# Rancang Bangun *Back-End* Aplikasi HaePe untuk Penjadwalan Tugas Rumah Tangga Berbasis *REST API*

**Hanhan Hanafiah Solihin[1], M. Rizki Pratama Suhernawan[2], Herna Gunawan[3], Dayanni Vera Versanika[4]**

[1]Prodi Sistem Informasi Fakultas Teknik Universitas Sangga Buana, Bandung, Indonesia
[2,3,4]Prodi Sistem Informasi STMIK Bandung, Bandung, Indonesia

Surel: hanhan.hanafiah@usbypkp.ac.id[1], rizkisuhernawan@stmik-bandung.ac.id[2], herna.gunawan@stmik-bandung.ac.id[3], dayannivv@stmik-bandung.ac.id[4]

**[Dikirim: 18 Juni 2025] [Direview: 19 Juli 2025] [Diterima: 23 Juli 2025]**

**DOI: 10.58761/jurtikstmikbandung.v14.i1.181**

***ABSTRAK***

*Berdasarkan Survei Angkatan Kerja Nasional (SAKERNAS) Agustus 2020, sebanyak 69,52% pekerja perempuan yang berstatus menikah, di antaranya 34,41% bekerja lebih dari 40 jam per minggu. Perempuan menikah atau ibu rumah tangga cenderung menghadapi konflik pekerjaan dan keluarga karena tuntutan peran ganda mereka, baik di lingkungan kerja maupun domestik. Literatur menunjukkan bahwa salah satu penyebab utama konflik ini adalah keterbatasan waktu yang disebabkan oleh beban kerja profesional dan pekerjaan rumah tangga. Penelitian lain mengungkapkan bahwa konflik akan semakin tinggi apabila salah satu pasangan merasa menanggung beban pekerjaan domestik lebih besar daripada pasangannya. Merespons permasalahan tersebut, penelitian ini merancang aplikasi penjadwalan tugas bernama HaePe, yang bertujuan membantu pasangan suami-istri dan anggota keluarga lainnya dalam mendistribusikan tugas rumah secara adil, berdasarkan kebiasaan dan rutinitas masing-masing. Aplikasi ini dikembangkan pada platform Android dan didukung oleh layanan Application Programming Interface (API) yang mengelola data serta mengintegrasikannya dengan layanan pihak ketiga seperti Google. Layanan API dirancang menggunakan arsitektur Representational State Transfer (REST) untuk mempermudah implementasi dan pengelolaan kode. Pengujian dilakukan menggunakan metode black-box guna memastikan fungsionalitas layanan berjalan dengan baik. Studi ini diharapkan dapat menjadi acuan dalam pengembangan layanan API berbasis arsitektur REST untuk mendukung solusi digital dalam manajemen tugas domestik.*

***Kata kunci**: REST API, penjadwalan tugas, sistem back-end, distribusi beban kerja, aplikasi*

***ABSTRACT***

*According to the Survei Angkatan Kerja Nasional (SAKERNAS) in August 2020, 69.52% of female workers who are married, with 34.41% of them working more than 40 hours per week, are potentially exposed to work-family conflict. Married women or mothers tend to experience such conflict due to the dual demands of professional and domestic roles. Literature indicates that one of the primary factors contributing to work-family conflict is time constraints caused by both professional work and household responsibilities. Other studies have shown that this conflict intensifies when one partner feels they bear a disproportionate share of domestic work compared to their spouse. In response to this issue, this study designs a task scheduling application named HaePe, aimed at helping married couples and other family members distribute household tasks fairly based on individual routines and habits. The application is developed for the Android platform and supported by an API service that manages data and integrates with third-party services such as Google. The API is designed using the Representational State Transfer (REST) architecture to facilitate code implementation and management. Functionality testing is conducted using the black-box method to ensure that the*

*services operate as intended. This study is expected to serve as a reference for implementing REST-based API services to support digital solutions for household task management.*
***Keywords**: REST API, task scheduling, back-end system, workload distribution, application*

## 1. PENDAHULUAN

Peningkatan partisipasi perempuan dalam dunia kerja telah membawa dampak positif terhadap pertumbuhan ekonomi dan kesetaraan gender. Namun, hal ini juga menimbulkan tantangan baru, terutama bagi perempuan yang telah menikah dan memiliki tanggung jawab domestik. Berdasarkan data Survei Angkatan Kerja Nasional (SAKERNAS) Agustus 2020, sebanyak 69,52% perempuan yang bekerja berstatus menikah, dan 34,41% di antaranya bekerja lebih dari 40 jam per minggu (Anggraini dkk., 2021). Kondisi ini menunjukkan bahwa perempuan menikah menghadapi beban ganda yang berpotensi memicu konflik antara peran di dunia kerja dan peran dalam rumah tangga (Yildiz dkk., 2021).

Konflik pekerjaan dan keluarga (*work-family conflict*) merupakan kondisi di mana tuntutan peran dalam satu domain (pekerjaan atau keluarga) mengganggu pelaksanaan peran dalam domain lainnya (Shi dkk., 2023). Salah satu faktor utama yang memicu konflik ini adalah keterbatasan waktu, baik akibat pekerjaan profesional maupun tanggung jawab rumah tangga. Penelitian sebelumnya juga menunjukkan bahwa konflik ini cenderung meningkat apabila salah satu pasangan merasa memiliki beban pekerjaan domestik yang tidak seimbang dibandingkan pasangannya (Reimann dkk., 2022).

