IMPLEMENTASI *IN-MEMORY DATABASE* (REDIS) UNTUK MEMPERCEPAT OPERASI RELASIONAL DALAM MENAMPILKAN REKAP DATA PADA *DASHBOARD* SISTEM DASAWISMA PKK PROVINSI SUMATERA SELATAN

LAPORAN PENELITIAN

DWI ROBBI PRASETYO

191420064

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS SAINS TEKNOLOGI

UNIVERSITAS BINA DARMA

PALEMBANG

2024

IMPLEMENTASI *IN-MEMORY DATABASE* (REDIS) UNTUK MEMPERCEPAT OPERASI RELASIONAL DALAM MENAMPILKAN REKAP DATA PADA *DASHBOARD* SISTEM DASAWISMA PKK PROVINSI SUMATERA SELATAN

DWI ROBBI PRASETYO

191420064

Laporan Penelitian ini diajukan sebagai syarat memperoleh gelar Sarjana Komputer

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS SAINS TEKNOLOGI

UNIVERSITAS BINA DARMA

PALEMBANG

2024

HALAMAN PENGESAHAN

IMPLEMENTASI *IN-MEMORY DATABASE* (REDIS) UNTUK MEMPERCEPAT OPERASI RELASIONAL DALAM MENAMPILKAN REKAP DATA PADA *DASHBOARD* SISTEM DASAWISMA PKK PROVINSI SUMATERA SELATAN

DWI ROBBI PRASETYO
191420064

Telah diterima sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer pada Program Studi Teknik Informatika

Palembang, 23 Agustus 2024
Program Studi Teknik Informatika
Fakultas Sains Teknologi
Universitas Bina Darma

Pembimbing	Dekan,
Fatoni, M. M., M.Kom	**Dr. Tata Sutabri, S.Kom., MMSI., MKM**

HALAMAN PERSETUJUAN

Skripsi Berjudul "Implementasi *In-Memory Database* (*Redis*) Untuk Mempercepat Operasi Relasional Dalam Menampilkan Rekap Data Pada *Dashboard* Sistem Dasawisma Pkk Provinsi Sumatera Selatan" Oleh "Dwi Robbi Prasetyo" telah dipertahankan di depan komisi penguji pada hari Jumat tanggal 23 Agustus 2024.

Komisi Penguji

1. Ketua : **Fatoni, M.M., M.Kom** (.....)
2. Anggota : **Muhammad Nasir, M.M., M.Kom** (.....)
3. Anggota : **Irman Effendy, M. Kom** (.....)

Mengetahui,
Program Studi Teknik Informatika
Fakultas Sains Teknologi
Universitas Bina Darma
Ketua,

Alek Wijaya, S.Kom., M.I.T.

SURAT PERNYATAAN

Saya yang bertanda tangan di bawah ini:

Nama : Dwi Robbi Prasetyo

NIM : 191420064

Dengan ini menyatakan bahwa:

1. Karya tulis saya adalah asli dan belum pernah diajukan untuk mendapatkan gelar akademik (Sarjana) di Universitas Bina Darma atau perguruan tinggi lainnya;
2. Karya tulis ini murni gagasan, rumusan dan penelitian saya dengan arahan dari tim pembimbing;
3. Di dalam karya tulis ini tidak terdapat karya atau pendapat yang telah ditulis atau di publikasikan orang lain, kecuali secara tertulis dengan jelas dikutip dengan mencantumkan nama pengarang dan memasukkan ke dalam daftar rujukan;
4. Saya bersedia karya tulis ini di cek keasliannya menggunakan *plagiarism checker* serta di unggah ke internet, sehingga dapat diakses secara daring;
5. Surat pernyataan ini saya tulis dengan sungguh-sungguh dan apabila terbukti melakukan penyimpangan atau ketidakbenaran dalam pernyataan ini maka saya bersedia menerima sanksi dengan peraturan dan perundang-undangan yang berlaku;

Demikian surat pernyataan ini saya buat agar dapat dipergunakan sebagaimana mestinya.

Palembang, 23 September 2024

Yang membuat pernyataan,

Dwi Robbi Prasetyo

NIM: 191420064

MOTTO DAN PERSEMBAHAN

MOTTO

"Pendidikan bukan tentang mengenai mengisi wadah yang kosong, tapi pendidikan merupakan proses untuk menyalakan api pikiran."
(B. Yeats)

PERSEMBAHAN

Rasa syukur dan bahagia ini akan saya persembahkan kepada:

1. Kepada Allah SWT yang telah memberikan saya kesehatan, kekuatan dan kemudahan agar dapat menyelesaikan karya akhir ini.
2. Kedua orang tua tercinta, yang selalu memberikan cinta, doa, dukungan, dan pengorbanan yang tiada henti. Terima kasih atas segala keikhlasan dan kasih sayang yang kalian berikan.
3. Saudara-saudaraku yang selalu memberikan motivasi dan semangat di setiap langkah perjuangan dalam meraih mimpi.
4. Para sahabat dan teman seperjuangan, yang selalu ada untuk berbagi tawa, cerita, dan dukungan selama masa kuliah dan penyusunan skripsi ini. Terima kasih atas segala bantuannya.
5. Dosen pembimbing, yang dengan sabar memberikan bimbingan, ilmu, dan arahan hingga skripsi ini dapat terselesaikan.
6. Almamater tercinta, Universitas Bina Darma yang telah menjadi tempat belajar dan tumbuh selama ini.

Semoga karya akhir ini dapat bermanfaat bagi semua pihak yang membacanya dan menjadi langkah awal untuk pencapaian yang lebih besar di masa depan.

ABSTRAK

Memiliki akses cepat ke informasi sangat penting. Sistem Dasawisma PKK Provinsi Sumsel digunakan untuk mengumpulkan informasi program Dasawisma, penduduk, dan kegiatan. Seiring waktu data semakin besar, sehingga ditemui kendala dalam hal kecepatan operasi relasional, terutama saat menampilkan rekap data pada *dashboard*. Untuk mengatasi kendala tersebut, penelitian ini mengusulkan implementasi *in-memory database* (Redis). Redis merupakan *database* berbasis memori sehingga penyimpanan dan akses data dapat secara cepat dan efisien. Penelitian ini menggunakan metode pengembangan ADDIE *(Analysis, Design, Development, Implementation, Evaluation)*. Tahap Analisis menentukan waktu dan tempat penelitian, melakukan pengumpulan data, dan penentuan kebutuhan. Tahap desain, merancang desain teknis. Tahap pengembangan, mengimplementasikan redis. Tahap implementasi, dilakukan pengujian. Tahap evaluasi, menganalisis data hasil pengujian. Tujuan penelitian ini adalah untuk mempercepat proses pengambilan data meskipun *volume* data semakin besar. Penelitian ini menghasilkan rata-rata waktu respon *request* ke-2 dan seterusnya hanya dibutuhkan waktu yang singkat dalam memuat data yang semakin besar selama belum ada perubahan data.

Kata Kunci: Data, Operasi relasional, *in-memory Database*, ADDIE, Redis

ABSTRACT

Having quick access to information is essential. The South Sumatra Province PKK Dasawisma System is used to collect information on Dasawisma programs, residents, and activities. Over time, the data is getting bigger, so there are obstacles in terms of the speed of relational operations, especially when displaying data recap on the dashboard. To overcome these obstacles, this research proposes the implementation of an in-memory database (Redis). Redis is a memory-based database so that data storage and access can be fast and efficient. This research uses the ADDIE development method (Analysis, Design, Development, Implementation, Evaluation). The analysis stage determines the time and place of research, conducts data collection, and determines the needs. Design stage, designing technical design. Development stage, implementing redis. Implementation stage, testing is carried out. Evaluation stage, analyzing test result data. The purpose of this research is to speed up the data retrieval process even though the volume of data is getting bigger. This research results in the average response time of the 2nd request and so on only takes a short time in loading increasingly large data as long as there is no data change.

Keywords: *Data, Relational operations, In-memory Database, ADDIE, Redis*

KATA PENGANTAR

Puji syukur kehadirat Tuhan Yang Maha Esa atas berkat rahmatnya sehingga peneliti dapat menyelesaikan Skripsi yang berjudul **"IMPLEMENTASI *IN-MEMORY DATABASE* (REDIS) UNTUK MEMPERCEPAT OPERASI RELASIONAL DALAM MENAMPILKAN REKAP DATA PADA *DASHBOARD* SISTEM DASAWISMA PKK PROVINSI SUMATERA SELATAN"** tepat pada waktu yang ditentukan.

Skripsi ini diajukan untuk memenuhi syarat dalam menyelesaikan program Sarjana (S1) pada jurusan Teknik Informatika Universitas Bina Darma Palembang. Dalam penulisan ini, peneliti menyadari bahwa tanpa adanya bimbingan, bantuan, dukungan dan petunjuk dari semua pihak tidak mungkin laporan ini dapat diselesaikan. Oleh karena itu pada kesempatan ini peneliti ingin menyampaikan rasa terima kasih kepada:

1. Prof. Dr. Sunda Ariana, M.Pd., M.M., selaku Rektor Universitas Bina Darma
2. Dr. Tata Sutabri, S.Kom., MMSI., MKM., selaku Dekan Fakultas Ilmu Komputer Universita Bina Darma
3. Alek Wijaya, S.Kom., M.IT., selaku Ketua Program Studi Teknik Informatika Universitas Bina Darma
4. Siti Saudah, M.Kom. selaku Sekretaris Program Studi Teknik Informatika Universitas Bina Darma
5. Fatoni M.M., M.Kom., selaku Dosen Pembimbing
6. Orang tua, saudara-saudari ku, seluruh teman dan sahabat-sahabatku yang telah banyak memberikan semangat dan motivasi sehingga dapat menyelesaikan laporan hasil penelitian ini
7. Semua pihak yang tidak bisa saya sebutkan satu persatu-satu namun tidak mengurangi rasa hormat saya ucapkan terimakasih yang sebesar-besarnya atas dukungannya dalam penulisan laporan hasil penelitian ini.

Dalam penyusunan skripsi ini, peneliti telah berusaha semaksimal mungkin supaya skripsi ini selesai dengan baik dan sempurna. Namun peneliti menyadari sebagai manusia yang tidak luput dari kesalahan dan kekhilafan, maka skripsi ini pun mungkin terdapat kekeliruan dan kekurangan kiranya mohon dimaklumi. Akhirnya, hanya kepada Tuhan Yang Maha Esa peneliti serahkan segalanya, mudah-mudahan skripsi ini dapat bermanfaat khususnya bagi peneliti dan umumnya bagi kita semua.

Palembang, 23 September 2024

Dwi Robbi Prasetyo

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

DAFTAR LAMPIRAN

Halaman

BAB I
PENDAHULUAN

1.1 Latar Belakang

Sistem Dasawisma PKK Provinsi Sumatera Selatan digunakan sebagai tempat untuk mengumpulkan data dalam jumlah yang semakin besar seiring waktu. Jumlah penduduk Provinsi Sumatera Selatan berdasarkan Sensus Penduduk tahun 2022 sebesar 8,657 juta jiwa. Data-data yang dimasukkan ke *website* Dasawisma PKK ini termasuk informasi tentang program Dasawisma, penduduk, dan kegiatan. Dengan jumlah data yang besar dapat mengakibatkan penurunan performa sistem dalam menampilkan rekap data pada bagian *Dashboard*.

Dashboard sistem Dasawisma PKK Provinsi Sumatera Selatan digunakan sebagai alat untuk memonitor dan menganalisis data terkait program Dasawisma yang dilaksanakan di Provinsi tersebut. Namun, dengan pertumbuhan jumlah data yang semakin besar dan kompleksitas operasi relasional yang tinggi, terjadi kendala performa yang signifikan dalam menampilkan rekap data pada *dashboard* tersebut.

Menampilkan rekap data pada *dashboard* melibatkan operasi relasional yang kompleks, seperti penggabungan data dari beberapa tabel, dan pengambilan data berdasarkan kriteria tertentu menggunakan *join query*. Menurut Sinuraya (Dalam Zulfa dkk., 2020) Strategi yang disediakan RDBMS (*Relational Database Management System*) untuk menampilkan data adalah tindakan kueri gabungan dengan mengambil data dari beberapa tabel yang berbeda. Menurut Teknovasi (Dalam Zulfa dkk., 2019) Kueri gabungan akan memerlukan waktu lebih lama untuk dieksekusi apabila semakin banyak tabel yang digabungkan. Operasi-operasi ini dapat membutuhkan waktu yang lama, terutama ketika data yang harus diolah dalam jumlah besar.

Luthfi (Dalam Zulfa dkk., 2019) *Database* relasional menyimpan datanya di dalam *disk* (HDD/SSD). Permasalahan tersebut dapat menjadi penyebab keterbatasan performa. *Disk* memiliki keterbatasan dalam kecepatan akses data, yang memperlambat waktu eksekusi *query* yang kompleks dan penarikan data dari *database*. Akibatnya, responsifitas sistem menurun dan waktu pemrosesan data menjadi lebih lama.

Untuk mengatasi kendala performa ini, penggunaan *database in-memory* seperti Redis menjadi relevan. Dalam beberapa tahun terakhir, Redis telah menjadi solusi yang populer untuk mengatasi masalah performa ini. Redis adalah *database open-source* yang memanfaatkan memori sebagai media penyimpanan data, yang memungkinkannya untuk memberikan akses langsung ke memori sehingga dapat memberikan kinerja yang jauh lebih cepat dalam pengolahan data dibandingkan dengan *database* relasional berbasis *disk*. Redis juga mendukung banyak struktur data, seperti *strings*, *hashes*, *sets*, *lists*, *sorted sets* (Meriani dkk., t.t.).

Dengan demikian, mengimplementasikan Redis sebagai *database in-memory*, diharapkan penelitian ini dapat mengatasi kendala performa yang ada dan meningkatkan kecepatan operasi relasional dalam menampilkan rekap data, sehingga dapat memberikan manfaat dalam meningkatkan responsifitas sistem, efisiensi pengambilan keputusan, dan memungkinkan pengguna untuk memperoleh informasi secara cepat dan akurat dari *website* tersebut.

1.2 Rumusan Masalah

Mengingat konteks yang diuraikan di atas, maka timbul beberapa masalah sebagai berikut:

1. Bagaimana mengimplementasikan *Database* Redis pada *web* Dasawisma PKK Provinsi Sumatera Selatan?
2. Berapa kecepatan *response time* yang didapatkan antara sebelum menggunakan *Database* Redis dan setelah menggunakan *Database* Redis?

3. Apakah dengan mengimplementasikan *Database* Redis dapat mempercepat kinerja *website* Dasawisma PKK Provinsi Sumatera Selatan dalam menampilkan Rekap Data ?

1.3 Tujuan Penelitian

Tujuan yang ingin dicapai dari penelitian ini adalah untuk mempercepat operasi relasional yang terlibat dalam pengambilan data yang saling berelasi, meningkatkan kinerja *website* sehingga pengguna dapat mengakses informasi dengan lebih cepat dan responsif, dan diharapkan kedepannya *website* Dasawisma PKK memungkinkan untuk menangani *volume* data yang lebih besar dengan kinerja yang baik, sehingga dapat mendukung pertumbuhan dan perkembangan *website* di masa depan.

1.4 Batasan Masalah

Berikut ini adalah beberapa batasan masalah pada penelitian ini:

1. Aspek *Dashboard*, Pengimplementasian *Database* Redis hanya berfokus pada halaman rekap data, artinya tidak termasuk halaman selain rekap data.
2. Penggunaan Redis, Operasi yang digunakan pada pengimplementasian *Database* Redis pada *website* Dasawisma PKK Provinsi Sumatera Selatan hanya beberapa, seperti *exists*, *hSet, hMSet*, *hGet, hGetAll, transaction, keys*, *expire*, *hIncrBy* dan *del*.
3. Konteks Wilayah, Data yang dimasukkan tidak pada semua wilayah yang ada di Provinsi Sumatera Selatan. Dikarenakan data yang dimasukkan ke *Database* MYSQL (*Database* Utama) menggunakan *dummy* data, maka tidak semua Wilayah Baik itu Kabupaten/Kota, Kecamatan, Kelurahan/Desa bisa terisi semua.
4. *User*, Pada kenyataanya *website* ini hanya bisa diakses oleh pengguna internal dan eksternal tertentu dari pihak yang terlibat dengan Dasawisma PKK Provinsi Sumatera Selatan.

5. *Platform* (sistem) yang akan dibuat yaitu berbasis web.
6. Redis, MySQL, dan Laravel adalah teknologi yang digunakan dalam pembangunan sistem.

1.5 Manfaat Penelitian

Dengan adanya penelitian ini diharapkan bisa memberikan manfaat, antara lain sebagai berikut:

1. Bagi Tim Dasawisma PKK
   a. Akses Data yang Lebih Cepat: Implementasi Redis sebagai *in-memory database* akan mempercepat akses data pada *dashboard* Dasawisma PKK. Tim Dasawisma PKK dapat dengan cepat dan efisien mengakses data rekap yang diperlukan untuk analisis, pemantauan, dan pengambilan keputusan. Hal ini akan meningkatkan efisiensi dan produktivitas tim dalam bekerja dengan data Dasawisma.
   b. Responsifitas Sistem yang Tinggi: Dengan Redis, Tim Dasawisma PKK akan mengalami respons sistem yang lebih cepat saat mengakses dan menampilkan rekap data pada *dashboard*. Waktu tunggu yang berkurang akan memungkinkan tim untuk mendapatkan informasi secara cepat dan mengambil tindakan yang diperlukan secara lebih responsif.
   c. Analisis Data yang Lebih Efektif: Implementasi Redis memungkinkan operasi relasional yang lebih cepat dan efisien. Tim Dasawisma PKK akan dapat melakukan analisis data dengan lebih efektif, termasuk perhitungan statistik, dan pengambilan data berdasarkan kriteria tertentu. Hal ini akan membantu tim dalam memahami data dengan lebih baik, mengidentifikasi pola, dan membuat keputusan yang lebih terinformasi.

2. Bagi Peneliti

   Peneliti akan memperoleh pengetahuan, memberikan kontribusi, keterampilan, serta memberikan dampak positif bagi sistem Dasawisma PKK Provinsi Sumatera Selatan dan orang yang menggunakannya.

3. Bagi Universitas

   Universitas akan memperoleh peningkatan peluang kolaborasi dengan industri, dan pengembangan sumber daya penelitian. Selain itu, hasil penelitian ini juga dapat berkontribusi pada pembaruan kurikulum pendidikan dan memberikan manfaat langsung kepada mahasiswa dalam meningkatkan pemahaman mereka tentang teknologi informasi terkini.

BAB II
KAJIAN TEORI

Landasan teori, Penelitian terdahulu dan kerangka berfikir penelitian yang dilakukan dibahas pada bagian ini. Terdapat beberapa penjelasan teoritis dan penelitian sebelumnya yang membahas mengenai implementasi *Database* Redis.

