JUSTIN
Jurnal Sistem dan Teknologi Informasi
p-ISSN : 2460-3562 / e-ISSN : 2620-8989
DOI: 10.26418/justin.v11i3.66823
Vol. 11, No. 3, Juli 2023

Rancang Bangun *Back-end* API pada Aplikasi Mobile AyamHub Menggunakan Framework Node JS Express

Eli Nurhayati[1], Agussalim[2]
[1, 2]*Program Studi Sistem Informasi Fakultas Ilmu Komputer, Universitas Pembangunan Nasional "Veteran" Jawa Timur*
Jl. Rungkut Madya No.1, Gn. Anyar, Kec. Gn. Anyar, Surabaya, Jawa Timur 60294
[1]20082010115@student.upnjatim.ac.id
[2]agussalim.si@upnjatim.ac.id

Abstrak

Back-end adalah bagian aplikasi yang beroperasi di sisi server dan bertanggung jawab dalam interaksi dengan database dan logika fungsional sebuah program. *Back-end* berkomunikasi dengan client melalui sebuah antarmuka Application Programming Interface (API). Salah satu arsitektur *back-end* yang paling populer yaitu Representational State Transfer (REST) dimana interaksi yang dilakukan menggunakan pola request-response. AyamHub adalah sebuah aplikasi mobile inovatif yang berfungsi sebagai penghubung antara peternakan dan UMKM/penjual ayam broiler di Indonesia. Dalam proses pengembangannya, AyamHub membutuhkan sebuah *back-end* yang akan menjadi fondasi sistem kedepannya. Untuk itu, peneliti mencoba melakukan perancangan arsitektur *back-end* untuk aplikasi mobile AyamHub menggunakan arsitektur REST dan framework Node.js Express. Pemilihan Node.js sebagai platform pengembangan didasarkan pada kelebihan yang dimiliki, terutama pada portabilitas dan teknik non-blocking yang memungkinkan sistem bekerja secara simultan tanpa harus menunggu penyelesaian operasi sebelumnya. Adapun pengembangan dilakukan dengan menggunakan metode waterfall dimana implementasi sistem dibuat dengan menggunakan layanan Google *Cloud Platform* (GCP) seperti App Engine, Cloud SQL, dan Cloud Storage. Setelah dilakukan pengujian menggunakan black box testing, diperoleh hasil bahwa keseluruhan API yang telah dibuat telah berhasil dan berjalan sesuai dengan harapan. Selain itu, dalam setiap tahap pengembangan *back-end*, penulis selalu berkoordinasi dengan mobile developer AyamHub untuk memastikan consume API berjalan lancar dan endpoint yang dibuat selaras dengan setiap kebutuhan fiturnya.

Kata kunci: *Back-end, Node.js, REST API,* AyamHub, Google *Cloud Platform* (GCP)

Back-end API Design for AyamHub Mobile Application Using Node JS Express Framework

Abstract

Back-end is the part of the application that operates on the server side and responsible for interaction with the database and functional logic of a program. *Back-end* communicates with the client through an Application Programming Interface (API). One of the most popular *back-end* architectures is Representational State Transfer (REST) where the interaction is done using a request-response pattern. AyamHub is an innovative mobile application that serves as a link between farms and MSMEs/broiler chicken sellers in Indonesia. In the development process, AyamHub requires a *back-end* that will become the foundation of the system in the future. For this reason, researchers are trying to design a *back-end* architecture for the AyamHub mobile application using the REST architecture and the Node.js Express framework. The choice of Node.js as a development platform is based on its advantages, especially in portability and non-blocking techniques that allow the system to work simultaneously without having to wait for the completion of previous operations. The development is carried out using the waterfall method where the system implementation is made using Google Cloud Platform (GCP) services such as App Engine, Cloud SQL, and Cloud Storage. After testing using black box testing, the results show that the entire API that has been created has been successful and runs as expected. In addition, at every stage of *back-end* development, author always coordinates with AyamHub's mobile developers to ensure that the consumption API runs smoothly and the endpoints created are aligned with each feature requirement.