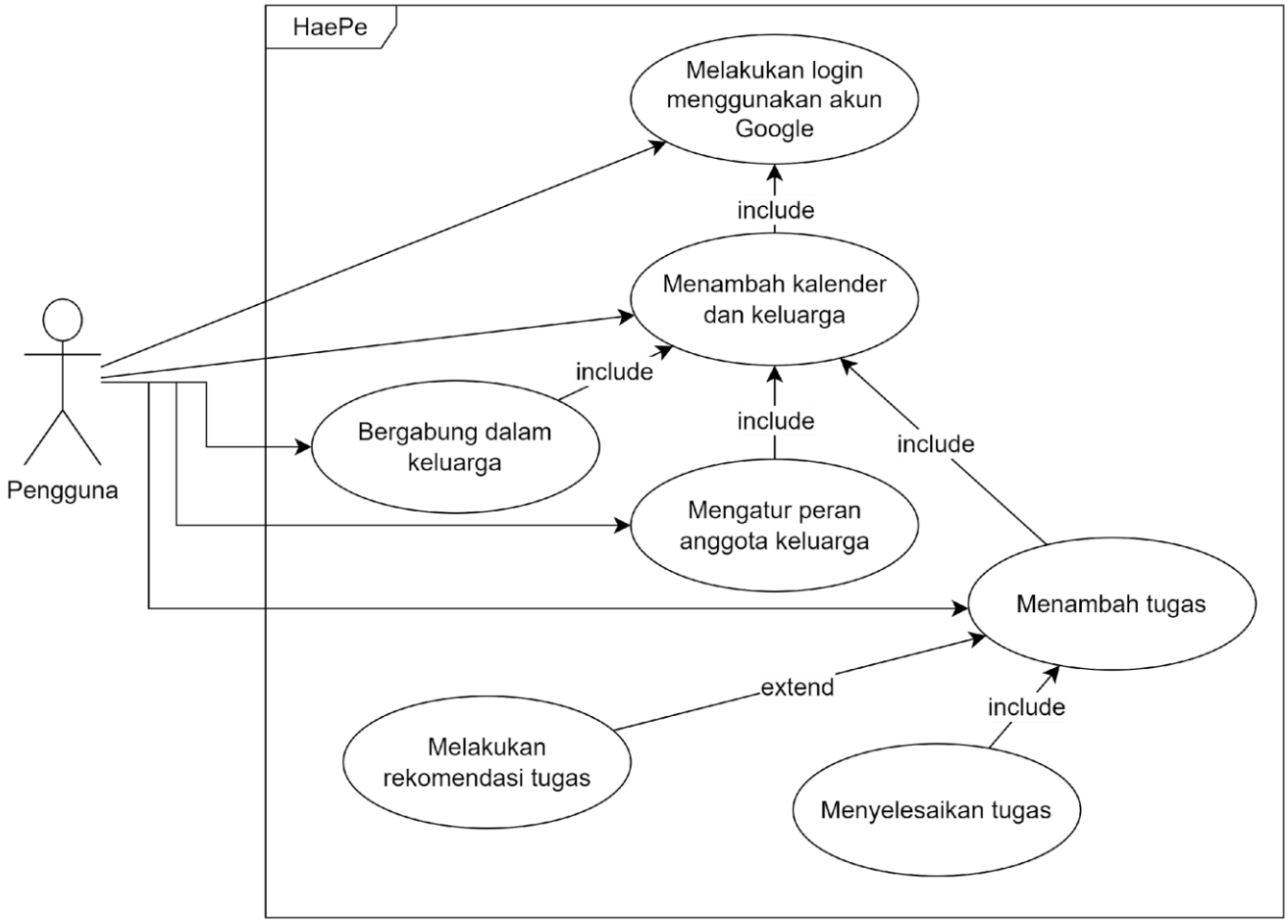
Distribusi pekerjaan rumah tangga yang tidak adil seringkali terjadi tanpa disadari, dan sulit untuk dikelola secara objektif tanpa alat bantu yang memadai (Cerrato & Cifre, 2018). Oleh karena itu, dibutuhkan solusi berbasis teknologi yang dapat membantu keluarga, khususnya pasangan suami-istri, untuk membagi tugas rumah secara lebih adil dan terstruktur. Aplikasi penjadwalan tugas rumah tangga menjadi salah satu pendekatan potensial yang dapat mengurangi ketimpangan dan konflik dengan meningkatkan kesadaran serta kolaborasi antar anggota keluarga (Lum, 2021).

Dalam penelitian ini, kami merancang dan membangun *HaePe*, sebuah aplikasi penjadwalan tugas rumah tangga berbasis mobile. Aplikasi ini dirancang untuk membantu anggota keluarga dalam mengelola pembagian tugas berdasarkan kebiasaan dan rutinitas masing-masing individu. Salah satu aspek penting dari pengembangan aplikasi ini adalah perancangan sistem back-end yang mendukung integrasi data serta konektivitas dengan layanan pihak ketiga, seperti Google Calendar .

Sistem back-end dibangun menggunakan arsitektur Representational State Transfer (REST) yang memungkinkan layanan Application Programming Interface (API) bersifat modular, ringan, dan mudah diintegrasikan (Ehsan dkk., 2022)(Hasanuddin dkk., 2022). Arsitektur ini dipilih karena sifatnya yang fleksibel dan sesuai untuk pengembangan aplikasi mobile yang membutuhkan pertukaran data secara efisien. Untuk memastikan keandalan fungsionalitas sistem, layanan API diuji menggunakan metode *black-box testing*. Metode ini memungkinkan pengujian dilakukan tanpa perlu mengetahui isian dan struktur kode karena hanya berfokus pada luaran dari suatu program (Wintana dkk., 2022)(Shaleh dkk., 2021).

Dengan merancang sistem back-end berbasis REST API untuk aplikasi *HaePe*, penelitian ini tidak hanya menawarkan solusi praktis bagi distribusi tugas rumah tangga, tetapi juga memberikan kontribusi pada praktik pengembangan layanan API yang dapat diadopsi untuk kebutuhan aplikasi serupa. Diharapkan, hasil dari studi ini dapat menjadi referensi dalam pengembangan sistem informasi berbasis keluarga yang mendukung keseimbangan kehidupan kerja dan rumah tangga.