2.1 Implementasi

Kamus Besar Bahasa Indonesia mengartikan implementasi sebagai pelaksanaan dan penerapan. Beberapa pendapat para ahli tentang implementasi adalah sebagai berikut. Usman (Dalam Rosad, 2019), menyampaikan pemikirannya tentang implementasi sebagai berikut. "Gerakan, aktivitas, tindakan, atau keberadaan mekanisme sistem semuanya disebabkan oleh implementasi". Implementasi lebih dari sekadar gerakan, implementasi adalah tindakan yang disengaja yang dimaksudkan untuk mencapai tujuan aktivitas tertentu. Berdasarkan uraian di atas, implementasi bukanlah kegiatan yang asal-asalan, melainkan tindakan yang direncanakan dan dilakukan dengan cermat sesuai dengan serangkaian pedoman yang diterima untuk mencapai tujuan yang diinginkan.

Sementara itu, Harsono (Dalam Rosad, 2019), mendefinisikan implementasi sebagai "perluasan kegiatan yang saling menyesuaikan proses interaksi antara tujuan dan tindakan untuk mencapainya dan memerlukan jaringan pelaksana yang efektif, birokrasi." Dengan demikian, implementasi dapat didefinisikan sebagai proses menempatkan ide, protokol, atau serangkaian langkah baru ke dalam tindakan dengan harapan bahwa orang lain akan mengadopsinya dan membuat penyesuaian yang diperlukan untuk mengubahnya menjadi tujuan yang dapat dicapai dengan jaringan pelaksana yang dapat dipercaya.

Dari kedua pengertian diatas dapat diartikan bahwa implementasi adalah penerapan suatu kegiatan yang sudah di rencanakan untuk mencapai tujuan dan memperoleh hasil akhir yang diinginkan.

2.2 Dasawisma PKK

Dasawisma PKK adalah kelompok PKK yang terdiri dari 10 hingga 20 rumah/kepala keluarga di tingkat RT. Tiga tanggung jawab utama kader Dasawisma adalah mendata warga, mengorganisir, dan menyebarluaskan informasi. Pemerintah Provinsi Sumatera Selatan akan menggunakan data yang telah diperolehnya untuk mengoptimalkan layanan masyarakat dan melakukan berbagai intervensi. Memelihara data statistik kependudukan sangat penting untuk mencegah terjadinya kesalahan, salah satunya dalam distribusi bantuan sosial. (Aziz & Yaya Mulyana Abdul, 2019).

2.3 Data

Data adalah informasi yang belum diproses yang telah diubah menjadi pengetahuan yang lebih berharga yang dapat membantu pengguna dalam mencapai tujuan yang diinginkan. Data adalah representasi dari benda, orang, tempat, peristiwa, kegiatan, ide, dan hal-hal penting lainnya dengan menggunakan ekspresi matematis, bahasa, dan simbol-simbol alternatif yang telah disepakati. Data adalah komponen bahan baku informasi yang disusun sebagai sekumpulan gelombang non-random yang mengindikasikan aktivitas atau jumlah di antara hal-hal lainnya. (Sumantri dkk., 2022).

2.4 Sistem

Para ahli mengatakan bahwa penafsiran berikut mengenai apa yang membentuk sebuah sistem adalah mungkin. menurut Ludwig von Bartalanfy, sistem adalah kumpulan elemen yang dihubungkan oleh interaksi antara elemen dan lingkungan sekitarnya. Lebih jauh, sistem didefinisikan oleh

Romney sebagai kumpulan dua atau lebih komponen yang saling terkait yang bekerja sama untuk mencapai suatu tujuan, dan oleh Anatol Raporot sebagai sekumpulan unit dan perangkat yang terkait. Sebaliknya, sistem, menurut definisi Hall, adalah pengelompokan dua atau lebih komponen atau subsistem yang saling berhubungan yang bekerja sama untuk mencapai tujuan bersama.

Terdapat tiga komponen yang membentuk sistem, yaitu *input*, proses, dan *output*. Sistem digerakkan atau diberi energi oleh masukannya, dan keluarannya adalah produk akhir dari tindakan-tindakan tersebut. Secara sederhana, *output* adalah apa yang ingin dicapai atau ditargetkan oleh sebuah sistem selama beroperasi, Sebaliknya, proses adalah tindakan yang mengubah masukan menjadi keluaran.

Berdasarkan beberapa definisi sistem yang dibahas di atas, dapat dikatakan bahwa sistem adalah sekumpulan komponen yang saling terhubung dan bekerja sama untuk mencapai tujuan bersama. (Frisdayanti, 2019).

2.5 *Dashboard* (*Web Admin*)

Dashboard (*Web Admin*) adalah sebuah sistem yang disediakan bagi para admin untuk memberikan dan mengelola data dari suatu informasi berbasis *web* di suatu lembaga organisasi atau perusahaan (D. Irawan & Hidayat, 2019). Beragam informasi dari berbagai sumber dengan cara (*format/layout*) yang seragam tersedia pada *dashboard web admin* atau biasa disebut sebagai *portal web admin*. Setiap sumber informasi mendapat area khusus pada halaman *website portal* dalam menampilkan informasinya (Ramadhan dkk., 2020). Kemampuan dari *portal web admin* yang lebih spesifik adalah menyediakan suatu informasi dimana dapat diakses dengan memakai berbagai perangkat seperti komputer, *notebook* atau bahkan telepon genggam.

2.6 *Website*

Web merupakan salah satu layanan yang ada di internet. Teknologi *server web* berfungsi sebagai penyedia halaman *web*, HTML (*HyperText Markup Language*) merupakan bahasa standar *web*, dan HTTP merupakan mekanisme pengiriman dokumen *web* untuk mengakses *internet*. *Website* adalah halaman informasi *online* yang dapat dikunjungi dari lokasi mana pun selama terhubung ke internet. Sebuah situs *web* terdiri dari beberapa elemen seperti teks, foto, suara, dan animasi untuk meningkatkan daya tariknya sebagai sumber informasi (Rochman dkk., 2020).

Berdasarkan kategorinya, *website* digolongkan menjadi 3. Antara lain sebagai berikut:

1. *Website* Statis, situs *web* yang menyediakan halaman statis. Jika halaman perlu diedit, pengeditan dilakukan secara manual dengan membuat perubahan pada kode yang membentuk struktur situs *web*.
2. *Website* Dinamis, Melalui sistemnya, data yang ada di dalamnya dapat diperbarui.
3. *Website* Interaktif, Situs *web* dengan fitur interaktif yang memungkinkan pengunjung untuk berkomunikasi dan memperdebatkan ide.

2.7 *Database*

Database atau basis data terdiri dari dua kata, yaitu Data dan *Base* atau basis. Fakta-fakta dari dunia nyata diwakili oleh data, yang dapat berupa apa saja, mulai dari orang (pekerja, siswa, profesor, dll.) hingga objek, hewan, peristiwa, dan banyak lagi. Biasanya, data disimpan dalam bentuk teks, grafik, angka, karakter, simbol, dan bentuk lainnya. Sedangkan *Base* dapat diartikan, tempat, gudang, atau tempat penyimpanan.

Sehingga dapat disimpulkan bahwa *Database* adalah kumpulan informasi yang diperoleh dari suatu data dan selanjutnya disimpan dalam suatu media atau komputer berbasis *disk* (seperti *harddisk*, *floppydisk*,

flashdisk) agar data tersimpan secara permanen. Tujuan utama *database* adalah memilih, mengelompokkan, dan mengatur data dengan tepat.

Dalam konteks era informasi yang canggih dan berkembang pesat saat ini, penggunaan *database* sebagai sumber informasi merupakan alat bantu yang dimaksudkan untuk memudahkan pencarian informasi atau data. Data yang dapat memberikan informasi dikumpulkan dan disimpan dalam *database* (Suprayogi dkk.).

2.7.1 *Database Management System* (DBMS)

Pengertian *Database Management System* menurut Zakia (dalam Gunadi & Widianto, 2020) merupakan penghubung antara pengguna dan *database*, namun demikian *database* itu sendiri diprogram dengan bahasa yang sudah ditentukan oleh perusahaan yang membuat sistem DBMS. Dengan menggunakan DBMS, pengguna diharapkan dapat mendefinisikan basis data menggunakan *Data Definition Language* (DDL) untuk menentukan jenis, struktur, dan batasan data yang nantinya akan disimpan dalam basis data. Selain itu pengguna dapat menambah, memodifikasi, menghapus, atau mengambil data dari *database* yang dibuat menggunakan *Data Menipulation Language* (DML). Dan untuk pengaturan memberikan hak otorisasi dalam mengakses *database*, mengalokasikan *space*, pendifinisian *space*, dan pengauditan penggunaan *database* itu semua dilakukan menggunakan *query Data Control Language* (DCL). Saat ini banyak ditemukan DBMS, antara lain MySQL, Oracle, PostgreSQL, MongoDB, Redis dan lain – lain.

2.7.2 *Relational Database Management System* (RDBMS)

Teori basis data relasional pertama kali dikembangkan pada tahun 1960-an oleh pakar komputer Inggris Edgar Frank Codd. Menurut Nugroho (Dalam Anugraha dkk., 2020) pengguna *database* dapat membuat, mengelola, dan menggunakan data dalam *model* relasional dengan bantuan RDBMS.

Basis data disusun ke dalam tabel dua dimensi. Ada dua jenis kolom dalam setiap tabel, yaitu baris data (*row*/*record*) dan lajur vertikal (*column*/*field*). Masing-masing dari kolom dapat dideklarasikan tipe datanya, dan *key* (*primary-key*, *foreign-key*). Tapi perlu diketahui bahwa tidak semua jenis *database* relasional mendukung fitur *multiple primary-key*. Bahasa SQL (*Structured Query Language*) digunakan untuk bisa berkomunikasi dengan *database* jenis relasional. SQL itu sendiri adalah sebuah bahasa standar untuk relasional *database*. Adapun beberapa dari DBMS yang menggunakan konsep basis data relasional antara lain *SQLite, MYSQL, PostgreSQL* dan *Oracle*.

2.7.3 NoSQL (*Not Only SQL*)

Basis data non-relasional, atau basis data NoSQL, menawarkan mekanisme untuk menyimpan dan mengambil data menggunakan model yang berbeda dari hubungan tabel konvensional yang terlihat pada basis data relasional. *Database* ini memiliki kinerja yang sangat baik, skalabilitas, dan dibuat untuk mengelola data tidak terstruktur atau semi-terstruktur dalam jumlah besar. Basis data NoSQL menyediakan model data yang lebih fleksibel daripada basis data relasional, yang memberlakukan skema yang ketat dan hubungan yang telah ditentukan sebelumnya antar tabel. Terdapat berbagai macam format data, seperti *key-value pair*, dokumen, kolom, atau grafik.. Karena kemampuan fleksibilitasnya, *database* NoSQL dapat mengelola struktur data dinamis yang rumit dan berubah sesuai kebutuhan penggunanya (Hasibuan & Nasution, 2023).

Basis data non-relasional sangat cocok untuk ide Big Data, yang membutuhkan penyimpanan data yang tidak terstruktur (*variety*), prosedur pembacaan/pembaruan yang sederhana dan cepat (*velocity*), dan penyimpanan data yang besar (*volume*) dengan sumber daya yang terbatas (*server*). (Lozada dkk., 2019). Karena hampir setiap interaksi sistem melibatkan data, jumlah (*volume*) data yang dihasilkan akan bertambah seiring berjalannya waktu. Selain ukuran, keragaman data yang dihasilkan

mencakup berbagai jenis dan varian dalam hal format file seperti teks dan gambar video, serta struktur data seperti data terstruktur, semi-terstruktur, dan tidak terstruktur. Kecepatan untuk menyelesaikan proses perhitungan data harus diperhitungkan karena data tersebut harus dievaluasi secara cepat dan real-time untuk menghasilkan informasi (Baig dkk., 2019). Setelah itu, informasi dikumpulkan secara sistematik dan dihubungkan satu sama lain agar terbuatlah sebuah *database*. Adapun beberapa DBMS yang menggunakan konsep basis data nonrelasional antara lain MongoDB, Apache Cassandra, dan Redis.

2.7.4 *Server* dan *Client* DBMS

DBMS (*Database Management System*) adalah suatu sistem perangkat lunak yang memungkinkan pengguna untuk membuat, memelihara, mengontrol, dan mengakses *database* secara praktis dan efisien. Dengan menggunakan DBMS pengguna akan mempermudah pengguna untuk mengelola dan mengubah data yang ada. Arsitektur DBMS dapat diklarifikasikan ke dalam dua jenis, yaitu *server* dan *client*. *Server* adalah komputer *database* terpusat, dimana informasinya dapat digunakan bersama-sama oleh beberapa pengguna yang menjalankan aplikasi komputer lokalnya yang disebut *client* (Pasha dkk., 2020) (Putra dkk., 2021).

2.7.5 Operasi Dasar *Database*

Operasi-operasi dasar yang dapat kita lakukan berkenaan dengan basis data dapat meliputi (Ruliah dkk.):

1. Pembuatan basis data baru (*create database*), yang identik dengan pembuatan lemari arsip yang baru.
2. Penghapusan basis data (*drop database*), yang identik dengan perusakan lemari arsip (sekaligus beserta isinya, jika ada).

3. Pembuatan *file*/tabel baru ke suatu basis data (*create table*), yang identik dengan penambahan map arsip baru ke sebuah lemari arsip yang telah ada.
4. Penghapusan *file*/tabel dari suatu basis data (*drop table*), yang identik dengan perusakan map arsip lama yang ada di sebuah lemari arsip.
5. Penambahan/pengisian data baru ke sebuah *file*/tabel di sebuah basis data (*insert*), yang identik dengan penambahan lembaran arsip dari sebuah map arsip.
6. Pengambilan data dari sebuah *file*/tabel (*retrieve/search*), yang identik dengan pencarian lembaran arsip dari sebuah map arsip.
7. Perubahan data dari sebuah *file*/tabel (*update*), yang identik dengan perbaikan isi lembaran arsip yang ada di sebuah map arsip.
8. Penghapusan data dari sebuah *file*/tabel (*delete*), yang identik dengan penghapusan sebuah lembaran arsip yang ada di sebuah map arsip.

2.7.6 Penerapan *Database*

Basis data dapat digunakan di hampir semua departemen dalam sebuah organisasi, termasuk sektor bisnis publik dan komersial (Ruliah dkk.). Secara lebih teknis/nyata, bidang-bidang fungsional seperti kepegawaian, akuntansi, pergudangan (inventori), layanan pelanggan, dan lainnya adalah yang sering menggunakan *database* untuk meningkatkan produktivitas, akurasi, dan kecepatan operasi. Sedangkan bank, asuransi, rumah sakit, pembuat barang, industri manufaktur, pendidikan/sekolah, telekomunikasi, dan jenis organisasi/perusahaan lainnya adalah bentuk-bentuk organisasi atau perusahaan yang menggunakan *database*.

2.8 MySQL

Menurut Rulianto (Dalam Hermiati dkk., 2021) MySQL merupakan salah satu jenis *server* basis data yang cukup terkenal dan juga *open source*, bahasa yang digunakan untuk mengakses *database*-nya adalah SQL

(*Structured Query Language*). Karena sebuah organisasi yang dikenal sebagai ANSI (*American National Standards Institute*) telah menstandarkan sejumlah pedoman penggunaan SQL. Salah satu cara untuk mendeskripsikan MySQL adalah sebagai sebuah implementasi sistem manajemen *database* relasional gratis yang tersedia di bawah lisensi GPL (*General Public License*). MySQL tersedia untuk semua pengguna tanpa batasan, namun menggunakan program ini sebagai produk turunan komersial dilarang. Pengguna *database* dapat membuat, mengelola, dan memanfaatkan data dalam model relasional dengan bantuan RDBMS. Dengan demikian, tabel-tabel yang ada pada *database* memiliki relasi antara satu tabel dengan tabel lainnya.

Seperti yang telah dijelaskan sebelumnya, MySQL adalah perangkat lunak *database* sumber terbuka, *multithreaded*, dan *multiuser* yang banyak digunakan untuk mengelola *database* yang memungkinkan pengelolaan data dapat dilakukan dengan mudah.

2.9 Redis

Redis (*Remote Dictionary Server*) merupakan *database* yang berbasis *key-value* yang dikembangkan oleh Salvatore Sanfilippo yang diliris pada 10 Mei 2009 (Irawan & Ramdhani). Redis merupakan salah satu jenis *database* jenis NoSQL. Redis memiliki keuntungan dapat diakses secara cepat karena penyimpanan *dataset* berbasis memori. Oleh karena itu, Redis dapat membaca data lebih cepat daripada basis data NoSQL lainnya. Redis dirancang dengan fokus pada kecepatan dan kinerja tinggi. Karena menggunakan penyimpanan data berbasis memori, Redis dapat memberikan akses data yang sangat cepat. Hal ini membuat redis sangat cocok digunakan dalam aplikasi yang membutuhkan *respons-time* rendah, seperti sistem *caching*, antrian pesan, dan sesi pengguna.

2.9.1 Tipe Data Redis

Redis memiliki berbagai macam struktur data untuk memenuhi kebutuhan dalam pembangunan atau pengembangan sebuah *software*. Berikut adalah beberapa tipe data pada redis, mencakup (Meriani dkk.):

1. *String*, data teks atau *biner* hingga berukuran 512 MB
2. *List*, kumpulan *string* pada urutan ditambahkan
3. *Set*, kumpulan *string* yang tidak berurutan dengan kemampuan memotong, menyatukan, dan membedakan tipe *set* lain
4. *Sorted Sets*, *Set* yang diurutkan berdasarkan nilai
5. *Hash*, struktur data untuk menyimpan daftar *key* dan *value*.

2.9.2 Fungsi Redis

Sebagai *database* yang bersifat *open-source*, redis memiliki beberapa fungsi utama, yaitu (Zulfa dkk., 2020):

1. *Caching*, Redis sering digunakan sebagai penyimpanan *cache* untuk mempercepat akses ke data yang sering digunakan. Data yang sering diakses dari *database* atau aplikasi dapat disimpan di Redis untuk mengurangi latensi.
2. Analisis *Real-Time*, Redis dapat digunakan untuk mengambil, memproses, dan menganalisis data secara *real-time* karena kemampuannya menyimpan data dalam memori dengan latensi rendah.
3. Data *Eviction*, Struktur data Redis umumnya ditandai dengan *Time To Live* (TTL) yang dipasang dalam hitungan detik, kemudian akan dihapus. *Eviction Policy* dapat dikonfigurasi pada *file* konfigurasi redis.

2.9.3 Keunggulan Redis

Redis memiliki beberapa keunggulan dari *database* jenis NoSQL lainnya yang menjadikannya populer dalam pengembangan aplikasi

berkinerja tinggi dan *real-time*. Beberapa keunggulan utama Redis meliputi (Prabowo dkk.):

1. Redis menyimpan datanya pada memori utama *server*. Sehingga yang membuatnya sangat cepat dalam pengambilan data. Dengan latensi yang rendah, Redis dapat menangani beban kerja tinggi dengan baik.
2. Redis mendukung banyak bahasa pemrograman, seperti *Python, PHP, C, C++, C3, Ruby, R, Go* dan lain sebagainya. Sehingga memudahkan pengembang untuk melakukan integrasi dengan redis.
3. Struktur data yang beragam, seperti *strings*, *list*, *set*, *hash*, *bitmaps*, *HyperLogLog*, *stream*, *Geospatial*, dan lainnya. Sehingga memungkinkan pengembang untuk memodelkan data dengan cara yang paling sesuai dengan kebutuhan *software*.
4. *Open Source*, Redis bersifat *open-source*, sehingga dapat digunakan tanpa biaya lisensi. Ini membuatnya sangat ekonomis untuk digunakan dalam proyek-proyek besar dan kecil.