Keywords: *Back-end, Node.js, REST API,* AyamHub, Google *Cloud Platform* (GCP)

I. PENDAHULUAN

Back-end adalah suatu bagian dari aplikasi yang beroperasi pada sisi server dan bertanggung jawab untuk berinteraksi langsung dengan database, khususnya terkait manipulasi data seperti menyimpan, mengambil, memperbarui, dan menghapus data [1]. Meskipun berperan langsung dalam pengolahan data, *back-end* tidak berinteraksi langsung dengan pengguna. Sebagai gantinya, *back-end* menyediakan antarmuka yang digunakan oleh aplikasi *client* untuk berkomunikasi dengan data melalui API (*Application Programming Interface*) [2]. API sendiri merupakan sebuah antarmuka yang menjadi penghubung antara sistem aplikasi yang berbeda sehingga sebagian atau keseluruhan fungsi sistem dapat diakses secara bersamaan [3]. API dapat meningkatkan efisiensi kinerja developer karena memungkinkan penggunaan fungsi dari aplikasi yang sudah ada tanpa harus mengembangkannya dari awal.

Salah satu arsitektur *back-end* yang populer adalah *Representational State Transfer* (REST). REST adalah sebuah gaya arsitektur untuk sebuah sistem yang terdistribusi dimana berfokus pada skalabilitas dari interaksi antar komponen sistem dan generalitas sebuah antarmuka [4]. Dalam berinteraksi, REST menggunakan pola *request-response* dengan memanfaatkan protokol HTTP [5]. REST bekerja dengan memisahkan peran antara client dan server sehingga jika terjadi perubahan di salah satu sisinya, maka tidak akan berdampak pada sisi lainnya. Adapun REST API ialah istilah yang digunakan untuk merujuk pada layanan web yang mengadopsi arsitektur REST sebagai antarmuka pemrograman aplikasi (API) [6].

Dalam pembangunan *back-end* server, diperlukan penggunaan bahasa pemrograman yang dijalankan di sisi server (*server-side*) sebagai bagian integral dari proses tersebut. Penelitian sebelumnya dilakukan oleh T. Bratakusuma, I. U. Azmi, dan S. Ayuningtiyas yang mengkaji pembangunan sistem *back-end* dengan menggunakan API REST pada Aplikasi Alat Tulis Kantor Bank Indonesia Perwakilan Purwokerto. Penelitian tersebut menggunakan Node.js sebagai platform pengembangan dikarenakan portabilitas pada berbagai sistem operasi tanpa harus mengubah kode program [7]. Selain itu, penelitian lain dilakukan oleh Hasanuddin, H. Asgar, dan B. Hartono dalam perancangan *back-end* server menggunakan arsitektur rest dan platform Node.js pada rancang bangun Rest API Aplikasi Weshare sebagai upaya mempermudah pelayanan donasi kemanusiaan. Meskipun dalam konteks aplikasi yang berbeda, alasan penggunaan Node.js pada penelitian ini hampir sama pada penelitian sebelumnya. Selain portabilitas, NodeJS dilengkapi dengan pustaka server HTTP internal yang memungkinkan penggunaan server web tanpa ketergantungan pada program server web seperti Apache atau Nginx [8].

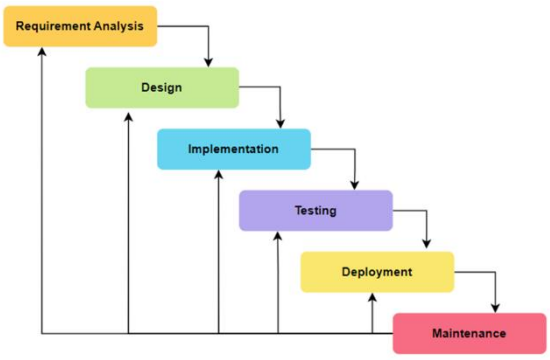
Beralih dari hal tersebut, Node.js adalah platform berbasis javascript *runtime* dengan skalabilitas tinggi yang dapat mengeksekusi kode javascript di luar lingkungan browser [9]. Kelebihan Node.js terletak pada pendekatan *non-blocking* yang memungkinkan sistem untuk menjalankan operasi secara simultan tanpa harus menunggu penyelesaian operasi sebelumnya [10]. Hal ini memungkinkan Node.js untuk menangani banyak permintaan secara paralel.