## 2. METODOLOGI




**Gambar 1. *Use Case Diagram* aplikasi HaePe**

Metode pengembangan perangkat lunak yang digunakan dalam penelitian ini adalah Extreme Programming (XP), salah satu pendekatan dalam Agile Software Development yang mendukung perubahan kebutuhan pengguna secara dinamis (Rasheed dkk., 2021)(Guerrero-Ulloa dkk., 2023). XP cocok diterapkan dalam proyek dengan karakteristik iteratif dan kolaboratif seperti pengembangan aplikasi *HaePe*. Adapun tahapan pengembangan perangkat lunak dalam penelitian ini mengacu pada XP dan terdiri dari empat tahap berikut (Prayitno dkk., 2023):

**1) Perencanaan (*Planning*)**

Tahap ini dimulai dengan observasi terhadap aplikasi produktivitas yang tersedia di pasar, serta analisis terhadap proyek-proyek open-source yang relevan. Selain itu, dilakukan pengumpulan data kebutuhan pengguna melalui studi pustaka dan analisis peran rumah tangga dalam konteks manajemen waktu. Hasil dari tahap ini adalah dokumen kebutuhan sistem sebagai dasar perancangan.

**2) Perancangan (*Design*)**

Pada tahap ini dilakukan pemodelan sistem menggunakan pendekatan berorientasi objek. Salah satu model utama yang digunakan adalah *use case diagram* untuk mengidentifikasi kebutuhan fungsional pengguna. Selain itu, dirancang pula struktur *endpoint* REST API yang akan digunakan dalam sistem back-end.

**3) Pengkodean (*Coding*)**

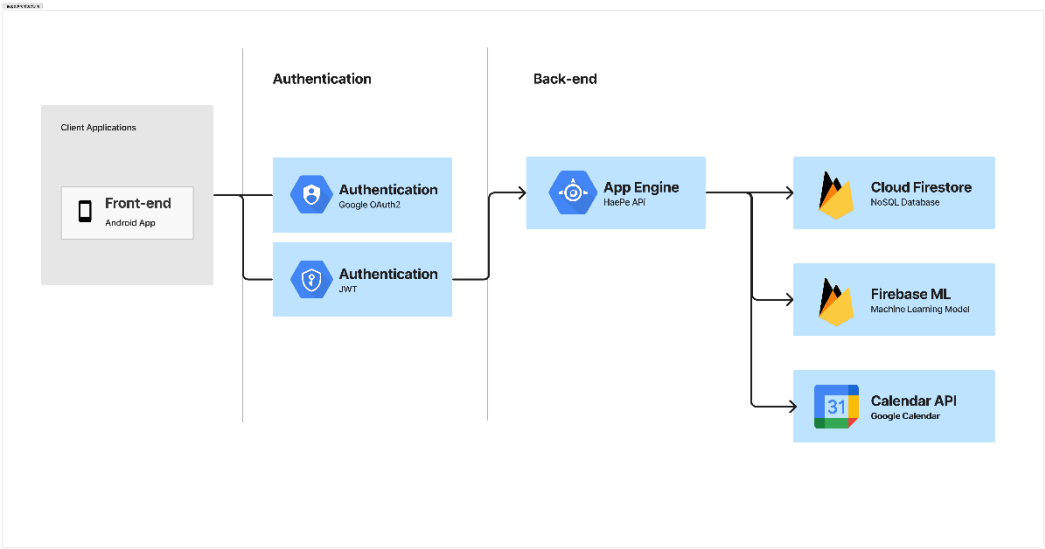
Implementasi sistem dilakukan berdasarkan hasil perancangan, khususnya pengembangan layanan REST API sebagai inti dari sistem back-end. API dikembangkan menggunakan arsitektur REST untuk memastikan

keterpisahan antarmuka klien dan server serta kemudahan integrasi dengan layanan pihak ketiga seperti Google Calendar.

**4) Pengujian (*Testing*)**

| | Peran | Umur | Minat | Minat1 | Minat2 | Keahlian | Keahlian1 | Aktivitas |
|---|---|---|---|---|---|---|---|---|
| 273 | Ibu | 21-55 | Makanan | Keuangan | Kesehatan | Masak | Berkebun | Membersihkan Rumah |
| 274 | Anak | 15-20 | Makanan | Hiburan | Edukasi | Masak | Sosialisasi | Memasak |
| 275 | Ibu | >55 | Alam | Hiburan | Edukasi | Masak | Sosialisasi | Memasak |
| 277 | Anak | 15-20 | Makanan | Teknologi | Hiburan | Masak | Sosialisasi | Mencuci Piring |
| 278 | Ayah | 21-55 | Olahraga | Alam | Edukasi | Kelistrikan | Berkebun | Memperbaiki Rumah/Kelistrikan/Perabotan |
| 279 | Ibu | 21-55 | Makanan | Edukasi | Keuangan | Masak | Manajemen | Belanja Kebutuhan |
| 281 | Ibu | 21-55 | Olahraga | Alam | Kesehatan | Masak | Manajemen | Memasak |
| 284 | Ibu | 21-55 | Makanan | Hiburan | Keuangan | Masak | Berkebun | Memasak |
| 285 | Ayah | 21-55 | Teknologi | Hiburan | Edukasi | Otomotif | Kelistrikan | Memperbaiki Rumah/Kelistrikan/Perabotan |
| 286 | Anak | 15< | Makanan | Teknologi | Hiburan | Masak | Manajemen | Memasak |

**Gambar 2. Contoh 10 data yang didapatkan dari kuesioner**




**Gambar 3. Arsitektur cloud aplikasi HaePe**

Pengujian dilakukan menggunakan metode *black-box testing* yang fokus pada pengujian fungsionalitas dari setiap *endpoint* API tanpa memeriksa struktur internal kode. Pengujian mencakup validasi input dan output serta pengujian respons sistem terhadap berbagai skenario penggunaan.