2.9.4 Kelemahan Redis

Selain memiliki keunggulan, *database* redis juga memiliki kelemahan, antara lain sebagai berikut :

1. Lebih mahal, dikarenakan redis menyimpan datanya pada memori, maka biaya untuk menambah sebuah memori akan lebih mahal daripada jenis *database* yang tipe penyimpananya berada pada *disk*.
2. Kinerja menurun saat memori penuh, Ketika memori redis mulai mendekati kapasitas maksimum, kinerjanya dapat menurun drastis, bahkan hingga menyebabkan kegagalan.
3. Keterbatasan *Querying*: Redis tidak dirancang untuk melakukan operasi kompleks pencarian dan pengambilan data seperti yang dilakukan oleh *database* relasional atau NoSQL yang lebih kuat dalam hal *querying*.

2.9.5 Redis Cocok Sebagai *Cache*

Redis adalah solusi yang cocok untuk melakukan *caching* data besar Terdapat beberapa komponen penting yang menjelaskan mengapa Redis cocok sebagai solusi *caching*. Antara lain sebagai berikut (Febriyani dkk.):

1. Kecepatan dan Responsifitas

   Redis dikenal sebagai *database cache in-memory* yang sangat cepat. Redis memiliki kemampuan pembacaan data yang lebih cepat dibandingkan NoSQL lainnya. Dikarenakan redis merupakan *database* yang berbasis memori. Pembacaan data di memori akan lebih cepat daripada *disk*. Walaupun Redis dikenal sebagai sebuah *database*, tetapi Redis juga dapat berperan sebagai *cache*. Tentu saja ini dapat meningkatkan responsifitas sistem Dasawisma dalam menampilkan rekap data yang kurang lebih 500 ribu sampai 1 juta. Ini melibatkan pemahaman tentang seberapa cepat Redis dapat mengambil dan mengirim data ke halaman rekap data menggunakan *cache*.

2. Jenis Tipe Data yang di *Cache*

   Redis memiliki fleksibilitas dalam menyimpan berbagai jenis data, termasuk *string*, *hash*, *list*, *set*, dll. Untuk kasus pada judul penelitian ini, maka peneliti memilih tipe data *string* dalam bentuk JSON sebagai jenis data yang akan di *cache* atau ditampung oleh redis. Dikarenakan tipe data *string* dapat mengakomodasi apapun itu jenis datanya.

3. Skalabilitas

   Redis sangat skalabel dan dapat dengan mudah didistribusikan. Mampu untuk menangani beban kerja besar yang diberikan oleh *client*. Sehingga dapat meningkatkan kecepatan *response time*.

4. Manajemen *Cache*

   Manajemen *Cache* di Redis sangat fleksibel dan efisien. Data di redis dapat diatur waktunya seberapa lama data tersebut akan tersimpan di dalam memori. Artinya, tidak perlu takut kapan data di Redis akan

dihapus, jika sudah diatur waktunya maka data akan dihapus secara otomatis oleh Redis.

2.10 *Laravel*

Menggunakan arsitektur *Model View Controller* (MVC), Laravel adalah *framework* yang dibuat menggunakan bahasa PHP yang dirancang untuk membuat aplikasi. Definisi *framework* adalah sebuah kerangka kerja yang memfasilitasi pembuatan dan pengembangan perangkat lunak oleh pengembang perangkat lunak. (Somya & Nathanael, 2019). *Framework* adalah kumpulan operasi dan instruksi dasar yang sering digunakan untuk menghasilkan perangkat lunak, memungkinkan pembuatan program yang lebih terorganisir, lebih cepat, dan lebih rapi.

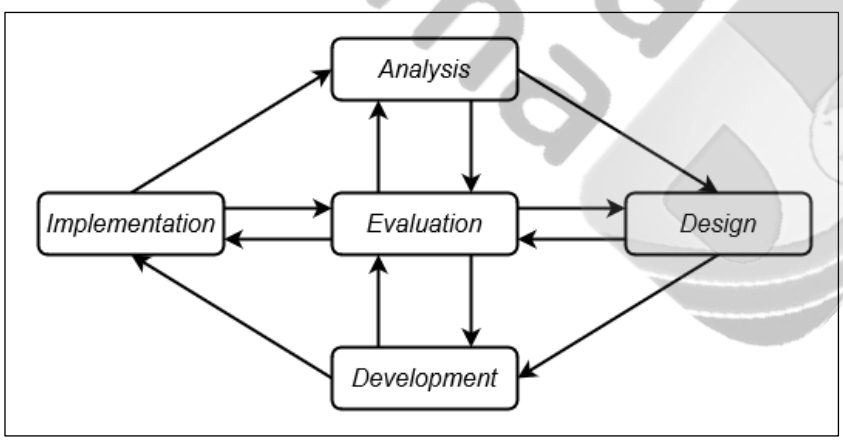
Taylor Otwell adalah pencipta *framework* Laravel, yang dirilis di bawah lisensi sumber terbuka pada tanggal 9 Juli 2011. Dengan kata lain, Anda tidak perlu membayar apa pun untuk menggunakannya.

2.11 *Cache*

Tindakan menyimpan sementara teks, gambar, atau materi lain di situs *web* untuk menghemat pemuatan dan lalu lintas *server* dikenal sebagai *cache*. *Cache*, sederhananya, adalah teknik yang memfasilitasi tampilan halaman *web* yang lebih cepat. berbeda dengan *cookie*, yang mencatat perilaku dan jejak pengguna saat mereka *online*. Data yang ada di dekatnya dapat disalin melalui *cache* (Khoshkholgh dkk., 2019). *Caching* terdiri dari dua jenis: *caching* sisi *client* dan *caching* sisi *server*. *Caching* sisi *server* terjadi di *server*, sedangkan *caching* sisi *client* terjadi di komputer pengguna dan dapat dikontrol melalui pengaturan *browser*.

2.12 Metode ADDIE

Metode ADDIE merupakan model yang melibatkan tahap-tahap pengembangan model dengan lima langkah meliputi: *Analysis*, *Design*, *Development*, *Implementation* dan *Evaluation*. Metode ADDIE dikembangkan oleh Dick dan Carry pada tahun 1996. Metode ini disusun secara terprogram dan dengan urutan yang sistematis. Menurut Molenda (Dalam Rohaeni, 2020) mengatakan bahwa metode *ADDIE* merupakan metode pembelajaran yang bersifat umum dan sesuai untuk penelitian pengembangan karena proses ini dianggap berurutan tetapi juga interaktif (komunikasi dua arah atau lebih dari komponen-komponen komunikasi). Sedangkan menurut F. G. Constancio (dalam Yulianti, 2021) Model ADDIE memberi peluang untuk melakukan evaluasi terhadap aktivitas pengembangan pada setiap tahap. Pada **Gambar 2.1** merupakan alur proses model ADDIE.



Gambar 2.1 Alur proses model ADDIE

Tahapan-tahapan yang dilakukan adalah sebagai berikut:

1. Tahap analisis (*analysis*), adalah memeriksa kebutuhan untuk mengembangkan barang baru (model, teknik, media, materi pendidikan), serta prasyarat dan kelayakan untuk melakukannya. Kesalahan pada produk yang ada saat ini atau produk yang sedang digunakan dapat menjadi pendorong untuk menciptakan produk baru. Masalah dapat muncul karena produk yang ada saat ini tidak memenuhi kebutuhan target, lingkungan belajar, karakteristik siswa, teknologi, dan sebagainya.

2. Tahap perancangan (*design)*, Pada tahap ini , dimulai dengan pembuatan konsep dan konten produk. Setiap desain produk dibuat khusus untuk produk tersebut. Instruksi ditulis dengan cara yang jelas dan komprehensif untuk menerapkan desain atau memproduksi produk. Desain produk masih bersifat konseptual pada tahap ini dan akan berfungsi sebagai dasar untuk proses pengembangan tahap berikutnya..
3. Tahap pengembangan (*development*), adalah upaya untuk mewujudkan produk yang telah dirancang sebelumnya. Kerangka konseptual untuk memperkenalkan produk baru dibuat pada tahap awal. Setelah itu, kerangka konseptual tersebut diubah menjadi produk jadi yang siap digunakan. Pada tahap ini, pengukuran kinerja produk juga memerlukan instrumen.
4. Tahap implementasi (*implementation*), untuk mendapatkan masukan pada produk yang dibuat atau diproduksi, produk tersebut diterapkan pada tahap ini menggunakan pendekatan penelitian dan pengembangan ADDIE. Pengujian item yang terkait dengan hasil tahap pengembangan dapat menghasilkan umpan balik awal (evaluasi pertama). Implementasi dilakukan dengan mengacu pada rencana produk yang dibuat.
5. Tahap evaluasi (*evaluation*), pada tahap terakhir ini yaitu dilakukan evaluasi dan menganalisis data hasil pengujian yang diperoleh, dengan membandingkan kinerja produk yang dikembangkan ketika sebelum dan setelah implementasi suatu solusi. Tujuan akhir evaluasi yakni mengukur ketercapaian tujuan pengembangan.

2.13 Penelitian Terdahulu

Penelitian terdahulu digunakan sebagai bahan perbandingan dan acuan. Berikut ini adalah beberapa penelitian terdahulu.

1. Strategi *caching* aplikasi berbasis *in-memory* menggunakan Redis *server* untuk mempercepat akses data relasional yang dilakukan oleh Mulki Indana Zulfa, Ari Fadli dan Arief Wisnu Wardhana. Untuk mempercepat

akses data relasional, studi ini akan melihat bagaimana data relasional akademis dimodelkan menjadi data yang kompatibel dengan Redis dan mengevaluasi seberapa baik kueri gabungan bekerja dalam hal ini. Untuk menghitung IPK mahasiswa dalam studi ini, data akademis direpresentasikan dalam RDBMS dan basis data dalam memori, yaitu Redis (IMDB). Penelitian ini menghasilkan, bahwa Redis dapat membantu meningkatkan kecepatan sistem MySQL hingga 1,7 kali, mengurangi waktu yang diperlukan untuk operasi *join* guna menampilkan data mahasiswa dan waktu yang diperlukan untuk membaca data IPK dari 61 milidetik menjadi 52 mikrodetik (Zulfa dkk., 2020).

2. Implementasi *Load Balancing NGINX* dan *MongoDB Cluster* serta Mekanisme Redis *Caching* yang dilakukan oleh, Faisal Al Isfahani dan Fuji Nugraha. Salah satu permasalahan yang muncul dalam penelitian ini adalah Volume data yang terus bertambah dikombinasikan dengan pemrosesan terpusat akan menghasilkan hasil yang kurang optimal. Ketidak optimalan tersebut berupa nilai *response time* yang tinggi, sehingga dibutuhkan sebuah arsitektur basis data selain arsitektur terpusat. Penelitian ini bertujuan untuk membangun arsitektur sistem terdistribusi dengan menggabungkan penyeimbangan beban *server web*, kluster basis data, dan teknologi *caching* yang memanfaatkan redis (Isfahani & Nugraha.).
3. Pengujian *Distributed Cached Database* Dengan Menggunakan Redis Pada Aplikasi MaBaUS. Penelitian ini dilakukan oleh Anggi Putri Meriani, Dwi Ramti Asih dan Siti Annisa. Pada tahap pengujian terhadap aplikasi MabaUS menggunakan Redis, disimpulkan bahwasanya *distributed cache* dapat mempercepat *request* terhadap data ataupun informasi yang akan di distribusikan. Hal tersebut berdasarkan angka kecepatan yang dihasilkan dari tahap pengujian yaitu terdapat kecepatan sebesar 570ms dengan tambahan 1,18 ms untuk aplikasi MaBaUS yang tidak

menggunakan *cache*, dan kecepatan 560 ms untuk Aplikasi MaBaUS yang menggunakan *cache* (Meriani dkk.).

2.14 Kerangka Berpikir

Pengertian kerangka berpikir menurut Sugiyono (Dalam Budiraharjo dkk., 2022) adalah model konseptual tentang bagaimana teori berhubungan dengan berbagai faktor yang telah diidentifikasi sebagai masalah penting. Sedangkan menurut Uma Sekaran (Dalam Hadi & Samad, 2019) mengemukakan bahwa "kerangka berpikir merupakan model konseptual tentang bagaimana teori berhubungan dengan berbagai faktor yang telah diidentifikasi sebagai hal yang penting". Dengan demikian, ide paling mendasar yang berfungsi sebagai dasar bagi semua ide lain atau sebagai metode untuk melaksanakan proyek penelitian secara lengkap dikenal sebagai kerangka berpikir. Kerangka berpikir pada penelitian ini dapat dilihat pada **Gambar 2.2**.

Gambar 2. 2 Kerangka Berpikir

BAB III
METODOLOGI PENELITIAN

3.1 *Analysis* (Analisis)

Pada tahap awal ini peneliti menentukan waktu dan tempat penelitian, melakukan analisa terhadap sistem Dasawisma PKK, penentuan kebutuhan, evaluasi redis sebagai solusi, dan rencana *caching*. Dengan menjalani tahapan analisis ini secara cermat dan komprehensif, peneliti akan memiliki landasan yang kuat untuk merencanakan, merancang, dan mengimplementasikan layanan yang efektif, efisien, dan sesuai dengan kebutuhan serta ekspektasi pengguna.

3.1.1 Waktu dan Tempat Penelitian

Waktu penelitian dilakukan selama 4 bulan mulai dari bulan November 2022 sampai dengan bulan Februari 2023. Sedangkan penelitian ini dilakukan pada Kantor Badan Penelitian dan Pengembangan Daerah Provinsi Sumatera Selatan, Jl. Demang Lebar Daun No. 4864, Kota Palembang.

3.1.2 Pengumpulan Data

Sebelum melakukan pengembangan sistem dibutuhkan adanya pengumpulan data. Peneliti menggunakan teknik pengumpulan data seperti, observasi dan studi literatur. Adapun penjelasannya adalah sebagai berikut:

1. Observasi

   Pertama, Peneliti mengumpulkan data awal yaitu dengan melakukan observasi langsung terhadap pengguna sistem Dasawisma PKK Provinsi Sumatera Selatan saat mereka menggunakan *website*. Observasi ini dapat memberikan wawasan tentang masalah kinerja yang ada dan bagaimana implementasi redis dapat membantu meningkatkan kecepatan operasi relasional.

Adapun fokus dari data yang dikumpulkan adalah pada bagian rekap data. Misalnya pada bagian rekap data anggota keluarga, ketika jumlah data bertambah dapat mempengaruhi kinerja aplikasi. Adapun data yang ditampilkan yaitu jumlah anggota keluarga, jumlah laki-laki, jumlah perempuan, jumlah yang sudah kawin, jumlah yang belum kawin, jumlah janda, jumlah duda, jumlah yang sudah bekerja, jumlah yang belum atau tidak bekerja, dan jumlah yang pendidikan terakhirnya (SD, SLTP, SLTA, Diploma, S1, S2, S3). Pada halaman awal, jumlah-jumlah tersebut berdasarkan kabupaten atau kota, jika ingin mengetahui lebih kecil lagi ruang lingkupnya seperti berdasarkan kecamatan, kelurahan/desa dan dasawisma, maka harus mengklik pada nama wilayah atau dasawisma. Tentu saja, dari pengambilan data yang ada pada semua kolom yang dipilih lalu masuk ke proses penghitungan berdasarkan suatu kondisi tertentu yang diinginkan, dapat menyebabkan tingginya beban pada sisi *server* dan *database* utama karena permintaan data yang semakin banyak. Semua data tersebut disimpan pada tabel yang berbeda-beda, sehingga untuk menampilkan rekap datanya pun harus mengambil dari tabel lainnya yang berkaitan dengan menggunakan operasi join yang terdapat di MySQL.

2. Studi Literatur

   Selanjutnya, Peneliti melakukan studi pustaka/literatur terkait hasil dari observasi tentang lambatnya pengambilan data dari *database* MySQL dan untuk memahami mengenai cara untuk mengatasi masalah yang dihadapi, seperti konsep-konsep dasar terkait dengan penggunaan redis sebagai *database caching* untuk mempercepat akses data relasional. Mempelajari tentang Redis, manfaatnya dalam meningkatkan kinerja akses data, dan bagaimana Redis dapat diintegrasikan dengan baik dengan *database* relasional. Informasi didapat dari jurnal maupun artikel media *online* dan juga terdapat pada dokumentasi resmi dari penyedia teknologi terkait.

Sehingga dari hasil pengumpulan data yang dilakukan diatas, dapat disimpulkan bahwa masalah yang muncul pada halaman rekap data adalah terjadinya kelambatan waktu respon yang dibutuhkan untuk memuat data sampai tampil penuh berdasarkan dengan jumlah data yang semakin banyak. Dan ketika ada *request* lagi, maka akan seterusnya lambat untuk memuat datanya.

Tentunya ini akan mengganggu aktifitas dari pengguna sistem dalam memonitor data. Sehingga akan memberikan rasa ketidaknyamanan dan akan menghambat pencapaian tujuan Dasawisma dalam pemantauan dan pelaporan yang efektif. Sehingga Dari beberapa poin-poin masalah yang didapatkan, maka peneliti menyimpulkan perlu melakukan proses *caching* datanya menggunakan Redis, guna meningkatkan kinerjanya dalam menampilkan data dengan jumlah yang semakin besar. Dikarenakan halaman ini merupakan informasi yang penting untuk ditampilkan ke pengguna mengenai hasil perhitungan beberapa *item* yang berdasarkan masing-masing wilayah maupun Dasawisma yang ada di Provinsi Sumatera Selatan.