AyamHub merupakan sebuah aplikasi mobile inovatif yang dirancang dengan tujuan utama yaitu sebagai perantara yang menghubungkan penjual ayam broiler dengan peternakan, sehingga memudahkan proses pengadaan dan distribusi produk ayam broiler. Untuk mendukung fungsionalitas aplikasi ini, diperlukan pengembangan *back-end* yang akan menjadi fondasi sistem. *Back-end* ini akan bertanggung jawab dalam mengelola logika bisnis, menyimpan dan memanipulasi data, serta menyediakan antarmuka pemrograman aplikasi (API) untuk komunikasi antara aplikasi mobile dan server. Dengan adanya *back-end* yang terpercaya dan efisien, AyamHub dapat menyediakan pengalaman pengguna yang baik dan memperkuat konektivitas antara UMKM dan peternakan.

Oleh karena itu, pada penelitian ini penulis mengusulkan perancangan sistem *back-end* dan arsitektur REST API pada aplikasi mobile AyamHub. Pada proses pengembangan *back-end* ini, penulis menggunakan framework Node.js Express. Tujuan utama dari penelitian ini adalah untuk menciptakan *back-end* yang efektif dan berfungsi dengan baik untuk aplikasi mobile AyamHub.

II. METODOLOGI

Dalam penelitian ini, pengembangan dan perancangan *back-end* dilakukan dengan menggunakan metode *Waterfall*. Metode *waterfall* adalah sebuah metode pengembangan perangkat lunak yang bersifat sistematis dan berurutan, dimana setiap tahap harus menunggu tahap sebelumnya selesai sebelum dapat melanjutkan ke tahap selanjutnya [11]. Gambar 1 berikut merupakan tahapan-tahapan waterfall yang dilakukan dalam penelitian ini :



Gambar 1. Tahapan Waterfall pada Perancangan *Back-end* Aplikasi Mobile AyamHub

1) Analisis kebutuhan (*Requirement Analysis*)
   Analisis kebutuhan dilakukan dengan menganalisis rancangan sistem aplikasi mobile AyamHub, baik fitur-fitur, proses bisnis, serta kebutuhan *software* maupun *hardware*.
2) Desain (*Design*)
   Pada tahap desain dilakukan proses perancangan arsitektur sistem secara terstruktur dengan mengacu

pada hasil analisis kebutuhan pada tahap awal. Adapun pada tahap desain ini dihasilkan *use case diagram*, *sequence diagram*, *class diagram*, dan struktur database dari aplikasi mobile AyamHub.

3) Implementasi (*Implementation*)

Berdasarkan hasil desain arsitektur sistem yang telah disusun, tahap selanjutnya yaitu berupa implementasi konkret ke dalam kode pemrograman untuk pengembangan bagian *back-end* menggunakan framework Node.js Express.

Dalam hal ini penulis menggunakan Google *Cloud Platform* (GCP) sebagai arsitektur pengembangan. GCP adalah sebuah layanan *public cloud computing* dari Google yang menawarkan beragam layanan *hosting*, termasuk komputasi, penyimpanan, dan pengembangan aplikasi yang berjalan di infrastruktur perangkat keras Google [12]. Google Cloud sendiri memiliki beragam produk yang dapat dipilih oleh developer sesuai kebutuhan pengembangan aplikasi, seperti Compute Engine, App Engine, Cloud Spanner, dll.

4) Pengujian (*Testing*)

Setelah implementasi ke dalam kode selesai, tahapan selanjutnya yakni pengujian sistem. Adapun pada tahap ini penulis menggunakan metode *black box testing* dimana metode ini merupakan metode pengujian yang berfokus pada fungsionalitas perangkat lunak [13]. Metode ini dipilih agar dapat mengetahui keberhasilan seluruh endpoint yang telah dibuat.

5) Deploy (*Deployment*)

Pada tahap ini penulis melakukan deployment agar API *back-end* yang telah dibuat dapat dikonsumsi oleh *mobile development*.

6) Pemeliharaan (*Maintenance*)

Tahapan terakhir yaitu pemeliharaan *back-end* yang telah dibuat guna mencari *error/bug* dalam aplikasi dan memastikan keseluruhan sistem berjalan dengan optimal.

