2010, A view of cloud computing. *Association for Computing Machinery, No.4*, Vol.53, 50–58. https://doi.org/10.1145/1721654.1721672.

[9] Luis M. Vaquero, Luis Rodero-Merino, and Rajkumar Buyya. 2011. Dynamically scaling applications in the cloud, *Association for Computing Machinery*, No.1, Vol.41, 45–52. https://doi.org/10.1145/1925861.1925869.

[10] Mohsen Attaran, 2017, Cloud Computing Technology: Leveraging the Power of the Internet to Improve Business Performance, *Journal of International Technology and Information Technology*, No.1, Vol.26, https://doi.org/10.58729/1941-6679.1283.

[11] Dimiter G. Velev, 2014, Challenges and Opportunities of Cloud-Based Mobile Learning, *International Journal of Information and Education Technology*, No.1, Vol.4, 10.7763/IJIET.2014.V4.367.

[12] Wu, D., Thames, J. L., Rosen, D. W., and Schaefer, D, 2013, Enhancing the Product Realization Process With Cloud-Based Design and Manufacturing Systems, *ASME. J. Comput. Inf. Sci. Eng*, No.4, Vol.13, https://doi.org/10.1115/1.4025257.

[13] J. Wu, L. Ping, X. Ge, Y. Wang and J. Fu, 2010, Cloud Storage as the Infrastructure of Cloud Computing, *International Conference on Intelligent Computing and Cognitive Informatics*, https://doi.org/10.1109/ICICCI.2010.119.

[14] M. Soni, 2015, End to End Automation on Cloud with Build Pipeline: The Case for DevOps in Insurance Industry, Continuous Integration, Continuous Testing, and Continuous Delivery, *IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)*, https://doi.org/10.1109/CCEM.2015.29.