3.1.3 Penentuan Kebutuhan

Selanjutnya, Peneliti melakukan perincian tentang penentuan kebutuhan berdasarkan pemahaman terhadap masalah yang telah diidentifikasi sebelumnya. Berikut ini pada **Tabel 3.1** merupakan spesifikasi *hardware* yang dibutuhkan:

Tabel 3.1 Spesifikasi *Hardware* yang dibutuhkan

No.	Item	Deskripsi
1.	Laptop	- *Toshiba Satellite L840* - *Memory: 8192 MB RAM* - *SSD: 256 GB* - *Processor: Intell Corel i5-3210M CPU @ 2.50 GHz (4 CPUs), ~ 2.5GHz*

Selain perangkat keras (*hardware*), terdapat juga beberapa perangkat lunak (*software*) yang juga diperlukan untuk penelitian ini, antara lain dapat dilihat pada **Tabel 3.2**

Tabel 3.2 Daftar *Software* yang dibutuhkan

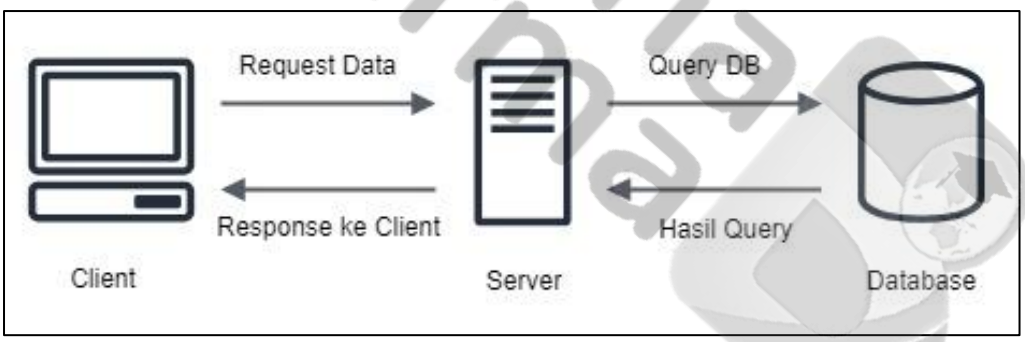
No.	*Software*	Kegunaan
1.	*Windows 10 Pro 64-bit*	Sebagai sistem operasi
2.	*Microsoft Office 2021*	Untuk pembuatan, pengolahan, dan dokumentasi data
3.	*Google Chrome*	Untuk melakukan studi literatur terkait informasi dan data yang dibutuhkan serta menguji sistem
4.	*Draw.io*	Untuk membuat *diagrams*, seperti ERD, *Flowchart* dan kerangka berpikir
5.	*Windows Terminal*	Untuk menjalankan *command* atau perintah sistem
6.	*Visual Studio Code*	Untuk melakukan pengkodingan sistem
7.	*Laravel Framework*	Sebagai kerangka kerja dalam membantu mengembangkan sistem berupa *website*
8.	*Laragon* - *Apache Httpd* - *MySQL* - *Redis*	Sebagai *tools* selama proses pengembangan dalam mode *development* atau *localhost*
9.	*Laravel Debugbar*	Untuk memantau hasil *response-time* yang dijalankan dari sebuah *query sql*
10.	*Biznet Gio*	Sebagai tempat *hosting/vps* dan pembelian *domain*
11.	*PuTTY*	Untuk melakukan koneksi ke *terminal server* via SSH (*Secure Shell*)

3.2 *Design* (Desain)

Pada tahap *Design,* peneliti merancang desain arsitektur sistem, merancang *caching* Redis, rancangan beberapa *diagram* seperti, *Entity Relationship Diagram* (ERD), *Sequence Diagram, flowchart* dan Skenario Pengujian. Berikut adalah langkah-langkah dari tahap *Design*.

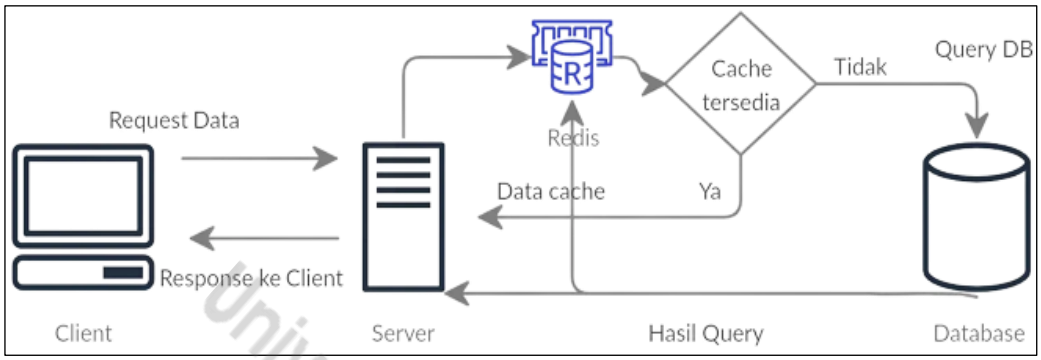
3.2.1 Arsitektur Sistem

Tahap Desain dimulai dengan merancang arsitektur sistem *caching* Redis. Ini mencakup pemilihan jenis arsitektur Redis yang sesuai. Dengan adanya arsitektur sistem ini diharapkan dapat mudah dipahami pada saat pengimplementasi Redis sebagai *caching* pada sistem Dasawisma PKK. Berikut ini adalah gambaran arsitektur antara sistem yang tanpa Redis *Cache* dan dengan Redis *Cache* yang dapat dilihat pada **Gambar 3.1**.



Gambar 3. 1 Arsitektur Sistem Tanpa Redis *Cache*

Pada **Gambar 3.1** menjelaskan bagaimana sebuah sistem secara umum bekerja, yaitu ketika ada request dari *client* ke *server* akan langsung diteruskan ke *database* berupa perintah *sql*/*nosql* yang kemudian hasilnya akan dikembalikan lagi ke *server* dan baru diterima oleh *client*. Pada kasus sistem dasawisma pkk provinsi sumatera selatan, alur proses seperti ini akan terus terjadi setiap terdapat request, sehingga menyebabkan beban kerja dari *server* dan *database* akan berat karena permintaan *request* yang semakin besar. Sedangkan jika sistem menerapkan redis sebagai *caching*, maka arsitektur sistemnya akan berbeda, yang dapat dilihat pada **Gambar 3.2**.



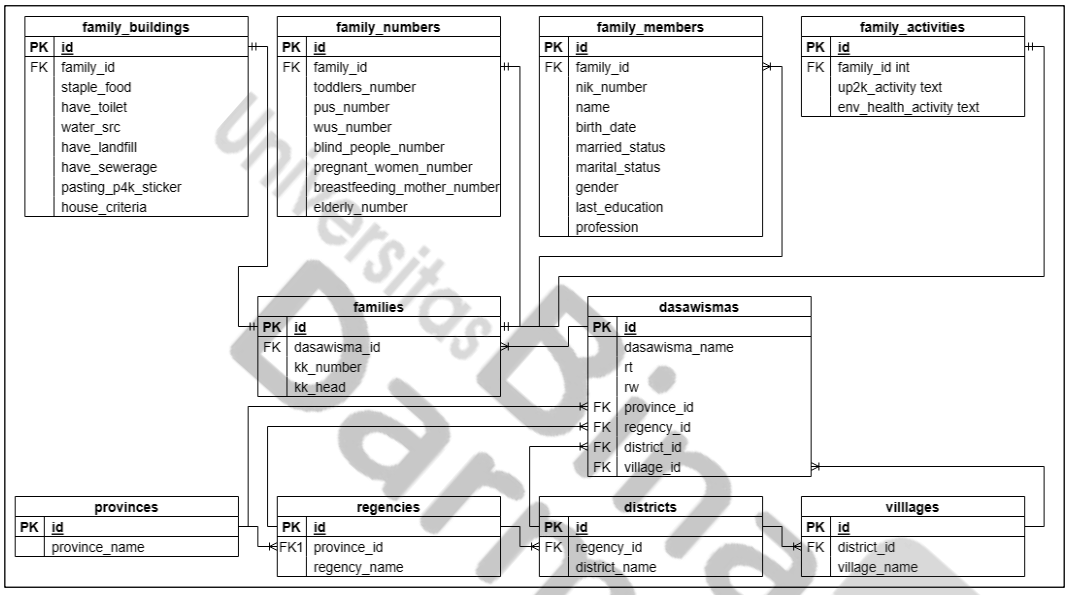
Gambar 3.2 Arsitektur Sistem

Sebuah sistem yang menerapkan redis sebagai *caching*, maka *request* pertama akan melakukan perintah dari *server* ke *database* utama akan tetapi saat terjadi lagi request yang sama, *server* akan melakukan pengecekan ke redis terlebih dahulu, jika data yang diminta tersedia di redis, maka *server* mengirim data tersebut ke *client*. Jika data yang diminta tidak tersedia di redis, maka *server* melakukan *querying* baru ke *database* utama (MySQL), setelah *database* mengembalikan data yang diminta, *server* lalu menyimpan data tersebut ke redis agar tersedia pada *request* yang akan datang dan meneruskan data tersebut ke *client*. Sehingga *response time* -nya lebih cepat dan *client* tidak menunggu terlalu lama.

3.2.2 *Entity Relationship Diagram* (ERD)

Rekap Data merupakan data berjenis relasional yang dibangun dari beberapa tabel yang saling berhubungan. Seperti rekap data info bangunan, rekap data jumlah anggota, rekap data anggota keluarga, dan rekap data aktivitas. Rekap ini dibangun dari banyak tabel yang saling berelasi sehingga jika dalam waktu yang bersamaan banyak *request* terhadap permintaan data ini maka *response time* yang diberikan oleh aplikasi dapat semakin lama. Oleh karena itu, rekap data yang dibentuk oleh *database* relasional akan ditranslasikan ke dalam struktur *in-memory database* untuk membantu aplikasi agar lebih cepat dalam memberikan informasi yang dibutuhkan kepada *user* lewat sistem *caching* data. Berikut pada **Gambar 3.3** merupakan

pemodelan data Dasawisma PKK provinsi Sumatera Selatan yang menampilkan rekap data berdasarkan wilayah maupun nama dasawisma dalam bentuk rancangan ERD.



Gambar 3. 3 ERD Rekap Data

Gambar 3.3 merupakan rancangan ERD yang dibutuhkan untuk menampilkan rekap data. Masing-masing tabel berelasi ke tabel lainnya. Berikut penjelasan lebih detail:

1. *provinces* dapat memiliki banyak *regencies* (*one to many*) sedangkan *regencies* dimiliki oleh satu *provinces* (*many to one*)
2. masing-masing *regencies* mempunyai banyak *districts* (*one to many*) sedangkan *districts* dimilki oleh satu *regencies* (*many to one*)
3. Masing-masing *districts* memiliki banyak *villages* (*one to many*) dan masing-masing *villages* hanya dimiliki oleh satu *districts* (*many to one*)
4. *provinces, regencies, districts, villages* dapat memiliki banyak *dasawismas* (*one to many*) sedangkan *dasawismas* dimiliki oleh *provinces, regencies, districts,* dan *villages* (*many to one*)
5. *dasawismas* dapat memiliki banyak *families* (*one to many*), sedangkan *families* hanya dimiliki oleh satu *dasawismas* (*many to one*)

6. *families* memiliki satu *family_buildings* (*one to one*), *family_numbers* (*one to one*), *family_members* (*one to many*), *family_activities* (*one to one*).

Berikut ini merupakan penjelasan struktur tabel yang sesuai dari *Entity Relationship Diagram* sebelumnya.

Tabel 3.3 Struktur Tabel *provinces*

Nama Kolom	Tipe Data	Atribut
id	*bigint*	*primary key*
*province_*name	*varchar(16)*	*not null*

Tabel 3.3 digunakan untuk menyimpan nama provinsi.

Tabel 3.4 Struktur Tabel *regencies*

Nama Kolom	Tipe Data	Atribut
id	*bigint*	*primary key*
province_id	*bigint*	*foreign key, not null*
regency_name	*varchar(100)*	*not null*

Tabel 3.4 digunakan untuk menyimpan nama-nama kabupaten/kota.

Tabel 3.5 Struktur Tabel *districts*

Nama Kolom	Tipe Data	Atribut
id	*bigint*	*primary key*
regency_id	*bigint*	*foreign key, not null*
district_name	*varchar(100)*	*not null*

Tabel 3.5 digunakan untuk menyimpan nama-nama kecamatan.

Tabel 3. 6 Struktur Tabel *villages*

Nama Kolom	Tipe Data	Atribut
id	*bigint*	*primary key*
district_id	*bigint*	*foreign key, not null*
village_name	*varchar(100)*	*not null*

Tabel 3.6 digunakan untuk menyimpan nama-nama kelurahan/desa.

Tabel 3. 7 Struktur Tabel *dasawismas*

Nama Kolom	Tipe Data	Atribut
id	*bigint(1)*	*primary key*
name	*varchar(100)*	*not null*
rt	*char(3)*	*not null*
rw	*char(3)*	*not null*
province_id	*bigint*	*foreign key, not null*
regency_id	*bigint*	*foreign key, not null*
district_id	*bigint*	*foreign key, not null*
village_id	*bigint*	*foreign key, not null*

Tabel 3.7 digunakan untuk menyimpan nama-nama dasawisma yang ada pada suatu kelurahan atau desa.

Tabel 3.8 Struktur Tabel *families*

Nama Kolom	Tipe Data	Atribut
id	*bigint*	*primary key*
dasawisma_id	*bigint*	*foreign key, not null*
kk_number	*varchar(100)*	*not null*
kk_head	*varchar(100)*	*not null*

Tabel 3.8 digunakan untuk menyimpan nama-nama kepala keluarga.

Tabel 3. 9 Struktur Tabel *family_buildings*

Nama Kolom	Tipe Data	Atribut
id	*bigint*	*primary key*
family_id	*bigint*	*foreign key, not null*
staple_food	*enum*	*null*
water_src	*varchar(100)*	*not null*
have_toilet	*boolean*	*not null*
have_landfill	*boolean*	*not null*
have_sewerage	*boolean*	*not null*
pasting_p4k_sticker	*boolean*	*not null*
house_criteria	*enum*	*not null*

Tabel 3.9 digunakan untuk menyimpan informasi data bangunan miik seorang kepala keluarga.

Tabel 3. 10 Struktur Tabel *family_numbers*

Nama Kolom	Tipe Data	Atribut
id	*bigint*	*primary key*
family_id	*bigint*	*foreign key, not null*
toddlers_number	*int(20)*	*not null, default(0)*
pus_number	*int(20)*	*not null, default(0)*
wus_number	*int(20)*	*not null, default(0)*
blind_people_number	*int(20)*	*not null, default(0)*
pregnant_women_number	*int(20)*	*not null, default(0)*
breastfeeding_mother_number	*int(20)*	*not null, default(0)*
elderly_number	*enum*	*not null, default(0)*

Tabel 3.10 digunakan untuk menyimpan informasi jumlah anggota dalam sebuah keluarga.

Tabel 3. 11 Struktur Tabel *family_members*

Nama Kolom	Tipe Data	Atribut
id	*bigint*	*primary key*
family_id	*bigint*	*foreign key, not null*
nik_number	*varchar(18)*	*null*
name	*varchar(100)*	*not null*
slug	*varchar(110)*	*unique, not null*
birth_date	*date*	*not null*
status	*enum*	*not null*
marital_status	*enum*	*not null*
gender	*enum*	*not null*
last_education	*enum*	*not null*
profession	*varchar(255)*	*not null*

Tabel 3.11 digunakan untuk menyimpan informasi nama-nama dan informasi pribadi anggota keluarga yang dimiliki oleh seorang kepala keluarga.

Tabel 3. 12 Struktur Tabel *family_activities*

Nama Kolom	Tipe Data	Atribut
id	*bigint*	*primary key*
family_id	*bigint*	*foreign key, not null*
up2k_activity	*text*	*null*
env_health_activity	*text*	*null*

Tabel 3.12 digunakan untuk menyimpan informasi data kegiatan yang dilakukan sebuah keluarga.

Berdasarkan dari beberapa tabel-tabel sebelumnya, misalnya ketika ingin mendapatkan rekap data anggota keluarga berdasarkan wilayah tertentu maka harus mengkalkulasi beberapa *attributes/fields* yang terdapat pada tabel *family_members*. Tabel ini juga berlasi ke tabel *families*, tabel

families berelasi ke tabel *dasawismas*, jika yang ingin diambil adalah hasil kalkulasi berdasarkan kabupaten/kota maka dari tabel *dasawismas* harus berelasi lagi ke tabel *provinces* dan dari tabel *provinces* harus berelasi ke tabel *regencies* serta digabung dengan beberapa operasi maupun operator lain sesuai dengan yang dibutuhkan dalam menampilkan rekap data.

Selain struktur tabel pada *database* utama, disajikan pula struktur tabel pada redis yang digunakan untuk menyimpan data *cache*. Dalam pengelolaan *cache*, terutama ketika menggunakan sistem *caching* seperti Redis, menetapkan konvensi nama *key* dari masing-masing *value* adalah praktik yang dapat membantu dalam pengorganisasian dan pengelolaan data dalam *cache*. Jangka waktu data tersimpan adalah 1 jam sebelum pada akhirnya dihapus secara otomatis ketika sudah melewati TTL. Adapun untuk rancangan struktur data *cache* di redis dapat dilihat pada **Gambar 3.4**.

Redis		
TTL	**Key**	**Value**
1h	data-recap:[area]:{fb/fn/fm/fa}:regencies:page-[number]	data_value
1h	data-recap:[area]:{fb/fn/fm/fa}:districts:by-regency:[regency]:page-[number]	data_value
1h	data-recap:[area]:{fb/fn/fm/fa}:villages:by-district:[district]:page-[number]	data_value
1h	data-recap:[area]:{fb/fn/fm/fa}:dasawismas:by-village:[village]:page-[number]	data_value
1h	data-recap:[area]:{fb/fn/fm/fa}:family-heads:by-dasawisma:[dasawisma]:page-[number]	data_value

Gambar 3.4 Rancangan Struktur Data *Cache* di Redis

Informasi yang disimpan dalam redis antara lain:

1. *'data-recap:[area]:{fb/fn/fm/fa}:regencies:page-[number]'* merupakan sebuah penanda untuk data yang menyimpan hasil dari *query* sql terhadap rekap data yang ada di level kabupaten/kota berdasarkan area dan nomor halaman. Lama waktu disimpannya adalah 1 jam.
2. '*data-recap:[area]:{fb/fn/fm/fa}:districts:by-regency:[regency]:page:[number]*' adalah sebuah penanda untuk data yang menyimpan hasil dari *query* sql terhadap rekap data yang ada di level kecamatan berdasarkan area dan nomor halaman. Lama waktu disimpannya adalah 1 jam.

3. '*data-recap:[area]:{fb/fn/fm/fa}:villages:by-district:[district]:page-[number]*' adalah sebuah penanda untuk data yang menyimpan hasil dari *query* sql terhadap rekap data yang ada di level kelurahan/desa yang berdasarkan area dan nomor halaman. Lama waktu disimpannya 1 jam.
4. '*data-recap:{fb/fn/fm/fa}:dasawismas:by-village:[village]:page-[number]*' adalah sebuah penanda untuk data yang menyimpan hasil dari *query* sql rekap data yang ada dilevel dasawisma berdasarkan area dan nomor halaman. Lama waktu disimpannya adalah 1 jam.
5. *'data-recap:{fb/fn/fm/fa}:family-heads:by-dasawisma:[dasawisma]:page-[number]*' adalah sebuah penanda untuk data yang menyimpan hasil dari query sql terhadap rekap data yang ada dilevel keluarga berdasarkan nama dasawisma dan nomor halaman. Lama waktu disimpannya adalah 1 jam.