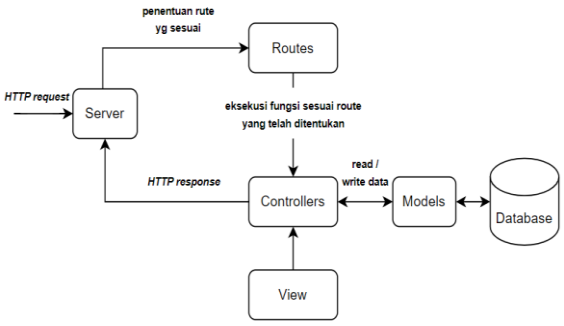
III. HASIL DAN PEMBAHASAN

1) Analisis kebutuhan (*Requirement Analysis*)

Berdasarkan hasil analisis kebutuhan yang telah dilakukan, diperoleh data terkait fitur-fitur yang ingin diimplementasikan pada aplikasi mobile AyamHub. Fitur-fitur yang diidentifikasi mencakup fitur *register*, *login*, dan *logout* pengguna untuk mengelola akses ke aplikasi. Selain itu, terdapat fitur pengguna untuk menambahkan peternakan dan mengelola data peternakan, yang memungkinkan pengguna untuk memantau dan mengatur informasi terkait peternakan mereka. Terdapat pula fitur bookmark peternakan, yang memungkinkan pengguna untuk menyimpan dan mengakses dengan mudah peternakan yang diminati. Terakhir, terdapat fitur deteksi penyakit ayam melalui feses yang bertujuan untuk membantu pengguna dalam mengidentifikasi dan mengatasi masalah kesehatan ayam secara efisien.

2) Desain (*Design*)

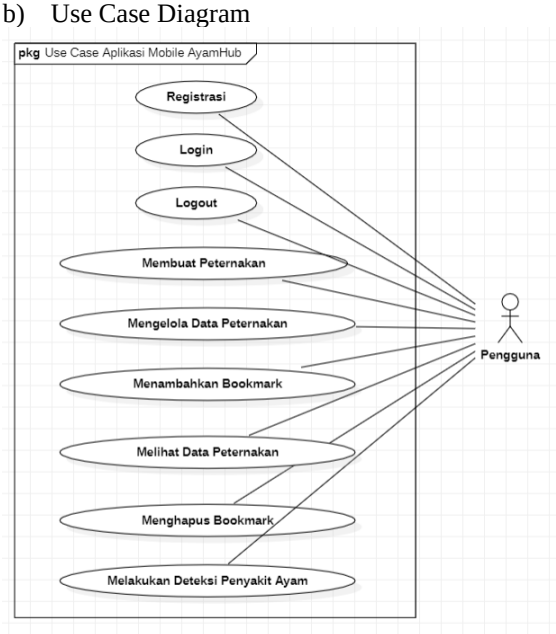
a) Rancangan arsitektur sistem



Gambar 2. Rancangan Arsitektur *Back-end*

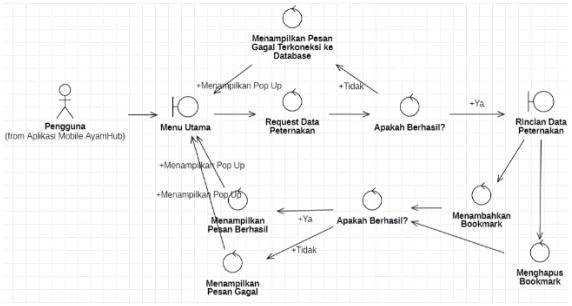
Dalam proses perancangan *back-end* aplikasi mobile AyamHub, terdapat beberapa komponen yang memiliki peran penting, yaitu *database*, *models*, *controllers*, *routes*, dan server. Komponen-komponen ini saling berinteraksi untuk memastikan fungsionalitas aplikasi yang optimal. Pada tahap awal, server berperan sebagai penghubung antara *client* dan aplikasi, menerima *request* dari *client*, dan menentukan jalur (*routes*) berdasarkan metode permintaan yang diterima. Selanjutnya, melalui *controllers*, fungsi-fungsi yang sesuai dengan jalur tersebut dieksekusi. *Controllers* berinteraksi langsung dengan *database* melalui *models* untuk melakukan proses pengambilan dan manipulasi data yang dibutuhkan [14]. Hasilnya kemudian dikirimkan kembali ke *client* sebagai *response* yang sesuai dengan permintaan yang diajukan. Rancangan arsitektur sistem *back-end* aplikasi AyamHub dapat dilihat pada Gambar 2.