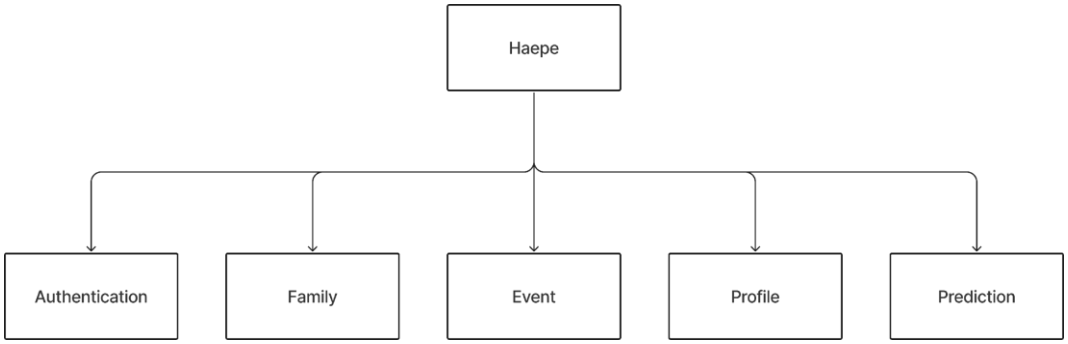
## 3. HASIL DAN PEMBAHASAN

**1) Perencanaan (*Planning*)**

Observasi dilakukan terhadap aplikasi produktivitas populer seperti Google Calendar dan beberapa proyek open-source untuk mengidentifikasi kebutuhan pengguna. Hasil observasi menunjukkan bahwa sistem harus mendukung autentikasi dua arah menggunakan Google Authentication, manajemen data pengguna berbasis peran keluarga, pengelolaan hak akses tugas berdasarkan peran, serta distribusi tugas yang dapat ditargetkan kepada anggota keluarga tertentu. Untuk mendukung fitur rekomendasi tugas, dikembangkan dataset dari 222 responden valid yang diperoleh melalui kuesioner, mencakup data peran, umur, minat, keahlian, dan aktivitas rumah tangga yang sering dilakukan.

**2) Perancangan (*Design*)**

Berdasarkan kebutuhan pengguna yang telah diidentifikasi pada tahap perencanaan, dilakukan perancangan sistem dengan pendekatan pemodelan perangkat lunak berbasis objek. Salah satu artefak utama yang dikembangkan adalah *use case diagram*, yang berfungsi sebagai acuan implementasi fungsionalitas aplikasi (*lihat Gambar 1*).




**Gambar 4. Peta *endpoint* dari API aplikasi HaePe**

Diagram ini menggambarkan hubungan antara aktor seperti orang tua, anak, dan anggota keluarga lainnya dengan fitur-fitur sistem, seperti pengelolaan akun, pembagian tugas, serta integrasi dengan layanan eksternal.

Selanjutnya, dilakukan analisis terhadap data kuesioner yang diperoleh dari 222 responden valid (*lihat Gambar 2*). Analisis ini menggunakan pendekatan regresi linear untuk mengetahui arah serta kekuatan pengaruh dari variabel independen terhadap variabel dependen (Nurani dkk., 2023). Variabel independen yang digunakan meliputi peran dalam keluarga, umur, tiga minat utama, dan dua keahlian yang dimiliki masing-masing individu. Adapun variabel dependen dalam model ini adalah jenis aktivitas domestik yang dilakukan.

Model regresi linear ini digunakan untuk mendukung sistem rekomendasi tugas rumah tangga yang menjadi fitur inti aplikasi. Hasil pelatihan model menunjukkan nilai kesalahan rata-rata mutlak (*mean absolute error*) sebesar 1.0868 dan nilai kesalahan rata-rata kuadrat (*mean squared error*) sebesar 2.1309. Nilai MAE digunakan sebagai tolok ukur utama mengingat jumlah kategori aktivitas yang didefinisikan dalam sistem terbatas pada tujuh jenis, sehingga sistem dapat merekomendasikan hingga dua deviasi tugas terdekat dari hasil prediksi. Dengan demikian, sistem akan menghasilkan tiga opsi tugas rumah tangga yang disarankan kepada pengguna.

Perancangan sistem juga mencakup definisi struktur layanan back-end menggunakan pendekatan arsitektur REST. Setiap *endpoint* dalam layanan API dirancang untuk mengelola komponen data secara modular, termasuk autentikasi pengguna, pengaturan profil, penyimpanan data tugas, serta komunikasi dengan Google Calendar. Pendekatan ini memastikan fleksibilitas dan keterpisahan antara sisi klien dan server, serta mendukung integrasi dengan layanan eksternal secara efisien.

**3) Pengkodean (*Coding*)**

Dalam tahap implementasi, sistem dibangun dengan pendekatan layanan REST API yang dijalankan di atas platform Google Cloud, khususnya menggunakan Google App Engine. Layanan API ini menjadi pusat komunikasi antara aplikasi *client* dan sumber data eksternal, serta mendukung integrasi dengan Google Calendar untuk sinkronisasi penjadwalan tugas antar anggota keluarga.

Sistem autentikasi menerapkan OAuth 2.0 (Hardt, 2012) melalui Google Authentication yang dikombinasikan dengan JSON Web Token (JWT) sebagai metode otorisasi. JWT digunakan untuk memastikan setiap permintaan dari *client* memiliki kredensial yang sah dan dapat divalidasi pada setiap transaksi API. Autentikasi dua arah ini memungkinkan pengguna melakukan sinkronisasi tugas ke Google Calendar secara aman (*lihat Gambar 3*).