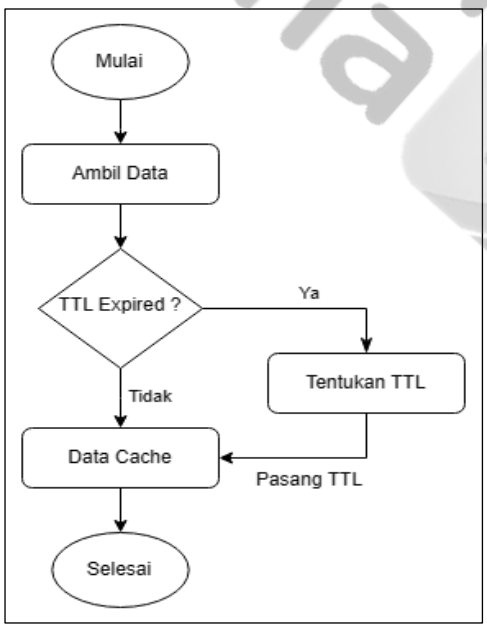
Jenis *value* yang disimpan adalah dalam bentuk tipe data *strings* dan *hashes*. Berbeda dengan *database* relasional, *in-memory database* tidak mengenal relasi antar data sehingga data dasawisma tersebut dimodelkan dengan cara lain melalui skema *key-value*. Walaupun begitu data dasawisma yang meliputi perhitungan dari nama *attributes* yang di ambil tetap dapat ditampilkan melalui aplikasi. Contoh operasi yang dapat digunakan dalam redis, adalah *hSet* dan *hGet*. *hSet* membutuhkan sebuah *key* yang unik untuk dapat menyimpan sebuah data. Data ini dapat diubah dan dihapus berdasarkan *key* tersebut. Untuk menampilkan data yang sudah tersimpan, dibutuhkan operasi *hGet.* Setiap *key* tersebut dapat diberikan lama waktu disimpan menggunakan operasi *expire* sehingga keberadaannya di dalam redis dapat diatur. Sehingga operasi *hSet* dan *hGet* sering digunakan untuk operasi *Create Read Update Delete* (CRUD) dalam implementasi *in-memory database* untuk aplikasi berbasis *web*.

In-memory database dalam penelitian ini tidak menjadi *database* utama. Keberadaannya hanya sebagai *database* pendamping yang berfungsi

sebagai *cache* sehingga jika ada *request* data yang sama tidak harus dimuat ulang dari *database* utama, cukup dimuat dari redis dengan waktu respon yang jauh lebih cepat.

3.2.3 Rancangan Redis Sebagai *Caching*

Rancangan Redis sebagai *caching* adalah tahap yang penting untuk memastikan bahwa Redis *caching* dapat diimplementasikan dengan sukses. Diharapkan dapat membantu dalam pemahaman lebih baik tentang alur kerja dan proses yang terlibat dalam implementasi Redis sebagai sistem *caching* data. Ini memungkinkan peneliti atau pengembang untuk memiliki pandangan yang jelas tentang bagaimana Redis akan digunakan untuk mencapai tujuan kinerja dalam implementasi Redis sebagai *caching*. Pada **Gambar 3.5** merupakan diagram alir atau *flowchart* untuk proses *caching* data



Gambar 3.5 *FlowChart Caching* Data

Peneliti memprioritaskan jenis data yang harus di *cache* adalah data pada halaman rekap data. Dikarenakan ini merupakan data yang sering diakses oleh pengguna sistem untuk mengetahui jumlah suatu item tertentu berdasarkan suatu wilayah atau dasawisma. Rancangan dari peneliti yang

akan dilakukan adalah ketika *client* mengakses sebuah *route* rekap data, sebelum mengambil datanya dari *database* utama (MySQL), dilakukan pengecekan terlebih dahulu menggunakan *key* yang telah ditentukan pada redis apakah datanya ada. Jika ada, maka ambil data dari redis, jika tidak ada maka lanjut ambil data ke *database* utama (MySQL) dan juga simpan data balikannya pada redis. Ketika *client* selanjutnya ingin mengakses *route* GET itu lagi selama belum ada perubahan data, maka akan mengambil datanya di redis, bukan lagi dari *database* MySQL. Lalu bagaimana jika terjadi perubahan data? misal ada penambahan data baru, pengubahan data, atau dihapus? Yang sistem akan lakukan adalah menghapus semua/spesifik data yang berkaitan pada redis dan biarkan nanti melakukan *caching* ulang dari MySQL ke Redis jika *user* kembali mengakses *route* rekap data. Data yang terdapat dalam *cache* akan tetap ada, selama waktu simpan yang telah ditentukan. Untuk waktu penyimpanan atau TTL (*Time-to-Live*) adalah konfigurasi yang menentukan berapa lama data akan tetap ada dalam redis sebelum dihapus. Disini pengembang atau peneliti menentukan bahwa waktu TTL akan di setel selama 1 jam kedepan. Berikut ini merupakan alur dari pengambilan, penyimpanan dan penghapusan data berdasarkan TTL yang telah ditentukan.

3.2.4 Skenario Pengujian

Pada tahap ini dijelaskan skenario pengujian sistem setelah pengimplementasian redis sebagai *caching* dengan jumlah data yang selalu bertambah sebanyak 200 Ribu dan dilakukan sebanyak 3 kali *request* dari sisi *client*. Data yang disimpan pada redis menggunakan 2 jenis tipe data pada redis, yaitu *strings* dan *hashes*.

Skenarion pengujian pertama dilakukan tanpa menggunakan redis. Dimulai dari request pertama sampai ketiga yang kemudian akan diteruskan ke sisi *server* dan dilanjutkan ke *database* MySQL. Data kembaliannya akan diteruskan kembali atau ditampilkan ke *client*.

Selanjutnya, Skenario pengujian kedua dilakukan dengan redis menggunakan tipe data *strings*. Dimulai ketika ada *request* dari sisi *client*

sesuai dengan jumlah data yang diminta, kemudian dilanjutkan pada sisi *server* akan melakukan pengecekan nama *key* yang sudah ditentukan sebelumnya ke redis, jika *key* -nya ada, maka data akan diambil. Sedangkan jika *key* yang dimaksud tidak ada pada redis, maka *server* akan mengirim perintah SQL ke database utama (MySQL). Setelah diproses oleh MySQL, data balikannya disimpan pada redis dalam bentuk json dan harus menyertakan berapa lama datanya akan disimpan. Sehingga *request* pertama akan diambil dari *database* utama, sedangkan *request* kedua dan seterusnya akan mengambil ke redis selama belum ada perubahan data dan lama waktu disimpan masih ada. Dan untuk melihat berapa lama waktu yang dibutuhkan untuk memproses *request*-nya yaitu menggunakan bantuan sebuah *library* dari Laravel yang bernama Laravel Debugbar.

Pada skenario pengujian ketiga kurang lebih sama dengan skenario kedua, yang membedakannya adalah data balikan dari proses *request* pertama, di MySQL akan disimpan dalam bentuk tipe data *hashes* (pasangan *key-value*) ke redis. Sehingga tidak perlu melalui proses *encode* maupun *decode* karena hasil balikan dari *query* SQL yang berbentuk *array* bisa langsung disimpan ke dalam redis dan ditentukan lama waktu simpannya adalah 1 jam.

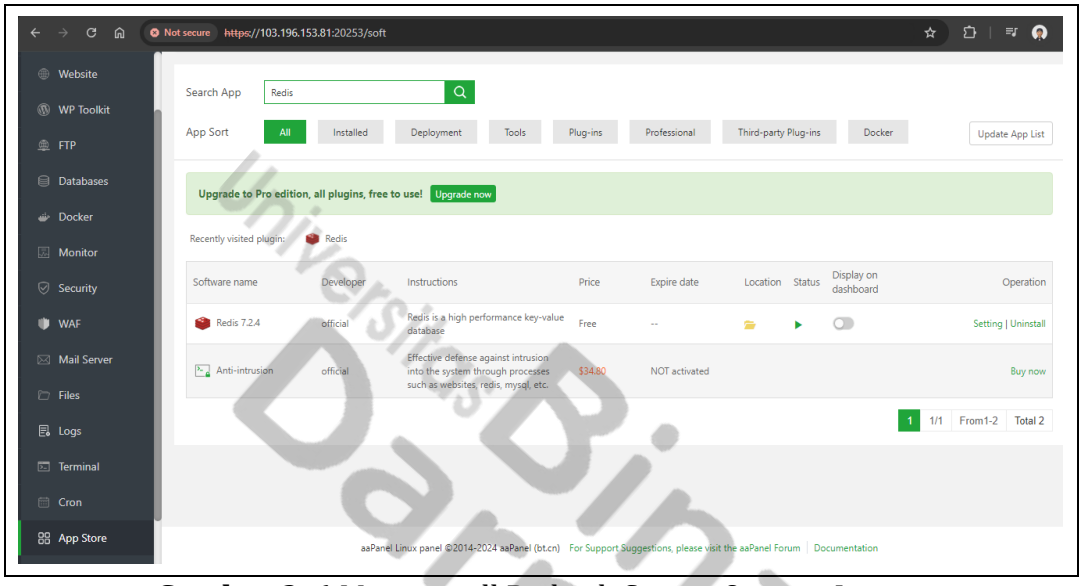
3.3 *Development* (Pengembangan)

Pada tahap *Development* berisi proses pengembangan sistem berupa penginstalan redis, konfigurasi dan pengkodean redis. Berikut adalah langkah-langkahnya.

3.3.1 Instalasi Redis Pada Sistem Operasi

Penginstallan Redis pada sistem operasi *Linux Ubuntu* 22.04 dilakukan dengan menggunakan layanan *Virtual Private Server* dari Biznet Gio, sesuai dengan spesifikasi pada bagian penentuan kebutuhan. Dan untuk memudahkan proses instalasi, pengembang menggunakan salah satu layanan

panel yang bernama aaPanel sebagai tempat untuk mengontrol sistem *linux server*. Berikut ini pada **Gambar 3.6** adalah langkah untuk instalasi Redis.

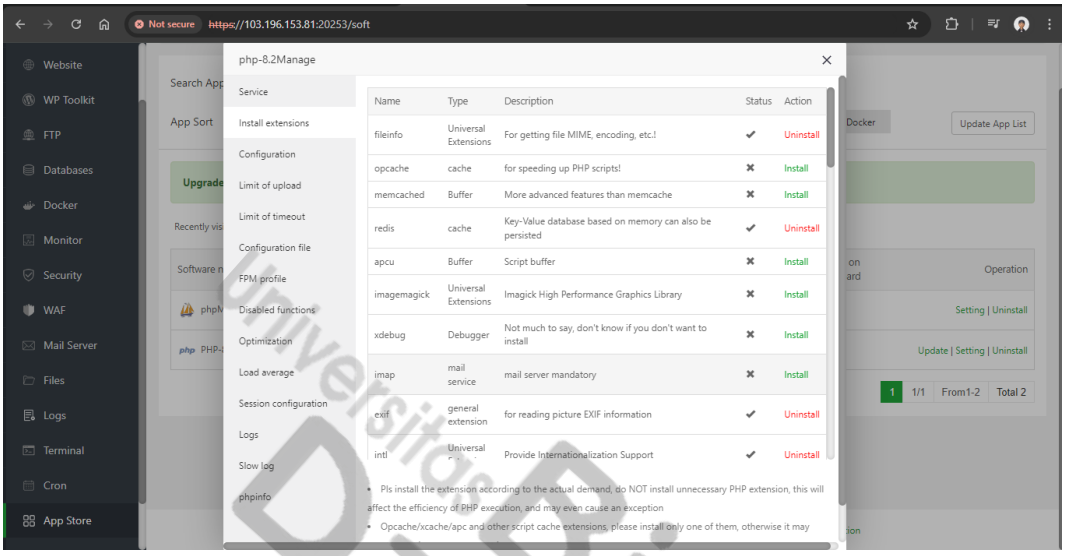


Gambar 3. 6 Menginstall Redis di Sistem Operasi Linux

Pada bagian menu sebelah kiri, masuk ke *App Store* lalu cari kata kunci redis sehingga akan muncul *package* redis. Dilanjutkan dengan mengeklik tombol *install* dan tunggu sampai prosesnya selesai.

3.3.2 Instalasi *Extension* Redis Pada PHP

Langkah selanjutnya adalah proses instalasi redis pada bahasa pemrograman PHP yang dapat dilihat pada **Gambar 3.7**. Masih sama dengan sebelumnya, pengembang menggunakan layanan aaPanel untuk membantu proses instalasi agar lebih mudah.

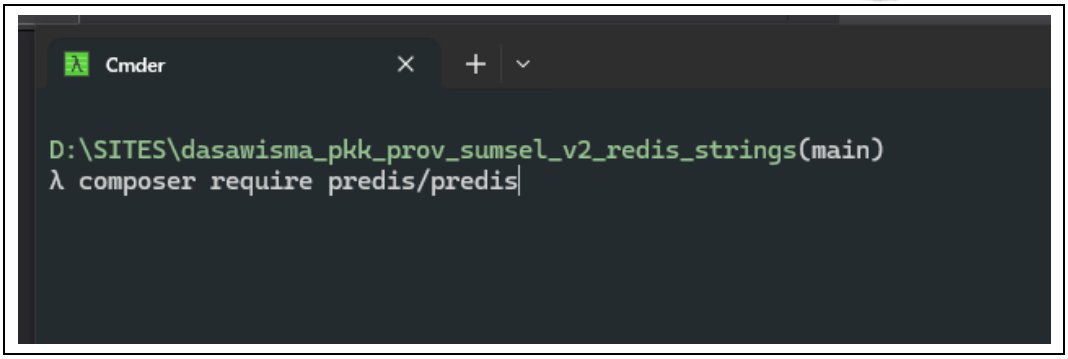


Gambar 3. 7 Menginstall Redis di PHP

Pada halaman pengaturan PHP, klik pada menu *install extensions* lalu lanjutkan dengan mengklik install pada ekstensi redis dan tunggu proses instalasi sampai dengan selesai.

3.3.3 Instalasi *Library* Redis Pada Laravel

Sistem dibuat dengan menggunakan *framework* laravel, Tentunya perlu juga untuk menginstall *library* redis pada laravel agar aplikasi bisa berkomunikasi dengan redis *server*.



Gambar 3. 8 Instalasi *Library* Redis Pada Laravel

Perintah *composer require predis/predis* pada **Gambar 3.8** dijalankan pada terminal yang mengarah ke folder projek. Ini merupakan perintah untuk menginstall *package predis* sebagai redis *client* pada projek. *Composer*

itu sendiri adalah *package manager* yang dimiliki oleh bahasa pemrograman PHP.

3.3.4 Konfigurasi Redis Pada Laravel

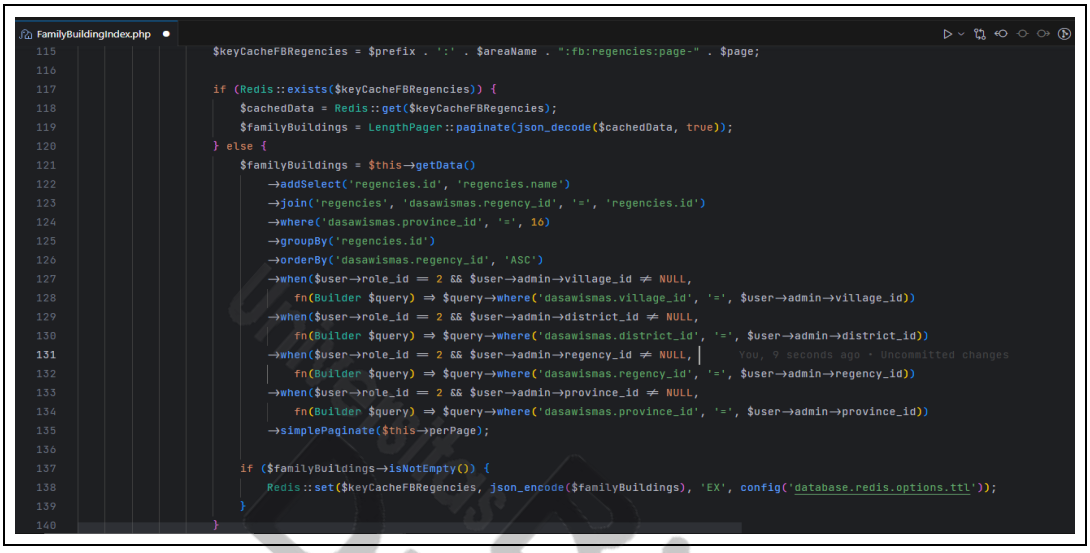
Langkah selanjutnya adalah melakukan konfigurasi antara redis dengan projek laravel pada file *config/database.php*. Di dalam file ini, terdapat *array* dengan *key* yang bernama redis yang berisi konfigurasi *server* redis yang dibutuhkan oleh aplikasi agar bisa berkomunikasi. **Gambar 3.9** merupakan konfigurasi redis pada laravel

Gambar 3. 9 Isi File Konfigurasi Redis Pada Laravel

Laravel akan membaca terlebih dahulu konfigurasi pada file *.env* yang terletak pada *root* projek, jika *variable*-nya ada maka akan diambil nilainya sedangkan jika tidak maka akan mengambil *default* konfigurasinya pada file *config/database.php*.

3.3.5 Pengkodean Redis ke Kode Program

Langkah selanjutnya jika semua proses installasi sudah selesai adalah pengkodean fitur redis sebagai *caching* data sebelum mengambil datanya dari *database* utama. Pengkodeannya dapat dilihat pada **Gambar 3.10**.



Gambar 3. 10 Pengkodean Redis Sebagai *Caching*

Gambar 3.10 merupakan kodingan redis ke kode program laravel Untuk bagian rekap data bangunan ketika diakses. Adapun penjelasannya adalah sebagai berikut. Terdapat *variable $keyCacheFbRegencies* yang berisi nama *key* yang ada di redis. Lalu dilakukan pengecekan menggunakan *if statement*. Jika *key* yang dicari ditemukan atau kondisinya bernilai benar, maka kode dalam blok *if statement* akan dijalankan. Di dalam blok *if statement* berisi pemanggilan data ke redis berdasarkan *key* yang telah di inisialisasi sebelumnya.

Sebaliknya, Jika *key* yang dicari tidak ada pada redis, maka kode dalam blok *else statement* akan dieksekusi. Di dalam blok *else* berisi pemanggilan terhadap fungsi *getData()* dan dilakukan operasi lain seperti penambahan operasi *select*, tambahan *query join* ke tabel lainnya, *where condition*, *groupBy*, dan *orderBy*. Setelah itu, data balikan akan dimasukkan ke dalam *database* redis menggunakan perintah *Redis::set(key, value, expirationTime)*. Selanjutnya data hasil dari blok *if* yang kondisinya terpenuhi atau *else* yang tidak terpenuhi akan dilempar pada halaman. Jika dilihat kodenya, terdapat pemanggilan fungsi *getData()*. Adapun isi fungsinya dapat dilihat pada **Gambar 3.11**.