b) Use Case Diagram



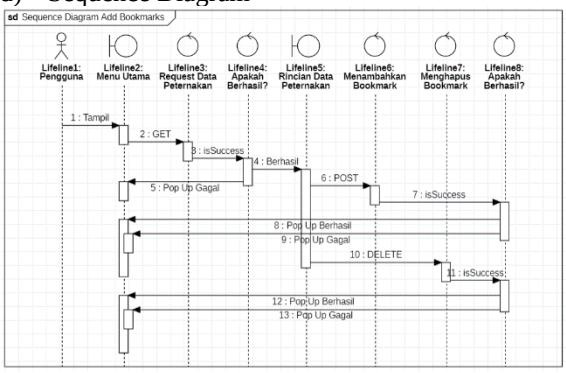
Gambar 3. Use Case Diagram

c) Robustness Diagram



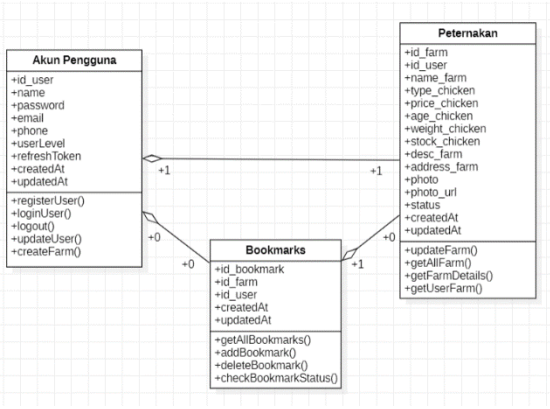
Gambar 4. Robustness Diagram

d) Sequence Diagram



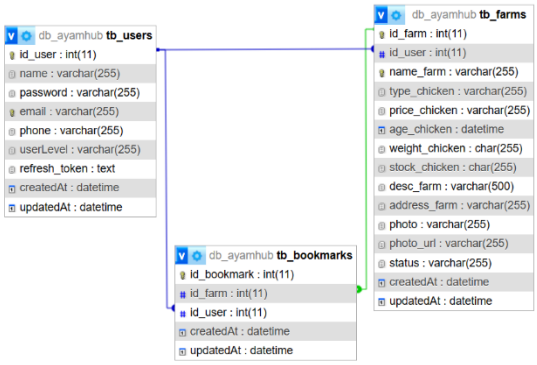
Gambar 5. Sequence Diagram

e) Class Diagram



Gambar 6. Class Diagram

f) Struktur Database



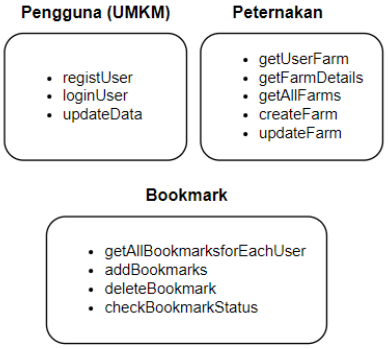
Gambar 7. Struktur Database

Pada Gambar 7 diatas, struktur *database* dari rancangan *back-end* aplikasi *mobile* AyamHub terdiri dari 3 tabel yaitu table *users*, table *farms*, dan table *bookmarks*. Id_user menjadi *foreign key* pada table *farms* dan table *bookmarks*, sedangkan id_farm menjadi *foreign key* pada table *bookmarks*.

Ketiga table ini memiliki relasi yang dibentuk atas ketergantungan satu sama lain. Relasi antara table *users* dan *farms* yakni 1 to 1 dimana 1 pengguna hanya dapat membuat 1 peternakan dan 1 peternakan hanya dapat dimiliki oleh 1 pengguna. Relasi antara table *farms* dan *bookmarks* yaitu 1 to many dimana 1 bookmarks hanya dapat menyimpan 1 data peternakan, sedangkan 1 peternakan dapat memiliki lebih dari 1 bookmarks. Selanjutnya relasi antara table *users* dan *bookmarks* yaitu *many* to *many* dimana 1 user dapat memiliki lebih dari 1 bookmarks, dan begitupun sebaliknya.