Dalam pengembangannya, lima *endpoint* utama disusun dan diimplementasikan sebagai berikut (*lihat Gambar 4*):

a. *Authentication* (/auth)
   Mengelola proses autentikasi pengguna melalui protokol OAuth 2.0 dengan menggunakan layanan Google Authentication. *Endpoint* ini menghasilkan *access token* yang digunakan untuk mengidentifikasi dan mengotorisasi pengguna pada transaksi berikutnya.
b. *Family* (/family)
   Mengelola data keluarga dan proses sinkronisasi tugas dengan akun Google Calendar pengguna. Setelah pengguna memberikan izin integrasi, sistem secara otomatis membuat entri tugas dalam kalender sesuai dengan waktu dan deskripsi yang telah ditentukan.
c. *Event* (/event)
   Berfungsi untuk menambahkan, memperbarui, dan mengambil data tugas rumah tangga. Tugas-tugas ini ditujukan kepada anggota keluarga tertentu, baik berdasarkan hasil rekomendasi sistem maupun preferensi manual pengguna.
d. *Profile* (/profile)
   Mengelola informasi profil pengguna, termasuk nama, tanggal lahir, umur, peran dalam keluarga, minat, dan keahlian. Data yang tersimpan melalui *endpoint* ini menjadi input utama dalam sistem rekomendasi aktivitas berbasis pembelajaran mesin.
e. *Prediction* (/predict)
   Menerima data profil pengguna dan mengirimkannya ke model pembelajaran mesin untuk menghasilkan tiga rekomendasi aktivitas rumah tangga yang relevan. Hasil rekomendasi ini ditampilkan kepada pengguna sebagai referensi dalam pembagian tugas.

**4) Pengujian (*Testing*)**

Setelah proses implementasi selesai, dilakukan tahap pengujian sistem untuk memastikan bahwa setiap layanan API berjalan sesuai dengan spesifikasi yang telah dirancang. Pengujian dilakukan menggunakan metode *black box testing*, yaitu metode yang berfokus pada pengujian masukan (*input*) dan keluaran (*output*) sistem tanpa memperhatikan struktur internal atau kode sumber sistem (Wintana dkk., 2022).

Pengujian difokuskan pada *endpoint-endpoint* utama yang telah dibangun, meliputi /auth, /family, /event, /profile, dan /predict. Pengujian ini bertujuan untuk memverifikasi validitas data yang diterima sistem, integritas data yang diproses, dan kesesuaian respons sistem terhadap skenario penggunaan yang umum maupun ekstrem.

Untuk mendukung proses ini, disusun sebanyak 36 skenario pengujian yang mencakup berbagai kasus seperti:

- Autentikasi berhasil dan gagal;
- Penyimpanan dan pembaruan data profil pengguna;
- Penjadwalan tugas baru dan pengambilan daftar tugas;
- Sinkronisasi dengan Google Calendar;
- Permintaan rekomendasi aktivitas berdasarkan variasi data profil.

Setiap skenario pengujian memiliki keluaran yang diharapkan (*expected output*) yang dibandingkan dengan keluaran aktual (*actual output*) dari sistem. Hasil pengujian menunjukkan bahwa seluruh *endpoint* berfungsi sebagaimana mestinya, dengan keluaran yang sesuai terhadap semua masukan valid, serta penanganan kesalahan yang tepat untuk masukan yang tidak valid. Rincian hasil pengujian dan tabel pengujian untuk setiap *endpoint* disajikan pada Lampiran A.1.

Tahap ini menjadi penting untuk menjamin kualitas dan reliabilitas sistem sebelum dilakukan pengujian lebih lanjut pada sisi aplikasi *client* berbasis Android, yang berinteraksi langsung dengan layanan REST API ini.

## 4. KESIMPULAN DAN SARAN

Penelitian ini berhasil merancang dan mengimplementasikan sistem back-end untuk aplikasi penjadwalan tugas rumah tangga bernama HaePe dengan memanfaatkan pendekatan REST API. Sistem dibangun menggunakan metode pengembangan perangkat lunak *Extreme Programming* (XP) yang terdiri dari tahapan perencanaan, perancangan, pengkodean, dan pengujian.

Lima *endpoint* utama berhasil dikembangkan, yaitu /auth untuk autentikasi pengguna, /profile untuk manajemen data pengguna, /event untuk pengelolaan tugas rumah tangga, /family untuk integrasi dengan Google Calendar, dan /predict untuk menghasilkan rekomendasi aktivitas berbasis model machine learning. Layanan REST API ini diimplementasikan menggunakan Google Cloud Platform, dengan sistem autentikasi dua arah yang menggabungkan Google Authentication dan JSON Web Token (JWT). Hasil pengujian black box menunjukkan bahwa seluruh *endpoint* bekerja dengan baik sesuai skenario pengujian yang disiapkan.

Implementasi sistem ini memberikan solusi terintegrasi dalam pendistribusian tugas rumah tangga berbasis teknologi, sekaligus menjadi contoh penerapan REST API yang baik dalam konteks manajemen keluarga digital.

Beberapa hal yang disarankan untuk pengembangan sistem ke depan antara lain:

1. Penguatan mekanisme keamanan: Meskipun JWT telah diimplementasikan, perlu ditambahkan perlindungan terhadap serangan umum seperti *token hijacking*, *rate limiting*, dan penggunaan HTTPS secara menyeluruh untuk memperkuat keamanan komunikasi data.
2. Skalabilitas sistem: Perlu dilakukan pengujian *load testing* dan *stress testing* untuk memastikan sistem tetap responsif saat digunakan oleh banyak pengguna secara bersamaan.
3. Optimalisasi model rekomendasi: Model regresi linear dapat ditingkatkan dengan mengevaluasi algoritma lain seperti Random Forest atau klasifikasi berbasis neural network untuk meningkatkan akurasi saran aktivitas.
4. Evaluasi pengalaman pengguna (UX): Uji coba langsung dengan pengguna akhir (*user acceptance testing*) dapat memberikan masukan untuk menyempurnakan fungsionalitas dan antarmuka aplikasi.
5. Ekspansi fitur kolaboratif: Dapat dipertimbangkan penambahan fitur seperti pengingat otomatis, pelacakan penyelesaian tugas, dan laporan aktivitas keluarga untuk mendukung kolaborasi dalam rumah tangga secara lebih menyeluruh.