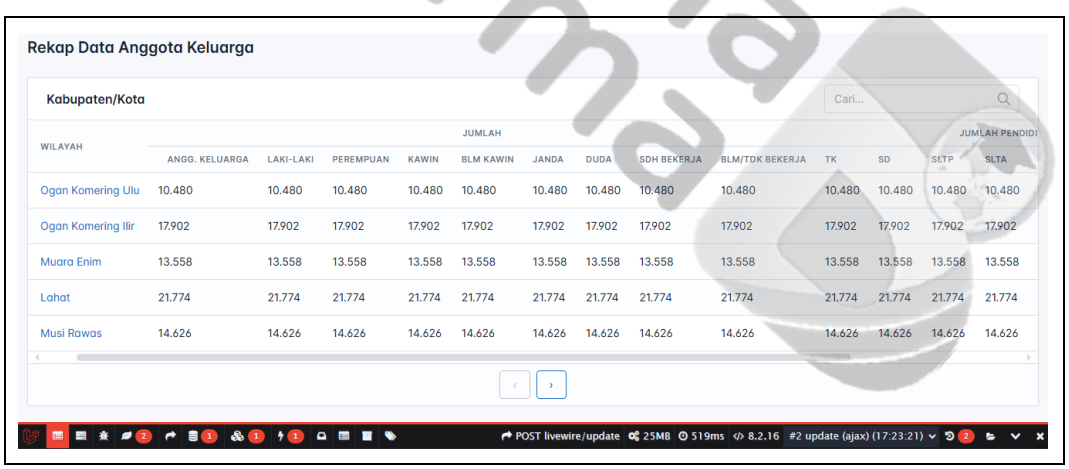
Gambar 3. 11 Isi Fungsi *getData()*

Fungsi *getData()* berisi perhitungan penjumlahan pada kolom-kolom yang dipilih menggunakan *method COUNT* yang dimiliki oleh MySQL dan pemanggilan *method join* untuk berelasi ke tabel-tabel lain yang dibutuhkan. Untuk lebih jelasnya dalam proses pengkodean redis ke kode program, dapat dilihat pada bagian **Lampiran 1**, **2**, dan **3.**

BAB IV
HASIL DAN PEMBAHASAN

4.1 Implementasi (Hasil Pengujian)

Tahap implementasi dilakukan pengujian sistem untuk memastikan bahwa redis sanggup dalam menangani data dalam jumlah besar. Pengujian dilakukan dengan jumlah data yang dimulai dari 200 Ribu sampai dengan 5 Juta dan dilakukan sebanyak tiga kali *request*. Jenis tipe data pada redis yang diuji adalah *strings* dan *hashes*. hasil yang diharapkan dengan menggunakan redis sebagai sistem *caching* adalah dapat meningkatkan kinerja sistem lebih cepat. **Gambar 4.1** merupakan halaman rekap data yang diuji.



Gambar 4. 1 Halaman Pengujian (Jumlah data 200 Ribu)

4.1.1 Pengujian Sistem Tanpa Redis

Gambar 4.1 merupakan tampilan awal pada halaman rekap data anggota keluarga, dengan jumlah data sebanyak 200 Ribu yang melewati proses perhitungan. Untuk melihat tampilan pada jumlah data dengan kenaikan 200 Ribu berikutnya dapat dilihat pada **Lampiran 4, 5, 6, 7** dan **8.** Pada **Tabel 4.1** merupakan hasil waktu respon yang dibutuhkan dari *request* ke-1, 2 dan 3 serta perhitungan rata-rata waktu respon.

Tabel 4. 1 Hasil Pengujian Sistem Tanpa Redis

Jumlah Data	Request ke-1 (s)	Request ke-2 (s)	Request ke-3 (s)	Rata-rata Waktu Respon
200 Ribu	1,13	0,825	0,735	0,90
400 Ribu	1,67	1,70	1,65	1,67
600 Ribu	7,74	4,70	3,85	5,43
800 Ribu	31,05	21,04	17,90	23,33
1 Juta	39,61	30,4	24,44	31,48
1,2 Juta	57,59	41,50	39,66	46,25
1,4 Juta	74,40	59,21	55,08	62,90
1,6 Juta	85,02	69,70	66,25	73,66
1,8 Juta	94,36	77,84	70,45	80,88
2 Juta	103,88	82,66	80,36	88,97
2,2 Juta	116,91	91,49	91,73	100,04
2,4 Juta	117,47	99,54	97,68	104,90
2,6 Juta	126,25	117,32	108,54	117,37
2,8 Juta	134,38	128,89	117,63	126,97
3 Juta	158,98	153,86	145,72	152,85
3,2 Juta	167,66	150,97	142,93	153,85
3,4 Juta	179,04	161,67	153,12	164,61
3,6 Juta	191,01	173,92	164,89	176,61
3,8 Juta	201,61	185,34	175,94	187,63
4 Juta	211,04	195,76	185,99	197,60
4,2 Juta	221,48	206,73	196,37	208,19
4,4 Juta	231,31	217,47	206,11	218,30
4,6 Juta	241,75	228,15	216,28	228,73
4,8 Juta	252,98	239,76	227,15	239,96
5 Juta	264,43	251,62	238,37	251,47

Hasil pengujian sebelum mengimplementasikan redis dibutuhkan waktu yang lama untuk memuat data yang semakin besar.

4.1.2 Pengujian Sistem Dengan Redis (Tipe Data *Strings*)

Tabel 4. 2 Hasil Pengujian Redis dengan Jenis Tipe Data *Strings*

Jumlah Data	Request ke-2 (s)	Request ke-3 (s)	Request ke-4 (s)	Rata-rata Waktu Respon
200 Ribu	0,430	0,453	0,464	0,449
400 Ribu	0,439	0,467	0,481	0,462
600 Ribu	0,485	0,519	0,459	0,488
800 Ribu	0,465	0,509	0,477	0,484
1 Juta	0,502	0,477	0,483	0,487
1,2 Juta	0,475	0,409	0,466	0,450
1,4 Juta	0,505	0,452	0,475	0,478
1,6 Juta	0,512	0,429	0,473	0,471
1,8 Juta	0,514	0,439	0,481	0,478
2 Juta	0,526	0,457	0,476	0,486
2,2 Juta	0,530	0,440	0,476	0,482
2,4 Juta	0,545	0,456	0,481	0,494
2,6 Juta	0,548	0,453	0,480	0,494
2,8 Juta	0,557	0,461	0,482	0,500
3 Juta	0,534	0,462	0,481	0,493
3,2 Juta	0,552	0,463	0,484	0,500
3,4 Juta	0,553	0,470	0,485	0,503
3,6 Juta	0,551	0,471	0,485	0,503
3,8 Juta	0,551	0,476	0,486	0,504
4 Juta	0,552	0,478	0,487	0,506
4,2 Juta	0,558	0,482	0,488	0,509
4,4 Juta	0,556	0,485	0,489	0,510
4,6 Juta	0,557	0,488	0,490	0,512
4,8 Juta	0,559	0,492	0,491	0,514
5 Juta	0,560	0,495	0,492	0,516

Berdasarkan **Tabel 4.2**, untuk hasil *request* pertama tidak dimasukkan, karena data yang di *cache* belum tersimpan pada redis, sehingga dari sisi *server* akan melakukan perintah pertamanya ke *database* utama, yaitu MySQL.

4.1.3 Pengujian Sistem Dengan Redis (Tipe Data *Hashes*)

Sama seperti sebelumnya, **Gambar 4.1** digunakan sebagai tampilan pengujian untuk data sebanyak 200 Ribu. Dan untuk tampilan pada jumlah data yang kenaikannya 200 Ribu berikutnya dapat dilihat pada bagian **Lampiran 4, 5, 6, 7** dan **8.** Berikut pada **Tabel 4.3** merupakan hasil waktu respon yang dibutuhkan dari *request* kedua, ketiga dan keempat serta dilanjutkan dengan perhitungan rata-rata respon waktunya.

Tabel 4. 3 Hasil Pengujian Sistem Dengan Redis *Hashes*

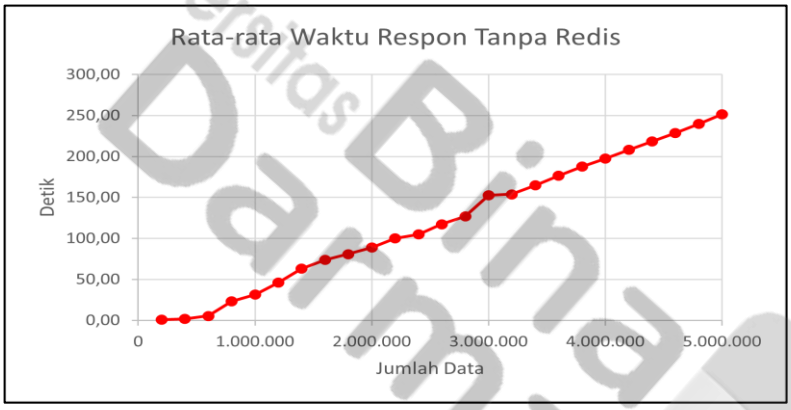
Jumlah Data	Request ke-2 (s)	Request ke-3 (s)	Request ke-4 (s)	Rata-rata Waktu Respon
200 Ribu	0,450	0,478	0,455	0,461
400 Ribu	0,435	0,437	0,459	0,444
600 Ribu	0,466	0,513	0,468	0,482
800 Ribu	0,502	0,476	0,47	0,483
1 Juta	0,437	0,445	0,473	0,452
1,2 Juta	0,472	0,484	0,471	0,476
1,4 Juta	0,476	0,474	0,478	0,476
1,6 Juta	0,480	0,478	0,481	0,480
1,8 Juta	0,475	0,464	0,482	0,474
2 Juta	0,473	0,473	0,485	0,477
2,2 Juta	0,488	0,478	0,488	0,485
2,4 Juta	0,484	0,471	0,492	0,482
2,6 Juta	0,486	0,472	0,493	0,484
2,8 Juta	0,488	0,473	0,496	0,486
3 Juta	0,493	0,478	0,499	0,490
3,2 Juta	0,495	0,475	0,502	0,491
3,4 Juta	0,495	0,475	0,505	0,492
3,6 Juta	0,499	0,477	0,507	0,495
3,8 Juta	0,502	0,478	0,510	0,497
4 Juta	0,504	0,479	0,513	0,499
4,2 Juta	0,506	0,479	0,516	0,500
4,4 Juta	0,509	0,485	0,518	0,504
4,6 Juta	0,511	0,484	0,521	0,506
4,8 Juta	0,514	0,486	0,524	0,508
5 Juta	0,516	0,488	0,527	0,510

Pada **Tabel 4.3**, Untuk *request* pertama tidak dimasukkan karena data belum ada di redis, sehingga *server* mengirim perintah ke *database* MySQL.

4.2 Evaluasi

Setelah pengujian, Tahap ini merupakan informasi hasil rata-rata waktu respon yang disajikan salam bentuk grafik.

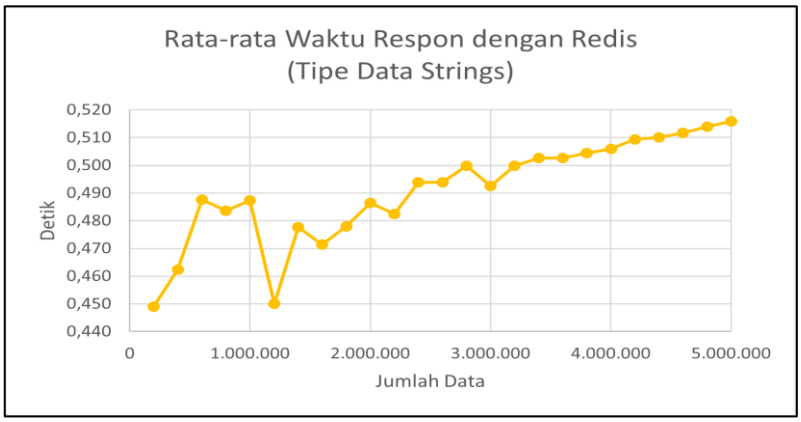
1. Grafik Rata-rata Waktu Respon Jika Tanpa Redis



Gambar 4. 2 Grafik Rata-rata Waktu Respon Sistem Jika Tanpa Redis

Berdasarkan grafik pada **Gambar 4.2**, terlihat bahwa waktu respon selalu meningkat seiring dengan jumlah data yang bertambah banyak, dikarenakan balikan pada MySQL tidak ditangani dengan redis sehingga menyebabkan proses pada *server* dan *database* utama akan besar.

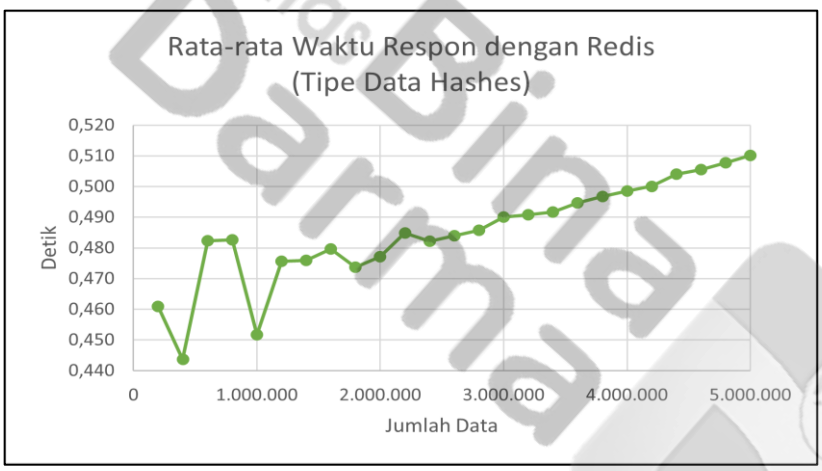
2. Grafik Rata-rata Waktu Respon Sistem Jika dengan Redis *Strings*



Gambar 4. 3 Grafik Rata-rata Waktu Respon Sistem Jika Dengan Redis *Strings*

Terlihat pada **Gambar 4.3**, bahwa terjadi penurunan waktu respon yang didapat dalam melakukan pengambilan data yang ketika tanpa menggunakan redis dibutuhkan waktu lebih dari 1 detik dan ketika sudah menggunakan redis hasilnya kurang dari 1 detik. Ini dikarenakan datanya diambil langsung ke memori, pengambilan data ke memori lebih cepat dari pada ke jenis *storage* lain.

3. Grafik Rata-rata Waktu Respon Sistem Jika dengan Redis *Hashes*



Gambar 4. 4 Grafik Rata-rata Waktu Respon Sistem Jika Dengan Redis *Hashes*

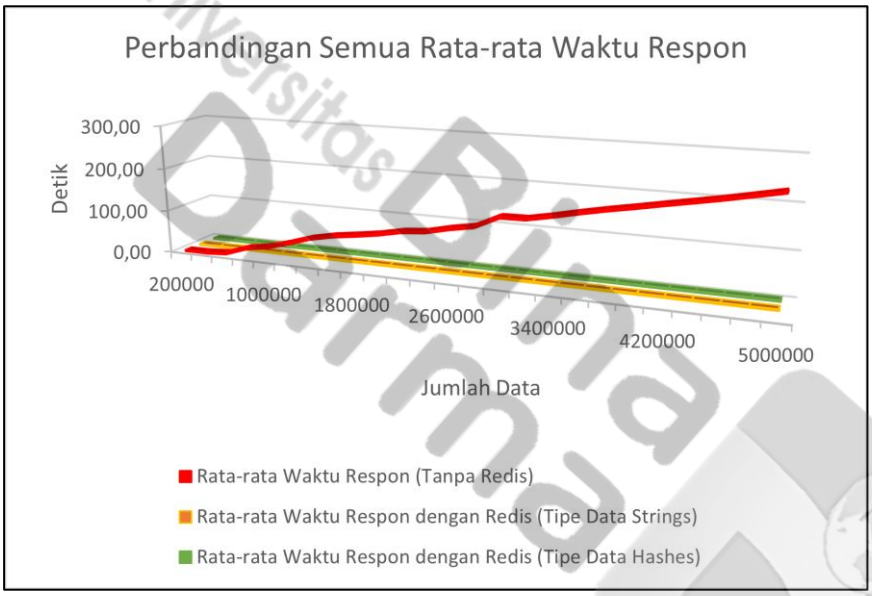
Terlihat juga pada **Gambar 4.4**, bahwa dengan menggunakan redis dengan tipe data *hashes* juga bisa menghasilkan waktu respon yang tidak kalah cepat dan tentunya lebih cepat dari tanpa menggunakan redis.

Secara keseluruhan dari tiga grafik sebelumnya, menunjukkan bahwa dengan mengimplementasi Redis dapat meningkatkan waktu respon sistem dalam memuat data besar, terutama untuk operasi baca sehingga bisa menjaga performa sistem tetap baik seiring pertambahan data. Berikut ini pada **Tabel 4.4**, merupakan tabel ringkasan rata-rata waktu respon yang dibutuhkan sistem dalam memuat data berdasarkan jumlah data yang tersedia.

Tabel 4. 4 Hasil Semua Waktu Respon

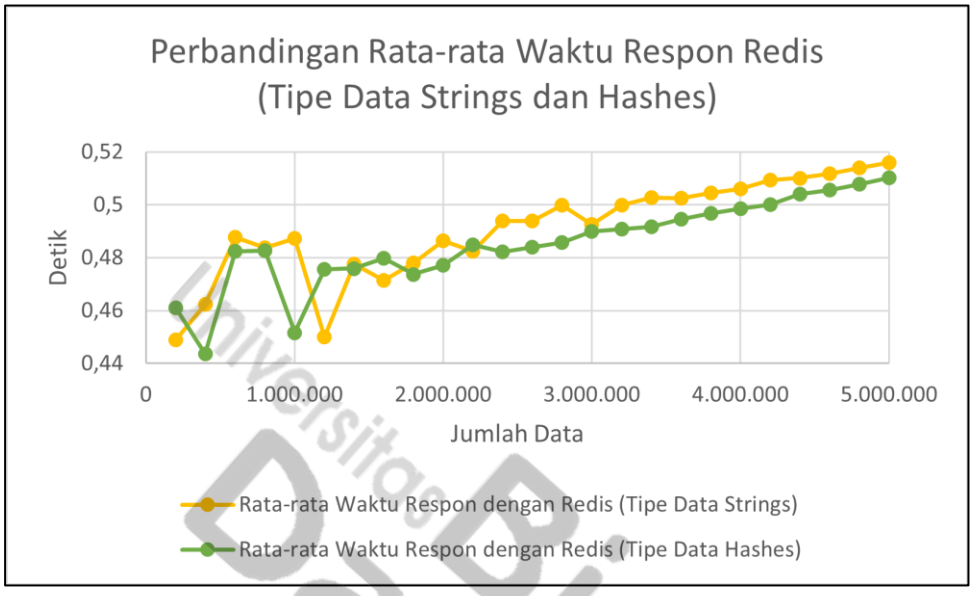
Jumlah Data	Rata-rata Waktu Respon (Tanpa Redis)	Rata-rata Waktu Respon (Dengan Redis *Strings*)	Rata-rata Waktu Respon (Dengan Redis *Hashes*)
200 Ribu	0,90	0,449	0,461
400 Ribu	1,67	0,462	0,444
600 Ribu	5,43	0,488	0,482
800 Ribu	23,33	0,484	0,483
1 Juta	31,48	0,487	0,452
1,2 Juta	46,25	0,450	0,476
1,4 Juta	62,90	0,478	0,476
1,6 Juta	73,66	0,471	0,480
1,8 Juta	80,88	0,478	0,474
2 Juta	88,97	0,486	0,477
2,2 Juta	100,04	0,482	0,485
2,4 Juta	104,90	0,494	0,482
2,6 Juta	117,37	0,494	0,484
2,8 Juta	126,97	0,500	0,486
3 Juta	152,85	0,493	0,490
3,2 Juta	153,85	0,500	0,491
3,4 Juta	164,61	0,503	0,492
3,6 Juta	176,61	0,503	0,495
3,8 Juta	187,63	0,504	0,497
4 Juta	197,60	0,506	0,499
4,2 Juta	208,19	0,509	0,500
4,4 Juta	218,30	0,510	0,504
4,6 Juta	228,73	0,512	0,506
4,8 Juta	239,96	0,514	0,508
5 Juta	251,47	0,516	0,510

Berdasarkan **Tabel 4.4**, Dapat dilihat hasil pada kolom ke-1, rata-rata waktu respon yang didapat selalu tinggi atau lambat karena sistem belum mengimplementasikan redis. Sedangkan pada 2 kolom terakhir, dengan mengimplementasikan redis, bisa menghasilkan rata-rata waktu respon yang sangat signifikan turunnya atau cepat dan selalu stabil *response time*-nya. Jika dibuat dalam 1 grafik maka akan tampil seperti pada **Gambar 4.5**.