3) Implementasi (*Implementation*)

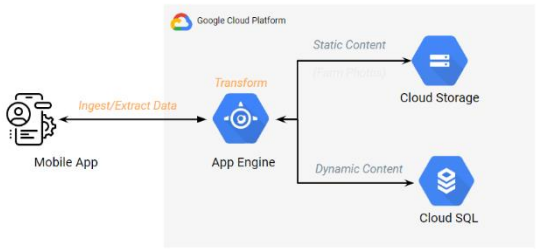
Langkah awal yang dilakukan oleh penulis yaitu merancang kebutuhan *endpoint* apa saja yang diperlukan. Proses ini dilakukan dengan berkomunikasi dengan tim *mobile developer* untuk memahami kebutuhan *endpoint* yang harus dibuat berdasarkan fitur-fitur yang ada. Setelah melakukan diskusi, diperoleh beberapa *endpoint* yang perlu dikembangkan sebagai berikut :



Gambar 8. Daftar Kebutuhan Endpoint API

Selanjutnya, penulis mempersiapkan kebutuhan *hardware* dan *software* yang diperlukan, melakukan instalasi Node.js Express dan beberapa package lain yang dibutuhkan seperti body-parser, cookie-parser, cors, dotenv, sequelize, dll.

Dalam pembuatan endpoint untuk proses *login* pengguna, penulis menggunakan JSON *Web Token* (JWT) sebagai metode autentikasi. Mekanisme JWT hampir mirip dengan kata sandi (*password*), dimana ketika proses *login* berhasil, server akan menghasilkan token yang kemudian disimpan di penyimpanan lokal atau *cookie browser* [15].



Gambar 9. Skema Arsitektur Google Cloud

Layanan GCP yang digunakan dalam proses implementasi *back-end* ini diantaranya yaitu App Engine sebagai *service deploy*, Cloud SQL (MySql) sebagai *database* relasional yang dapat diakses secara *realtime*, dan Cloud Storage sebagai tempat untuk menyimpan data statis seperti foto peternakan. Skema arsitektur layanan GCP dapat dilihat pada Gambar 9.

4) Pengujian (*Testing*)

Hasil pengujian menggunakan menggunakan black box testing :

TABEL I
PENGUJIAN UNTUK MODUL PENGGUNA

No.	API	Kondisi	Hasil	Status
1	registUser	Seluruh data terisi	Registrasi berhasil	Sukses
2		Terdapat data yang tidak diisi	Registrasi gagal	
3		Email yang digunakan belum terdaftar	Registrasi berhasil	
4		Email yang digunakan telah terdaftar	Pengguna diminta menggunakan email lain	
5	login	Email/password benar	Login berhasil, diarahkan ke menu beranda	Sukses
6		Email/password salah	Login gagal	
7		Email belum terdaftar	Muncul pop up email belum terdaftar	
8	updateData	Seluruh data terisi	Update data berhasil	Sukses
9		Pengguna menginputkan email yang telah terdaftar untuk pengguna lain	Muncul pop up email telah terdaftar. Pengguna diminta menggunakan email lain	

TABEL 2
PENGUJIAN UNTUK MODUL PETERNAKAN

No.	API	Kondisi	Hasil	Status
1	getUserFarm	Pengguna memiliki peternakan	Ditampilkan data peternakan pengguna	Sukses
2		Pengguna tidak memiliki peternakan	Diarahkan untuk membuat peternakan terlebih dahulu	
3	getAllFarms	Terdapat peternakan terdaftar	Ditampilkan data seluruh peternakan	Sukses
4		Tidak terdapat peternakan terdaftar	Data peternakan kosong	
5	getFarmDetails	Berhasil *attach* peternakan	Ditampilkan data berdasarkan peternakan yang dipilih	Sukses
6		Gagal *attach* peternakan	Gagal menampilkan data peternakan yang dipilih	
7	CreateFarm	Seluruh field data terisi dan nama peternakan belum terdaftar	Berhasil mendaftarkan peternakan	Sukses
8		Seluruh field data terisi, namun nama peternakan telah terdaftar	Muncul pop up bahwa nama peternakan telah terdaftar	
9		Seluruh field data terisi, namun tidak mengupload foto peternakan	Pengguna diminta mengupload foto peternakan	
10	UpdateFarm	Seluruh field data terisi dengan baik	Update data berhasil	Sukses
11		Nama peternakan diupdate dengan nama peternakan lain yang telah terdaftar	Muncul pop up bahwa nama peternakan telah terdaftar	
12		Pengguna menghapus foto peternakan, tetapi tidak mengupload foto terbaru	Pengguna diminta mengupload foto	