## ACKNOWLEDGMENT

Penulis mengucapkan terima kasih kepada program Studi Independen Bangkit 2022 yang telah memberikan dukungan dan fasilitas, khususnya akses layanan Google Cloud Platform, yang sangat membantu dalam proses pengembangan dan implementasi sistem *back end* aplikasi HaePe. Fasilitas ini memungkinkan pengujian dan penyebaran layanan REST API dilakukan secara optimal dalam lingkungan cloud yang andal dan skalabel. Ucapan terima kasih juga disampaikan kepada seluruh pihak yang telah memberikan dukungan moral dan teknis selama proses penelitian dan pengembangan sistem ini berlangsung.

## DAFTAR PUSTAKA

Anggraini, S., Nurhayati, Lukitasari, I., Bodromurti, W., & Surida, D. (2021). Profil Perempuan Indonesia Tahun 2021. Dalam *Kementrian Pemberdayaan Perempuan dan Perlindungan Anak*. https://www.kemenpppa.go.id/page/view/MzgxNA==

Cerrato, J., & Cifre, E. (2018). Gender inequality in household chores and work-family conflict. *Frontiers in Psychology*, *9*(AUG). https://doi.org/10.3389/fpsyg.2018.01330

Ehsan, A., Abuhaliqa, M. A. M. E., Catal, C., & Mishra, D. (2022). RESTful API Testing Methodologies: Rationale, Challenges, and Solution Directions. Dalam *Applied Sciences (Switzerland)* (Vol. 12, Nomor 9). MDPI. https://doi.org/10.3390/app12094369

Guerrero-Ulloa, G., Rodríguez-Domínguez, C., & Hornos, M. J. (2023). Agile Methodologies Applied to the Development of Internet of Things (IoT)-Based Systems: A Review. Dalam *Sensors* (Vol. 23, Nomor 2). MDPI. https://doi.org/10.3390/s23020790

Hardt, D. (2012). *The OAuth 2.0 authorization framework*. RFC 6749.

Hasanuddin, Asgar, H., & Hartono, B. (2022). RANCANG BANGUN REST API APLIKASI WESHARE SEBAGAI UPAYA MEMPERMUDAH PELAYANAN DONASI KEMANUSIAAN. *Jurnal Informatika Teknologi dan Sains*, *4*(1), 8–14. https://doi.org/10.51401/jinteks.v4i1.1474

Lum, M. (2021). *Divvy - a better way to divvy up chores*. Pearl Hacks 2021. https://devpost.com/software/divvy-diu20k

Nurani, A. T., Setiawan, A., & Susanto, B. (2023). Perbandingan Kinerja Regresi Decision Tree dan Regresi Linear Berganda untuk Prediksi BMI pada Dataset Asthma. *Jurnal Sains dan Edukasi Sains*, *6*(1), 34–43. https://doi.org/10.24246/juses.v6i1p34-43

Prayitno, E., Siregar, J., Bahri, C., Sariasih Ayu, F., & Armelsa, D. (2023). Perancangan Sistem Informasi Penerimaan Peserta Didik Baru (PPDB) Berbasis Web Menggunakan Extreme Programming (XP). *Smart Comp: Jurnalnya Orang Pintar Komputer*, *12*(1). https://doi.org/10.30591/smartcomp.v12i1.4781

Rasheed, A., Zafar, B., Shehryar, T., Aslam, N. A., Sajid, M., Ali, N., Dar, S. H., & Khalid, S. (2021). Requirement Engineering Challenges in Agile Software Development. Dalam *Mathematical Problems in Engineering* (Vol. 2021). Hindawi Limited. https://doi.org/10.1155/2021/6696695

Reimann, M., Schulz, F., Marx, C. K., & Lükemann, L. (2022). The family side of work-family conflict: A literature review of antecedents and consequences. *Journal of Family Research*, *34*(4), 1010–1032. https://doi.org/10.20377/jfr-859

Shaleh, I. A., Yogi, J. P., Pirdaus, P., Syawal, R., & Saifudin, A. (2021). Pengujian Black Box pada Sistem Informasi Penjualan Buku Berbasis Web dengan Teknik Equivalent Partitions. *Jurnal Teknologi Sistem Informasi dan Aplikasi*, *4*(1), 38. https://doi.org/10.32493/jtsi.v4i1.8960

Shi, S., Chen, Y., & Cheung, C. M. K. (2023). How technostressors influence job and family satisfaction: Exploring the role of work–family conflict. *Information Systems Journal*, *33*(4), 953–985. https://doi.org/10.1111/isj.12431

Wintana, D., Pribadi, D., & Nurhadi, M. Y. (2022). Analisis Perbandingan Efektifitas White-Box Testing dan Black-Box Testing. *Jurnal Larik: Ladang Artikel Ilmu Komputer*, *2*(1), 8–16. https://doi.org/10.31294/larik.v2i1.1382

Yildiz, B., Yildiz, H., & Ayaz Arda, O. (2021). Relationship between work–family conflict and turnover intention in nurses: A meta-analytic review. Dalam *Journal of Advanced Nursing* (Vol. 77, Nomor 8, hlm. 3317–3330). Blackwell Publishing Ltd. https://doi.org/10.1111/jan.14846