Gambar 4. 5 Grafik Perbandingan Semua Waktu Respon

Dari hasil akhir dari perbandingan waktu respon yang ditampilkan pada satu grafik menunjukkan bahwa jika tanpa menggunakan redis ketika data bertambah banyak, maka waktu yang di butuhkan untuk memuat data akan selalu tinggi atau lama.



Gambar 4. 6 Grafik Perbandingan Waktu Respon Tipe Data *Strings* dengan *Hashes*

Sedangkan jika sistem telah mengimplementasikan redis, baik itu dengan tipe data *strings* atau *hashes*, hasil perbandingan rata-rata waktu respon dapat dilihat pada **Gambar 4.6**, yang artinya walaupun data bertambah banyak, hasil rata-rata waktu respon pada *request* ke-2 dan seterusnya selama belum ada perubahan data hanya dibutuhkan waktu yang singkat dan tentunya stabil dalam memuat jumlah data yang semakin besar. Manakah yang lebih baik antara tipe data *strings* dengan *hashes* untuk kasus masalah ini?. jawabannya adalah pada kasus penelitian ini, tipe data *hashes* mempunyai kinerja yang cepat.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari hasil pengujian dan evaluasi mengenai waktu respon yang didapat dalam memuat data, dapat disimpulkan bahwa dengan menggunakan redis untuk jumlah data sebanyak 5 juta diperoleh rata-rata waktu respon 0,516 detik untuk jenis struktur data *strings*, dan 0,510 detik untuk struktur data *hashes*. Ini lebih cepat dibandingkan tanpa menggunakan redis diperoleh rata-rata waktu respon 251,47 detik. Maka terjadi penurunan sebesar 99,7%. Yang artinya terdapat peningkatan kinerja sebesar 489,05%. Redis cocok untuk digunakan khususnya ketika butuh untuk proses *caching* data. Dan jika dilihat perbandingan dua respon waktu dari dua penggunaan tipe data pada redis untuk sistem *caching*, tipe data *hashes* adalah yang paling efisien, dikarenakan pada saat pengambilan datanya langsung bisa ditampilkan dalam bentuk paginasi karena respon data pada redisnya sudah berupa *array* sehingga waktu yang dibutuhkan untuk memuat datanya walaupun dengan *volume* yang semakin besar, rata-rata waktu yang dihasilkan tetap stabil. Sedangkan untuk tipe data *strings* harus diubah dulu dari *json* ke bentuk *array* agar bisa diproses untuk dipaginasi. Dan rata-rata waktu yang dihasilkan berdasarkan *volume* data yang terus bertambah pun tetap bisa mendapatkan nilai yang stabil.

Dengan demikian, jika redis salah satunya digunakan sebagai solusi untuk masalah pada suatu sistem yang sudah mengalami penurunan kinerjanya, ini sangat cocok jika di implementasikan dengan benar. Dan tidak lupa untuk selalu memantau kapasitas memori yang masih tersedia guna memastikan redis bisa bekerja secara optimal.

5.2 Saran

Pada Sistem Dasawisma PKK Provinsi Sumatera Selatan ini yang di implementasikannya Redis sebagai *cache* masih terdapat beberapa kekurangan, sehingga diharapkan dapat dikembangkan lagi. Adapun yang disarankan oleh peneliti yaitu, Untuk sistem sekarang hanya menggunakan 2 GB RAM, Sehingga jika nanti batas maksimal RAM sudah penuh maka perlu adanya penambahan ukuran kapasitas RAM pada sisi *server* agar data yang ditampung dengan limit besar bisa tersimpan dengan baik. Dan jika pada sisi *database* atau kode program masih bisa dilakukan normalisasi dan optimasi sangat disarankan untuk dibuat lebih efisien lagi, Sehingga performanya akan lebih meningkat juga.

DAFTAR PUSTAKA

Anugraha, N., Angriawan, R., & Mashud, M. (2020). Sistem Informasi Geografis Layanan Publik Lingkup Kota Makassar Berbasis Web. *DoubleClick: Journal of Computer and Information Technology*, *4*(1), 35. https://doi.org/10.25273/doubleclick.v4i1.6073

Aziz, & Yaya Mulyana Abdul. (2019). Model Kebijakan Peningkatan Laporan Kematian dalam Administrasi Kependudukan dan Catatan Sipil di Kabupaten Bandung Barat. *Sosiohumaniora*, *19*(2), 140–148.

Baig, M. I., Shuib, L., & Yadegaridehkordi, E. (2019). Big data adoption: State of the art and research challenges. *Information Processing & Management*, *56*(6), 102095.

Budiraharjo, J., Milisani, M., & Marosani, Y. (2022). PENGARUH PROSES REKRUTMEN DAN SELEKSI TERHADAP KUALITAS KERJA PEGAWAI BADAN PERENCANAAN PEMBANGUNAN DAERAH DKI JAKARTA. *Jurnal Manajemen*.

Febriyani, F., Pramukantoro, E. S., & Bakhtiar, F. A. (t.t.). *Perbandingan Kinerja Redis, Mosquitto, dan MongoDB sebagai Message Broker pada IoT Middleware*.

Frisdayanti, A. (2019). *PERANAN BRAINWARE DALAM SISTEM INFORMASI MANAJEMEN*. *1*.

Gunadi, F., & Widianto, S. R. (2020). Evaluasi Kualitas Pelaporan Manajemen pada Sistem Epicor Perusahaan Manufaktur Berbasis McCall. *MULTINETICS*, *6*(1), 21–31. https://doi.org/10.32722/multinetics.v6i1.2765

Hadi, S. M., & Samad, A. (2019). Sistem Informasi Pengolahan Data Bantuan Beasiswa Siswa Miskin (BSM) Pada Kantor Wilayah Kementerian Agama Provinsi Maluku Utara. *Jurnal Ilmiah ILKOMINFO - Ilmu Komputer & Informatika*, *2*(1). https://doi.org/10.47324/ilkominfo.v2i1.15

Hasibuan, S. H., & Nasution, M. I. P. (2023). *A Comparative Study of Relasional and NoSQL database for Big Data Analytics*. *2*(3).

Hermiati, R., Asnawati, A., & Kanedi, I. (2021). PEMBUATAN E-COMMERCE PADA RAJA KOMPUTER MENGGUNAKAN BAHASA PEMROGRAMAN PHP DAN DATABASE MYSQL. *JURNAL MEDIA INFOTAMA*, *17*(1). https://doi.org/10.37676/jmi.v17i1.1317

Irawan, D., & Hidayat, A. T. (2019). *RANCANG BANGUN DASHBOARD KEPEGAWAIAN SEKOLAH TINGGI ILMU EKONOMI MUSI RAWAS (STIE MURA) LUBUKLINGGAU*. *9*.

Irawan, G. H., & Ramdhani, N. (t.t.). *Analisa Performa Cache Database Redis dan MySQL dengan PHP*.

Isfahani, F. A., & Nugraha, F. (t.t.). *Implementasi Load Balancing NGINX dan Mongodb Cluster serta Mekanisme Redis Caching*.

Khoshkholgh, M. G., Navaie, K., Shin, K. G., Leung, V. C. M., & Yanikomeroglu, H. (2019). Caching or No Caching in Dense HetNets? *2019 IEEE Wireless Communications and Networking Conference (WCNC)*, 1–7. https://doi.org/10.1109/WCNC.2019.8885724

Lozada, N., Arias-Pérez, J., & Perdomo-Charry, G. (2019). Big data analytics capability and co-innovation: An empirical study. *Heliyon*, *5*(10), e02541. https://doi.org/10.1016/j.heliyon.2019.e02541

Meriani, A. P., Asih, D. R., & Annisa, S. (t.t.). *Pengujian Distributed Cached Database Dengan Menggunakan Redis Pada Aplikasi MaBaUS*.

Pasha, D., Priandika, A. T., & Indonesian, Y. (2020). ANALISIS TATA KELOLA IT DENGAN DOMAIN DSS PADA INSTANSI XYZ MENGGUNAKAN COBIT 5. *Jurnal Ilmiah Infrastruktur Teknologi Informasi*, *1*(1), 7–12. https://doi.org/10.33365/jiiti.v1i1.268

Prabowo, I. A., Pomalingo, S., Istiono, W., & Muhariya, A. (t.t.). *SISTEM KOMPUTER DAN INFORMASI*.

Putra, M. W., Darwis, D., & Priandika, A. T. (2021). Pengukuran Kinerja Keuangan Menggunakan Analisis Rasio Keuangan Sebagai Dasar

Penilaian Kinerja Keuangan (Studi Kasus: CV Sumber Makmur Abadi Lampung Tengah). *Jurnal Ilmiah Sistem Informasi Akuntansi*, *1*(1), 48–59. https://doi.org/10.33365/jimasia.v1i1.889

Ramadhan, W. F., Dewi, W. N., & Nas, C. (2020). APLIKASI WEB PORTAL MANAJEMEN INFORMATIKA BERBASIS WEBSITE DENGAN MENGGUNAKAN FRAMEWORK CODEIGNITER DAN MYSQL PADA UNIVERSITAS CATUR INSAN CENDEKIA. *Jurnal Digit*, *10*(2), 124. https://doi.org/10.51920/jd.v10i2.164

Rochman, A., Hanafri, M. I., & Wandira, A. (2020). Implementasi Website Profil SMK Kartini Sebagai Media Promosi dan Informasi Berbasis Open Source. *Academic Journal of Computer Science Research*, *2*(1). https://doi.org/10.38101/ajcsr.v2i1.272

Rohaeni, S. (2020). PENGEMBANGAN SISTEM PEMBELAJARAN DALAM IMPLEMENTASI KURIKULUM 2013 MENGGUNAKAN MODEL ADDIE PADA ANAK USIA DINI. *Instruksional*, *1*(2), 122. https://doi.org/10.24853/instruksional.1.2.122-130

Rosad, A. M. (2019). IMPLEMENTASI PENDIDIKAN KARAKTER MELALUI MANAGEMEN SEKOLAH. *Tarbawi: Jurnal Keilmuan Manajemen Pendidikan*, *5*(02), 173. https://doi.org/10.32678/tarbawi.v5i02.2074

Ruliah, D., Kom, M., Suryadi, A., Kom, S., & Kom, M. (t.t.). *Basis Data dan Sistem Basis Data*.

Somya, R., & Nathanael, T. M. E. (2019). PENGEMBANGAN SISTEM INFORMASI PELATIHAN BERBASIS WEB MENGGUNAKAN TEKNOLOGI WEB SERVICE DAN FRAMEWORK LARAVEL. *Jurnal Techno Nusa Mandiri*, *16*(1), 51–58. https://doi.org/10.33480/techno.v16i1.164

Sumantri, R. B. B., Universitas Harapan Bangsa, Setiawan, R. A., & Sandi A, A. S. (2022). SISTEM INFORMASI GEOGRAFIS UNTUK PEMETAAN PARIWISATA KABUPATEN KARANGANYAR BERBASIS WEB.

METHOMIKA Jurnal Manajemen Informatika dan Komputerisasi Akuntansi, *6*(1), 1–9. https://doi.org/10.46880/jmika.Vol6No1.pp1-9

Suprayogi, A., Guna, N. S., & Darmawan, F. R. (t.t.). *IMPLEMENTASI DAN ANALISA PERFORMA DATABASE CACHE REDIS MENGGUNAKAN DIGITAL*.

Yulianti, H. (2021). Pemanfaatan Sistem Pelatihan E-Learning Pada Pengembangan Kinerja Karyawan di Masa Pandemi Covid-19 Dengan Pengujian ISO 9126. *MULTINETICS*, *7*(1), 65–81. https://doi.org/10.32722/multinetics.v7i1.3769

Zulfa, M. I., Fadli, A., & Wardhana, A. W. (2019). *DESAIN MODEL VIEW CONTROLLER DALAM IMPLEMENTASI REDIS SERVER UNTUK MEMPERCEPAT AKSES DATA RELASIONAL*.

Zulfa, M. I., Fadli, A., & Wardhana, A. W. (2020). Application caching strategy based on in-memory using Redis server to accelerate relational data access. *Jurnal Teknologi dan Sistem Komputer*, *8*(2), 157–163. https://doi.org/10.14710/jtsiskom.8.2.2020.157-163

EXPLORE
Jurnal Sistem Informasi dan Telematika
(Telekomunikasi, Multimedia dan Informatika)

Implementasi Redis Untuk Mempercepat Pengambilan Data (Studi Kasus: Sistem Dasawisma PKK Sumsel)

Dwi Robbi Prasetyo, Fatoni*, Muhammad Nasir, Irman Effendy
Jurusan Teknik Informatika, Fakultas Sains Teknologi, Universitas Bina Darma
Palembang, Indonesia
[1]dwirobbin@gmail.com, [2]fatoni@binadarma.ac.id*, [3]nasir@binadarma.ac.id, [4]irman.effendy@binadarma.ac.id

Abstract-Having quick access to the latest information is essential as the need for information continues to increase. The South Sumatra Province PKK Dasawisma System is used as a place to collect data such as information about Dasawisma programs, residents, and activities. Over time the data will be very large, so a way to handle it (Big data) is needed. In data management, especially when displaying data recaps on the Dasawisma PKK dashboard page, there are often obstacles in terms of the speed of relational operations that affect system responsiveness. To overcome these obstacles, this research proposes the implementation of an in-memory database (Redis) as a solution to speed up relational operations in displaying data recaps. Redis is a memory-based database that enables fast data storage and efficient access. ADDIE, which stands for Analysis, Design, Development, Implementation, and Evaluation, is the development process used in this research. The Analysis stage is carried out by determining the time and place of research, collecting data, and determining needs. At the design stage, activities to design technical designs were carried out. Furthermore, the development stage is carried out implementing redis cache. Then, the implementation stage is tested to measure the speed comparison between before and after redis implementation. The last stage of evaluation, which is analyzing the test result data obtained, by comparing the performance of the system when before and after the implementation of redis. obtained, by comparing system performance when before and after the implementation of redis cache. redis cache implementation. The purpose of this research is to speed up process of retrieving data from the database even though the volume of data continues to growing large. This research resulted in an average response time for the request and so on only takes a short time in loading an increasingly large amount of data as long as there has been no change in the data volume. data as long as there are no data changes.

Keywords: Data, Relational operations, In-memory Database, ADDIE, Redis.

Abstrak-Memiliki akses cepat ke informasi terbaru sangat penting karena kebutuhan akan informasi yang terus meningkat. Sistem Dasawisma PKK Provinsi Sumatera Selatan digunakan sebagai tempat untuk mengumpulkan data seperti informasi tentang program Dasawisma, penduduk, dan kegiatan. Seiring waktu data akan sangat besar, Sehingga diperlukan sebuah cara untuk menanganinya (*Big data*). Dalam pengelolaan data, terutama saat menampilkan rekap data pada halaman *dashboard* Dasawisma PKK, seringkali ditemui kendala dalam hal kecepatan operasi relasional yang mempengaruhi responsifitas sistem. Untuk mengatasi kendala tersebut, penelitian ini mengusulkan implementasi *in-memory database* (Redis) sebagai solusi untuk mempercepat operasi relasional dalam menampilkan rekap data. Redis merupakan basis data berbasis memori yang memungkinkan penyimpanan data secara cepat dan akses yang efisien. ADDIE adalah singkatan dari *Analysis, Design, Development, Implementation, and Evaluation*, merupakan proses pengembangan yang digunakan dalam penelitian ini. Tahap Analisis dilakukan dengan menentukan waktu dan tempat penelitian, melakukan pengumpulan data, dan penentuan kebutuhan. Pada tahap desain, dilakukan kegiatan merancang desain teknis. Selanjutnya tahap pengembangan dilakukan pengimplementasian redis *cache*. Kemudian, tahap implementasi dilakukan pengujian untuk mengukur perbandingan kecepatan antara sebelum dan setelah implementasi redis. Tahap terakhir evaluasi, yaitu menganalisis data hasil pengujian yang diperoleh, dengan membandingkan kinerja sistem ketika sebelum dan setelah implementasi redis *cache*. Tujuan dari penelitian ini adalah untuk mempercepat proses pengambilan data dari *database* meskipun *volume* data terus bertambah besar. Penelitian ini menghasilkan rata-rata waktu respon untuk *request* ke-2 dan seterusnya hanya dibutuhkan waktu yang singkat dalam memuat jumlah data yang semakin besar selama belum ada perubahan data.

Kata Kunci: Data, Operasi relasional, *in-memory Database*, ADDIE, Redis.

1. Pendahuluan

Kemudahan akses internet ke berbagai informasi, menyebabkan sebagian besar aktivitas manusia selalu bersentuhan dengan teknologi saat ini [1]. Sistem Dasawisma PKK Provinsi Sumatera Selatan digunakan sebagai tempat untuk mengumpulkan data seperti informasi tentang program Dasawisma, penduduk, dan kegiatan. Jumlah penduduk Provinsi Sumatera Selatan berdasarkan Sensus Penduduk tahun 2022 sebesar 8,657 juta jiwa. Dasawisma PKK adalah kelompok PKK yang terdiri dari 10 hingga 20 rumah/kepala keluarga di tingkat RT. Tiga tanggung jawab utama kader Dasawisma adalah mendata warga, mengorganisir, dan menyebarluaskan informasi. Pemerintah Provinsi Sumatera Selatan akan menggunakan data yang telah diperolehnya untuk mengoptimalkan layanan masyarakat dan melakukan berbagai intervensi. Memelihara data statistik kependudukan sangat penting untuk mencegah terjadinya kesalahan, salah satunya dalam distribusi bantuan sosial [2].