TABEL 3
PENGUJIAN UNTUK MODUL BOOKMARKS

No.	API	Kondisi	Hasil	Status
1	add Bookmark	Proses menambahkan bookmark berhasil	Peternakan ditambahkan ke daftar bookmark pengguna. Icon	Sukses

			bookmark menjadi berwarna	
2		Proses menambahkan bookmark gagal	Muncul pop up gagal menambahkan bookmark	
3	delete Bookmark	Proses hapus bookmark berhasil	Peternakan dihapus dari daftar bookmark pengguna. Icon bookmark kembali menjadi polos	Sukses
4		Proses hapus bookmark gagal	Peternakan tetap berada di daftar bookmark. Icon bookmark tetap berwarna	
5	getAllBookmarksForEachUser	Pengguna telah menambahkan peternakan ke dalam bookmark	Ditampilkan daftar peternakan yang telah dibookmark	Sukses
6		Pengguna belum menambahkan peternakan ke dalam bookmark	Data bookmark peternakan kosong	
7	checkBookmarkStatus	Peternakan telah dibookmark	isBookmarked bernilai true	Sukses
8		Peternakan telah dibookmark	isBookmarked bernilai false	

5) Deploy (*Deployment*)

Deployment dilakukan dengan menggunakan service App Engine pada GCP agar API dapat dikonsumsi secara *online* oleh *mobile development.* Hasil *deploy* dapat dilihat pada Gambar 10.

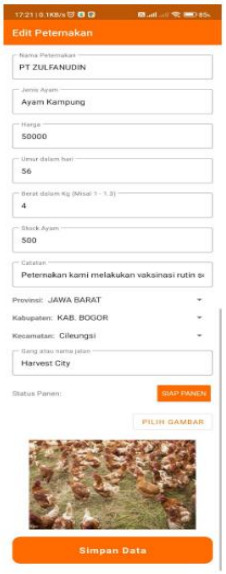
Gambar 10. Deploy pada App Engine

Selanjutnya, berikut merupakan hasil *consume* API oleh *mobile development* dimana proses pengambilan dan manipulasi data dari *database* telah berhasil dilakukan.

Gambar 10. Laman Profil

Gambar 11. Laman Beranda

Gambar 12. Laman Bookmarks



Gambar 13. Laman Edit Peternakan

Gambar 14. Laman Preview Peternakan yang Telah Dibuat

6) Pemeliharaan (*Maintenance*)

Proses maintenance dilakukan dengan memantau *credit* pada GCP untuk memastikan *credit* masih cukup untuk penggunaan service-service pada GCP. Selain itu penulis juga memantau kuota penyimpanan yang tersisa pada database Cloud SQL untuk memastikan sisa memori yang ada. Jika dirasa sisa memori kurang mencukupi, penulis dapat meningkatkan ukuran memori sesuai peningkatan kebutuhan.

IV. KESIMPULAN

Berdasarkan hasil penelitian yang dilakukan, dapat disimpulkan bahwa rancang bangun *back-end* aplikasi mobile AyamHub berhasil dibuat menggunakan framework Node.js Express. Proses pengembangan dilakukan menggunakan metode *waterfall* dengan implementasi menggunakan layanan Google *Cloud Platform* (GCP), seperti App Engine, Cloud Storage, dan Cloud SQL. Hasil pengujian menggunakan metode black box testing menunjukkan bahwa keseluruhan API berfungsi dengan baik dan berjalan sesuai harapan. Selain itu, dalam sisi *mobile development* juga telah berhasil melakukan *consume* API pada seluruh endpoint yang telah dibuat.

DAFTAR PUSTAKA

[1] S. Mufti Prasetiyo *et al.*, "OKTAL : Jurnal Ilmu Komputer dan Science PERANCANGAN BACKEND DATABASE DENGAN MYSQL PADA SISTEM MANAGEMENT ASSET," vol. 2, no. 5, 2023, [Online]. Available: https://journal.mediapublikasi.id/index.php/oktal

[2] R. Andria Siregar and N. Hidayat, "Implementasi Algoritme BLAKE2B Pada JSON Web Token Untuk Mekanisme Autentikasi Query Language," 2021. [Online]. Available: http://j-ptiik.ub.ac.id

[3] A. Triawan, A. Ramot, and Y. Siboro, "Penerapan Application Programming Interface (API) Pada Push Notification Untuk Informasi Monitoring Stok Barang Minim," vol. 11, pp. 107–114, 2021, doi: 10.36350/jbs.v11i2.

[4] I. Ahmad Faruqi *et al.*, "PERANCANGAN *BACK-END* APLIKASI RUMANTARA DENGAN GAYA ARSITEKTUR REST MENGGUNAKAN METODE ITERATIVE INCREMENTAL."