## LAMPIRAN A

**Tabel A.1 Pengujian *black box* pada layanan API aplikasi HaePe**

| No | Skenario Pengujian | Hasil yang Diharapkan | Kesimpulan |
|---|---|---|---|
| 1. | Memanggil *endpoint* **Login** dengan idToken yang valid. | Sistem akan merespon dengan data JSON berisi {statusCode: 200, status: “success”, message: “Login Berhasil.”, data: AuthToken}. | Valid |
| 2. | Memanggil *endpoint* **Login** dengan idToken yang tidak valid. | Sistem akan merespon dengan data JSON berisi {statusCode: 500, status: “fail”, message: “Token validation error: ID Token tidak sesuai / ID *token* sudah kadaluarsa.”} | Valid |
| 3. | Memanggil *endpoint* **Family Insert** ketika pengguna belum tergabung dalam keluarga. | Sistem akan merespon dengan data JSON berisi {statusCode: 201, status: “success”, message: “Berhasil menambahkan keluarga.”, data: familyId} | Valid |
| 4. | Memanggil *endpoint* **Family Insert** ketika pengguna tergabung dalam keluarga. | Sistem akan merespon dengan data JSON berisi {statusCode: 400, status: “fail”, message: “User sudah memiliki keluarga.”} | Valid |
| 5. | Memanggil *endpoint* **Family Get** ketika tergabung dalam keluarga. | Sistem akan merespon dengan data JSON berisi {statusCode: 200, status: “success”, message: “Data keluarga ditemukan.”, data: FamilyDataCollection} | Valid |

| | | | |
|---|---|---|---|
| 6. | Memanggil *endpoint* **Family Get** ketika belum tergabung dalam keluarga. | Sistem akan merespon dengan data JSON berisi {statusCode: 404, status: “fail”, message: “User tidak memiliki data keluarga.”} | Valid |
| 7. | Memanggil *endpoint* **Family Update** dengan *request body* berisi {name: String}, dan *path variable* /:id dimana :id adalah familyId yang valid. | Sistem akan merespon dengan data JSON berisi {statusCode: 200, status: “success”, message: “Data keluarga berhasil diperbaharui.”, data: familyId, name} | Valid |
| 8. | Memanggil *endpoint* **Family Update** dengan *request body* berisi {name: String}, dan *path variable* /:id dimana :id adalah familyId yang tidak valid. | Sistem akan merespon dengan data JSON berisi {statusCode: 404, status: “fail”, message: “Data keluarga tidak ditemukan.”} | Valid |
| 9. | Memanggil *endpoint* **Family Update** dengan *request body* berisi {name: String}, dan *path variable* /:id dimana :id adalah familyId yang valid tapi pengguna tidak memiliki *role owner*. | Sistem akan merespon dengan data JSON berisi {statusCode: 403, status: “fail”, message: “Tidak memiliki otoritas untuk melakukan aksi ini.”} | Valid |
| 10. | Memanggil *endpoint* **Family Join** dengan *request body* berisi {id: String dimana id adalah familyId yang valid dan pengguna belum bergabung ke dalam keluarga. | Sistem akan merespon dengan data JSON berisi {statusCode: 200, status: “success”, message: “User berhasil ditambahkan ke dalam keluarga.”, data: id} | Valid |
| 11. | Memanggil *endpoint* **Family Join** dengan *request body* berisi {id: String dimana id adalah familyId yang valid dan pengguna sudah bergabung ke dalam keluarga. | Sistem akan merespon dengan data JSON berisi {statusCode: 400, status: “fail”, message: “User sudah bergabung ke dalam keluarga.”} | Valid |
| 12. | Memanggil *endpoint* **Family Join** dengan *request body* berisi {id: String dimana id adalah familyId yang tidak valid. | Sistem akan merespon dengan data JSON berisi {statusCode: 404, status: “fail”, message: “Id keluarga tidak ditemukan.”} | Valid |
| 13. | Memanggil *endpoint* **Family Leave** dengan *path variable* berisi /:id dimana :id adalah familyId yang valid dan memiliki minimal satu member dengan role owner selain dari pengguna di dalam keluarga. | Sistem akan merespon dengan data JSON berisi {statusCode: 200, status: “success”, message: “User berhasil meninggalkan grup keluarga.”} | Valid |
| 14. | Memanggil *endpoint* **Family Leave** dengan *path variable* berisi /:id dimana :id adalah familyId yang valid dan tidak memiliki minimal satu member dengan role owner selain dari pengguna di dalam keluarga. | Sistem akan merespon dengan data JSON berisi {statusCode: 400, status: “fail”, message: “Tetapkan role owner pada member lain terlebih dahulu.”} | Valid |
| 15. | Memanggil *endpoint* **Family Role Update** dengan *path variable* berisi /:id dimana :id adalah familyId yang valid dengan *request body* berisi {userId: String, role: String} dan pengguna memiliki *role owner*. | Sistem akan merespon dengan data JSON berisi {statusCode: 200, status: “success”, message: “Berhasil mengubah role.”, data: id, role} | Valid |
| 16. | Memanggil *endpoint* **Family Role Update** dengan *path variable* berisi /:id dimana :id adalah familyId yang tidak valid dengan *request body* berisi {userId: String, role: String} dan pengguna memiliki *role owner*. | Sistem akan merespon dengan data JSON berisi {statusCode: 404, status: “fail”, message: “Data keluarga tidak ditemukan.”} | Valid |
| 17. | Memanggil *endpoint* **Family Role Update** dengan *path variable* berisi /:id dimana :id adalah familyId yang valid dengan *request body* berisi {userId: String, role: String} dan pengguna memiliki *role owner* tapi mengubah satu-satunya member dengan *role owner*. | Sistem akan merespon dengan data JSON berisi {statusCode: 400, status: “fail”, message: “Tetapkan role owner pada member lain terlebih dahulu.”} | Valid |
| 18. | Memanggil *endpoint* **Family Delete** dengan *path variable* berisi /:id dimana :id adalah familyId yang valid dan pengguna memiliki *role owner*. | Sistem akan merespon dengan data JSON berisi {statusCode: 200, status: “success”, message: “Data keluarga berhasil dihapus.” } | Valid |
| 19. | Memanggil *endpoint* **Family Delete** dengan *path variable* berisi /:id dimana :id adalah familyId yang tidak valid dan pengguna memiliki *role owner*. | Sistem akan merespon dengan data JSON berisi {statusCode: 400, status: “fail”, message: “Data keluarga tidak ditemukan.” } | Valid |