[5] R. Afriansyah, M. Sholeh, and D. Andayati, "PERANCANGAN APLIKASI PEMROGRAMAN ANTARMUKA BERBASIS WEB MENGGUNAKAN GAYA ARSITEKTUR REPRESENTASI UNTUK SISTEM PRESENSI SEKOLAH," *Jurnal SCRIPT*, vol. 9, no. 1, 2021.

[6] R. Jannah and J. Salat, "PENGEMBANGAN APLIKASI FINGERPRINT KARYAWAN PADA UNIVERSITAS JABAL GHAFUR BERBASIS ANDROID," *Jurnal Sains Riset* |, vol. 12, no. 2, p. 444, 2022, doi: 10.47647/jsr.v10i12.

[7] T. Bratakusuma, I. U. Azmi, and S. Ayuningtiyas, "Pengembangan Back End Pada Aplikasi Alat Tulis Kantor Bank Indonesia Perwakilan Purwokerto Menggunakan Nodejs Back End Development on Stationary Applications Bank Indonesia Representative Office Purwokerto Using Nodejs," 2022. [Online]. Available: https://ejournal.pnc.ac.id/index.php/senovtek

[8] Hasanuddin, H. Asgar, and B. Hartono, "RANCANG BANGUN REST API APLIKASI WESHARE SEBAGAI UPAYA MEMPERMUDAH PELAYANAN DONASI KEMANUSIAAN," *JINTEKS (Jurnal Informatika Teknologi dan Sains)*, vol. 4, no. 1, pp. 8–14, 2022, Accessed: Jun. 23, 2023. [Online]. Available: https://doi.org/10.51401/jinteks.v4i1.1474

[9] B. R. Suteja and R. Agustaf, "INTEROPERABILITAS APLIKASI BERBASIS WEB SERVICE," *JURNAL INFORMASI INTERAKTIF*, vol. 5, no. 3, pp. 106–114, 2020, Accessed: Jun. 23, 2023. [Online]. Available: https://e-journal.janabadra.ac.id/index.php/informasiinteraktif/article/view/1306

[10] A. Mubariz *et al.*, "Perancangan *Back-end* Server Menggunakan Arsitektur Rest dan Platform Node.JS (Studi Kasus: Sistem Pendaftaran Ujian Masuk Politeknik Negeri Ujung Pandang)," 2020.

[11] L. Sriwidya Lafu, "IMPLEMENTASI SISTEM PENJUALAN ONLINE BERBASIS E-COMMERCE PADA USAHA UKM IKE SUTI MENGGUNAKAN METODE WATERFALL IMPLEMENTATION OF ONLINE SALES SYSTEM BASED ON E-COMMERCE IN UKM BUSINESSES IKE SUTI USING THE WATERFALL METHOD," 2021.

[12] J. A. Falaq, R. Tulloh, and M. Iqbal, "IMPLEMENTASI JARINGAN HOTSPOT BERBAYAR BERBASIS VOUCHER MENGGUNAKAN PLATFORM GOOGLE CLOUD Implementation of A Paid Hotspot Network Based on Vouchers Using the Google *Cloud Platform*."

[13] N. W. Rahadi and C. Vikasari, "Pengujian Software Aplikasi Perawatan Barang Milik Negara Menggunakan Metode Black Box Testing Equivalence Partitions," *Infotekmesin*, vol. 11, no. 1, pp. 57–61, Jan. 2020, doi: 10.35970/infotekmesin.v11i1.124.

[14] S. Nugraha, A. B. Prasetijo, and D. Eridani, "Perancangan *Back-end* Aplikasi Reservasi Talanoa Kopi and Space Menggunakan Framework Express.js Back End Design of the Talanoa Kopi

and Space Reservation Application Using Express.js Framework," *Jurnal Teknik Komputer*, vol. 1, no. 3, pp. 126–131, 2022, doi: 10.14710/jtk.v1i3.36901.

[15] R. Gunawan and A. Rahmatulloh, "JSON Web Token (JWT) untuk Authentication pada Interoperabilitas Arsitektur berbasis RESTful Web Service," *Jurnal Edukasi dan Penelitian Informatika (JEPIN)*, vol. 5, no. 1, p. 74, Apr. 2019, doi: 10.26418/jp.v5i1.27232.