| | | | |
|---|---|---|---|
| 20. | Memanggil *endpoint* **Family Delete** dengan *path variable* berisi /:id dimana :id adalah familyId yang valid dan pengguna tidak memiliki *role owner*. | Sistem akan merespon dengan data JSON berisi {statusCode: 404, status: “fail”, message: “Tidak memiliki otoritas untuk melakukan aksi ini.” } | Valid |
| 21. | Memanggil *endpoint* **Event Get**. | Sistem akan merespon dengan data JSON berisi {statusCode: 200, status: “success”, message: “Berhasil mengambil data event.”, data: EventCollectionData} | Valid |
| 22. | Memanggil *endpoint* **Event Insert** dengan *request body* berisi {start: Timestamp, end: Timestamp, summary: String, descripstion: String, userId: String} dimana userId adalah userId yang valid dan tergabung di dalam keluarga. | Sistem akan merespon dengan data JSON berisi {statusCode: 201, status: “success”, message: “Berhasil menambahkan event.”, data: id, creator, start, end, summary, description, assignFor, eventId} | Valid |
| 23. | Memanggil *endpoint* **Event Insert** dengan *request body* berisi {start: Timestamp, end: Timestamp, summary: String, descripstion: String, userId: String} dimana userId adalah userId yang valid dan tidak tergabung di dalam keluarga. | Sistem akan merespon dengan data JSON berisi {statusCode: 404, status: “fail”, message: “User id yang dimasukkan tidak terdaftar dalam keluarga anda.”} | Valid |
| 24. | Memanggil *endpoint* **Event Insert** dengan *request body* berisi {start: Timestamp, end: Timestamp, summary: String, descripstion: String, userId: String} dimana userId adalah userId yang valid dan tergabung di dalam keluarga tapi sudah memiliki jadwal. | Sistem akan merespon dengan data JSON berisi {statusCode: 400, status: “fail”, message: “Sudah ada jadwal.”} | Valid |
| 25. | Memanggil *endpoint* **Event Delete** dengan *path variable* /:id dimana :id adalah eventId yang valid. | Sistem akan merespon dengan data JSON berisi {statusCode: 200, status: “success”, message: “Data event berhasil dihapus.”} | Valid |
| 26. | Memanggil *endpoint* **Event Delete** dengan *path variable* /:id dimana :id adalah eventId yang tidak valid. | Sistem akan merespon dengan data JSON berisi {statusCode: 404, status: “fail”, message: “Data event tidak ditemukan.”} | Valid |
| 27. | Memanggil *endpoint* **Event Update** dengan *path variable* /:id dimana :id adalah eventId yang valid dengan *request body* berisi {start, end, summary, description, userId} dimana pengguna adalah pembuat event atau memiliki *role owner*. | Sistem akan merespon dengan data JSON berisi {statusCode: 200, status: “success”, message: “Berhasil mengubah event.”, data: creator, start, end, summary, description, assignFor, eventId}. | Valid |
| 28. | Memanggil *endpoint* **Event Update** dengan *path variable* /:id dimana :id adalah eventId yang valid dengan *request body* berisi {start, end, summary, description, userId} dimana pengguna adalah pembuat event atau memiliki *role owner* tapi event sudah selesai dilakukan. | Sistem akan merespon dengan data JSON berisi {statusCode: 400, status: “fail”, message: “Event sudah selesai.”}. | Valid |
| 29. | Memanggil *endpoint* **Event Update** dengan *path variable* /:id dimana :id adalah eventId yang valid dengan *request body* berisi {start, end, summary, description, userId} dimana pengguna adalah pembuat event atau memiliki *role owner* tapi sudah ada jadwal. | Sistem akan merespon dengan data JSON berisi {statusCode: 400, status: “fail”, message: “Sudah ada jadwal.”}. | Valid |
| 30. | Memanggil *endpoint* **Event Update** dengan *path variable* /:id dimana :id adalah eventId yang tidak valid dengan *request body* berisi {start, end, summary, description, userId} dimana pengguna adalah pembuat event. | Sistem akan merespon dengan data JSON berisi {statusCode: 404, status: “fail”, message: “Event tidak ditemukan.”}. | Valid |
| 31. | Memanggil *endpoint* **Event Todo Done** dengan *path variable* /:id dimana :id adalah eventId yang valid dimana pengguna adalah petugas event. | Sistem akan merespon dengan data JSON berisi {statusCode: 200, status: “success”, message: “Event sudah diselesaikan.”, data: creator, start, end, summary, description, assignFor, eventId}. | Valid |

| | | | |
|---|---|---|---|
| 32. | Memanggil *endpoint* **Event Todo Done** dengan *path variable* /:id dimana :id adalah eventId yang tidak valid dimana pengguna adalah petugas event. | Sistem akan merespon dengan data JSON berisi {statusCode: 404, status: “fail”, message: “Event tidak ditemukan.”}. | Valid |
| 33. | Memanggil *endpoint* **Event Todo Done** dengan *path variable* /:id dimana :id adalah eventId yang valid dimana pengguna adalah bukan petugas event. | Sistem akan merespon dengan data JSON berisi {statusCode: 403, status: “fail”, message: “Event ini tidak ditugaskan kepada Anda.”}. | Valid |
| 34. | Memanggil *endpoint* **Profile Get**. | Sistem akan merespon dengan data JSON berisi {statusCode: 200, status: “success”, message: “Berhasil mengambil data event.”, data: id, familyId, name, email, skills, role, birthday, interests, age, interestsList, skillsList}. | Valid |
| 35. | Memanggil *endpoint* **Profile Update** dengan *request body* berisi {name: String, birthday: date, role: String, interests: ArrayOfString, skills: ArrayOfString}. | Sistem akan merespon dengan data JSON berisi {statusCode: 200, status: “success”, message: “Berhasil mengubah profil.”, data: name, skills, role, birthday, interests}. | Valid |
| 36. | Memanggil *endpoint* **Submit Prediction** dengan *request body* berisi {role: String, age: String, interests: ArrayOfString, skills: ArrayOfString}. | Sistem akan merespon dengan data JSON berisi {statusCode: 200, status: “success”, message: ArrayOfString}. | Valid |