Permasalahan yang terjadi pada sistem Dasawisma PKK Provinsi Sumatera Selatan adalah terjadinya kelambatan pengambilan data yang akan ditampikan pada sisi *client* dengan jumlah data yang seiring waktu terus bertambah banyak dikarenakan kompleksitas operasi relasional yang tinggi. Operasi relasional yang kompleks berisi, seperti penggabungan data dari beberapa tabel, dan penghitungan data berdasarkan kondisi tertentu menyebabkan terjadinya kendala performa yang signifikan dalam menampilkan data pada sistem *dashboard (web admin)*. *Dashboard (Web Admin)* yang juga dikenal sebagai portal admin *web* adalah sebuah sistem yang digunakan *administrator* untuk memasok dan mengelola data dari informasi berbasis *web* di suatu lembaga organisasi atau perusahaan [3]. Setiap bagian dari data memiliki ruangnya sendiri di situs *web* [4]. Sedangkan sistem, adalah kumpulan bagian yang saling terkait yang bekerja sama untuk mencapai satu tujuan [5]. Sehingga diperlukan sebuah cara baru agar dapat mempercepat proses pengambilan data dari *database*. Salah satunya adalah dengan mengimplementasikan redis.

Menurut Harsono (Dalam [6]), mendefinisikan implementasi sebagai "pertumbuhan kegiatan yang saling menyesuaikan, yang membutuhkan jaringan pelaksana dan birokrasi yang efisien, dan interaksi antara tujuan dan tindakan untuk mencapainya". Sehingga, implementasi dapat didefinisikan sebagai proses menempatkan ide, protokol, atau serangkaian langkah baru ke dalam tindakan dengan harapan bahwa orang lain akan mengadopsinya dan membuat penyesuaian yang diperlukan untuk mengubahnya menjadi tujuan yang dapat dicapai dengan jaringan pelaksana yang dapat dipercaya.

Salvatore Sanfilippo menciptakan Redis (*Remote Dictionary Server*), sebuah basis data berbasis *key-value* yang diliris pada 10 Mei 2009 [7]. Redis adalah *database open-source* yang memiliki keuntungan dapat diakses secara cepat karena penyimpanan *dataset* berbasis memori. Hal ini membuat redis sangat cocok digunakan dalam aplikasi yang membutuhkan *response-time* rendah, seperti sistem *caching* data dalam jumlah yang besar. Alasannya adalah karena kecepatan dan responsifitas yang sangat cepat. Memiliki fleksibilitas dalam menyimpan berbagai jenis data. Dan tentu saja manajemen data *cache* yang sangat fleksibel dan efisien [8].

Tindakan menyimpan sementara teks, gambar, atau materi lain di situs *web* untuk menghemat pemuatan dan lalu lintas *server* dikenal sebagai *cache*. Sederhananya, adalah teknik yang memfasilitasi tampilan halaman *web* yang lebih cepat. Karena data yang ada di dekatnya dapat disalin melalui *cache* [9]. *Caching* terdiri dari dua jenis, yaitu *caching* pada sisi *client* dan sisi *server*. *Caching* sisi *server* terjadi di *server*, sedangkan *caching* sisi *client* terjadi di komputer pengguna dan dapat dikontrol melalui pengaturan *browser*.

Pengembangan projek dilakukan menggunakan bantuan *framework* laravel yang menggunakan arsitektur *Model View Controller* (MVC). Laravel adalah *framework* yang dibuat menggunakan bahasa PHP yang dirancang untuk membuat aplikasi. Definisi *framework* adalah sebuah kerangka kerja yang memfasilitasi pembuatan dan pengembangan perangkat lunak oleh pengembang perangkat lunak [10]. *Framework* Laravel dibuat oleh Taylor Otwell, dirilis di bawah lisensi sumber terbuka pada 9 Juli 2011. Artinya, tidak perlu membayar untuk menggunakannya.

Menurut Teknovasi (Dalam [11]) Kueri gabungan akan memerlukan waktu lebih lama untuk dieksekusi apabila semakin banyak tabel yang digabungkan. Operasi-operasi ini dapat membutuhkan waktu yang lama, terutama ketika data yang harus diolah dalam jumlah besar. Sedangkan menurut Luthfi *Database* relasional menyimpan datanya di dalam *disk* (HDD/SSD). Permasalahan tersebut dapat menjadi penyebab keterbatasan performa. *Disk* memiliki keterbatasan dalam kecepatan akses data, memperlambat waktu eksekusi *query* yang kompleks dan penarikan data dari *database*. Akibatnya, responsifitas sistem menurun dan waktu pemrosesan data menjadi lebih lama.

Untuk mengatasi kendala performa ini, penggunaan *database in-memory* seperti Redis menjadi relevan. Dalam beberapa tahun terakhir, Redis telah menjadi solusi yang populer untuk mengatasi masalah performa ini. Redis adalah *database open-source* yang memanfaatkan memori sebagai media penyimpanan data, yang memungkinkannya untuk memberikan akses langsung ke memori sehingga dapat memberikan kinerja yang jauh lebih cepat dalam pengolahan data dibandingkan dengan *database* relasional berbasis *disk*. Redis juga mendukung banyak struktur data, seperti *strings, hashes, sets, lists, sorted sets* [12].

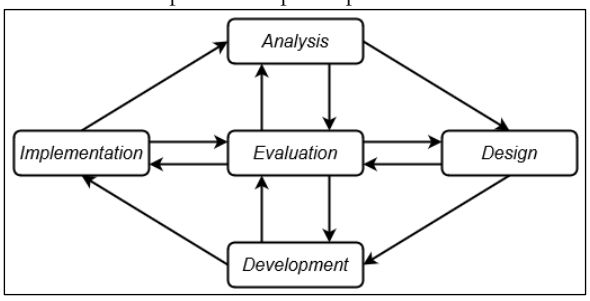
Terdapat beberapa penelitian terdahulu yang juga menjadi sumber referensi. Seperti penelitian yang dilakukan oleh Mulki Indana Zulfa, Ari Fadli, dan Arief Wisnu Wardhana, misalnya, mempelajari bagaimana menggunakan server Redis untuk solusi *caching* aplikasi dalam memori yang mempercepat akses data relasional. Yang menghasilkan bahwa redis dapat membantu meningkatkan kecepatan sistem MySQL hingga 1,7 kali, mengurangi waktu yang diperlukan untuk operasi *join* guna menampilkan data mahasiswa dan waktu yang diperlukan untuk membaca data IPK dari 61 milidetik menjadi 52 mikrodetik [13]. Penelitian selanjutnya adalah dari Faisal Al Isfahani dan Fuji Nugraha yang berjudul Implementasi *Load Balancing* NGINX dan MongoDB *Cluster* serta Mekanisme Redis *Caching* yang bertujuan

untuk membangun arsitektur sistem terdistribusi dengan menggabungkan penyeimbangan beban *server web*, kluster basis data, dan teknologi *caching* yang memanfaatkan redis [14]. Yang ketiga adalah penelitian yang dilakukan oleh Anggi Putri Meriani, Dwi Ramti Asih dan Siti Annisa, berjudul Pengujian *Distributed Cached Database* Dengan Menggunakan Redis Pada Aplikasi MaBaUS. Menghasilkan bahwasanya distributed *cache* dapat mempercepat *request* terhadap data ataupun informasi yang akan di distribusikan. Hal tersebut berdasarkan angka kecepatan yang dihasilkan dari tahap pengujian yaitu terdapat kecepatan sebesar 570ms dengan tambahan 1,18 ms untuk aplikasi MaBaUS yang tidak menggunakan *cache*, dan kecepatan 560 ms untuk Aplikasi MaBaUS yang menggunakan *cache* [12].

Dengan demikian, dalam penelitian ini ingin mengimplementasikan Redis sebagai *database in-memory* dan mengujinya, yang diharapkan dapat mengatasi kendala performa yang ada dan meningkatkan kecepatan dalam mengambil atau menampilkan rekap data, sehingga dapat memberikan manfaat dalam meningkatkan responsifitas sistem, efisiensi pengambilan keputusan, dan memungkinkan pengguna untuk memperoleh informasi secara cepat dan akurat.

2. Metodologi

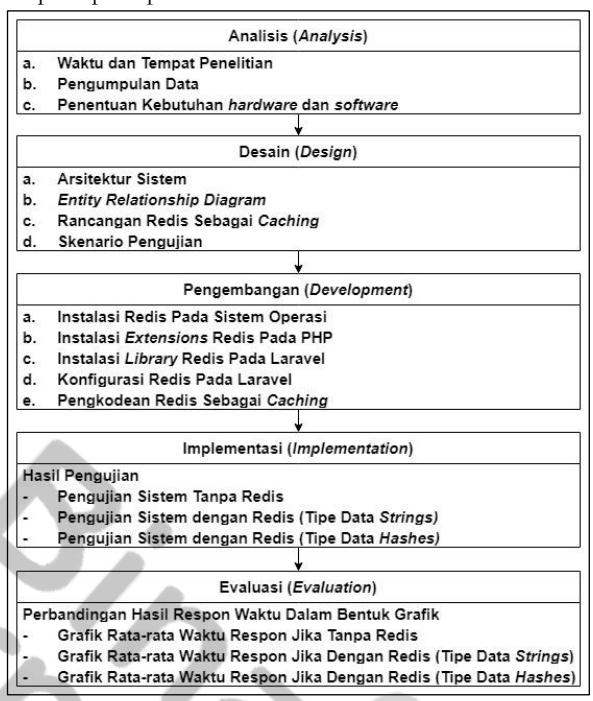
Penelitian ini akan menggunakan metode ADDIE. Berikut ini merupakan tahap-tahap metode ADDIE.



Gambar 1. Metode ADDIE

Metode ADDIE adalah model yang menggabungkan fase analisis, desain, pengembangan, implementasi, dan evaluasi pengembangan model. Dibuat pada tahun 1996 oleh Dick and Carry. Menurut Molenda (Dalam [15]) Pendekatan pembelajaran umum ini sesuai untuk penelitian pengembangan karena mengikuti proses yang berurutan namun interaktif (komunikasi dua arah atau banyak komponen komunikasi). Sedangkan menurut F. G. Constancio (Dalam [16]) Model ADDIE dapat menilai aktivitas pengembangan di setiap level.

Sebelum proses dimulai, ditentukan terlebih dahulu kerangka berfikir agar setiap tahap-tahap mempunyai tugas yang jelas. Menurut Sugiyono (Dalam [17]) kerangka berpikir adalah model konseptual yang menunjukkan bagaimana teori berhubungan dengan beberapa aspek yang telah ditentukan sebagai masalah yang signifikan.. Lalu, menurut Uma Sekaran (Dalam [18]) merupakan, "Model konseptual tentang bagaimana sebuah teori berhubungan dengan berbagai faktor yang telah diidentifikasi sebagai hal yang penting". Sehingga, ide paling mendasar atau sebagai metode untuk melaksanakan proyek penelitian secara lengkap dikenal sebagai kerangka berpikir. berikut adalah kerangka berpikir pada penelitian ini:



Gambar 2. Kerangka Berpikir

A. *Analysis* (Analisis)

Pada tahap awal ini peneliti menentukan waktu dan tempat penelitian, melakukan pengumpulan data, dan penentuan kebutuhan. Dengan menjalani tahapan analisis ini secara cermat dan komprehensif, peneliti akan memiliki landasan yang kuat untuk merencanakan, merancang, dan mengimplementasikan layanan yang efektif, efisien, dan sesuai dengan kebutuhan serta ekspektasi pengguna.

1. Waktu dan Tempat Penelitian

Periode penelitian berlangsung selama empat bulan, dari November 2022 hingga Februari 2023. Kantor Badan Penelitian dan Pengembangan Daerah Provinsi Sumatera Selatan yang berlokasi di Jalan Demang Lebar Daun No. 4864, Kota Palembang menjadi lokasi penelitian ini.

2. Pengumpulan Data

Ada dua pendekatan yang digunakan untuk mengumpulkan data: pendekatan pertama dilakukan untuk memproses data sebelum fase pengembangan sistem:

a. Observasi

Observasi dilakukan langsung terhadap pengguna sistem Dasawisma PKK Provinsi Sumatera Selatan saat mereka menggunakan sistem. Terdapat kurang lebih 3.263 pengguna sistem. Adapun fokus dari data yang dikumpulkan adalah pada bagian rekap data. Misalnya pada bagian rekap data anggota keluarga, ketika jumlah data bertambah dapat mempengaruhi kinerja aplikasi. Adapun data yang ditampilkan yaitu jumlah anggota keluarga, jumlah laki-laki, jumlah perempuan, jumlah yang sudah kawin, jumlah yang belum kawin, jumlah

janda, jumlah duda, jumlah yang sudah bekerja, jumlah yang belum atau tidak bekerja, dan jumlah yang pendidikan terakhirnya (SD, SLTP, SLTA, Diploma, S1, S2, S3).

b. Studi Literatur

Selain itu, data juga dikumpulkan melalui pencarian berbagai publikasi, termasuk jurnal, media internet, dan dokumentasi resmi dari para penyedia teknologi yang relevan.

3. Penentuan Kebutuhan

Berikut ini adalah spesifikasi *hardware* yang dibutuhkan:

Tabel 1 Spesifikasi *Hardware* yang dibutuhkan

Item	Deskripsi
Laptop	- *Toshiba Satellite L840* - *Memory: 8192 MB RAM* - *SSD: 256 GB* - *Processor: IntelI CoreI i5-3210M CPU @ 2.50 GHz (4 CPUs), ~ 2.5GHz*

Untuk penelitian ini, beberapa perangkat lunak juga diperlukan selain perangkat keras. Perangkat lunak ini meliputi yang berikut ini:

Tabel 2 Daftar *Software* yang dibutuhkan

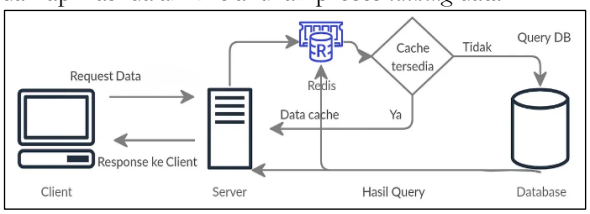
No.	*Software*	Kegunaan
1.	*Windows 10 Pro 64-bit*	Sebagai sistem operasi mode *development*
2.	*Microsoft Office 2021*	Untuk pembuatan, pengolahan, dan dokumentasi data
3.	*Google Chrome*	Untuk melakukan studi literatur dan menguji sistem
4.	*Draw.io*	Untuk membuat berbagai jenis *diagrams*, seperti ERD, *Flowchart* dan kerangka berpikir
5.	*Windows Terminal*	Untuk menjalankan *command* atau perintah
6.	*Visual Studio Code*	Untuk pengkodingan sistem
7.	*Laravel Framework*	Sebagai kerangka kerja dalam mengembangkan sistem
8.	*Laragon* - *Apache Httpd* - *MySQL* - *Redis*	Sebagai *tools* selama proses pengembangan dalam mode *development*
9.	*Laravel Debugbar*	Sebagai *tools* untuk memantau hasil *response-time*
10.	*Biznet Gio*	Sebagai tempat *hosting/vps* dan pembelian *domain*
11.	*PuTTY*	Untuk melakukan koneksi ke *terminal server* via SSH (*Secure Shell*)

A. *Design* (Desain)

Pada tahap *Design* merancang desain arsitektur sistem, merancang *caching* Redis, rancangan beberapa diagram seperti, *flowchart* dan Skenario Pengujian.

1. Arsitektur Sistem

Setelah melakukan analisa, maka dibuatlah arsitektur dari aplikasi dalam melakukan proses *caching* data.



Gambar 3. Arsitektur Sistem Dengan Redis

Gambar arsitektur sistem diatas menjelaskan bahwa *request* pertama akan melakukan perintah dari *server* ke *database* utama, tetapi saat terjadi lagi *request* yang sama, *server* akan melakukan pengecekan ke redis terlebih dahulu, jika data yang diminta tersedia di redis, maka *server* akan mengambilnya dari redis dan diteruskan ke *client*. Jika data yang diminta tidak tersedia di redis, maka *server* melakukan *querying* baru ke *database* utama (MySQL), setelah *database* mengembalikan data yang diminta, data kembalian akan disimpan ke redis agar tersedia pada *request* yang akan datang dan meneruskannya ke *client*. Sehingga *response time* yang didapat lebih cepat dan *client* tidak menunggu terlalu lama.

2. Rancangan *Caching* Redis

Dalam pengelolaan *cache*, terutama ketika menggunakan sistem *caching* seperti Redis, menetapkan konvensi nama *key* dari masing-masing *value* adalah praktik yang dapat membantu dalam pengorganisasian dan pengelolaan data dalam *cache*. Berikut ini adalah penamaan nama *key* agar datanya bisa diakses atau diambil.

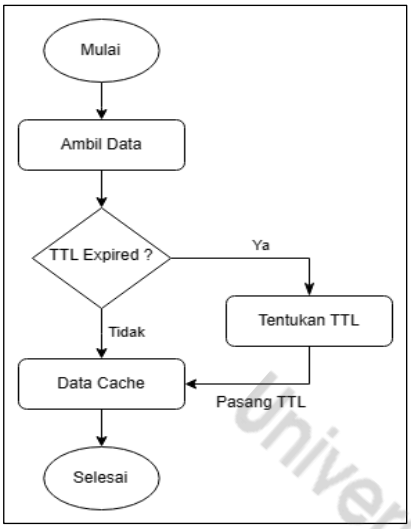
Redis		
TTL	Key	Value
1h	data-recap:[area]:{fb/fn/fm/fa}:regencies:page-[number]	data
1h	data-recap:[area]:{fb/fn/fm/fa}:districts:by-regency:[regency]:page-[number]	data
1h	data-recap:[area]:{fb/fn/fm/fa}:villages:by-district:[district]:page-[number]	data
1h	data-recap:[area]:{fb/fn/fm/fa}:dasawismas:by-village:[village]:page-[number]	data
1h	data-recap:[area]:{fb/fn/fm/fa}:family-heads:by-dasawisma:[dasawisma]:page-[number]	data

Gambar 4. Rancangan Struktur Data *Cache* di Redis

Informasi yang disimpan dalam redis berupa *key* dan *value*. *Key* merupakan sebuah penanda untuk data yang menyimpan hasil dari *query* sql terhadap rekap data yang ada di level kabupaten/kota, kecamatan, kelurahan/desa, dasawisma, keluarga berdasarkan area/dasawisma dan nomor halaman. Sedangkan *value* berisi data yang disimpan dalam bentuk struktur data jenis *strings* dan *hashes*. Lama waktu disimpannya adalah 1 jam.

3. *Flowchart*

FlowChart dibuat dengan tujuan untuk menjelaskan secara sederhana alur proses yang dilakukan.



Gambar 5. *Flowchart* Sistem dengan Redis

Gambar diatas menjelaskan, ketika *client* mengakses sebuah *route* rekap data, sebelum mengambil datanya dari *database* utama (MySQL), dilakukan pengecekan terlebih dahulu menggunakan *key* yang telah ditentukan pada redis apakah datanya ada dan masa TTL (*Time to Live*) masih valid. Jika bernilai benar, maka ambil data dari redis, jika tidak ada maka ambil data dari *database* MySQL dan juga simpan data balikannya pada redis. Ketika *client* selanjutnya ingin mengakses route GET itu lagi selama belum ada perubahan data dan TTL valid, maka akan mengambil datanya di redis, bukan lagi dari *database* MySQL. Lalu bagaimana jika terjadi perubahan data? misal ada penambahan data baru, pengubahan data, atau dihapus? Yang sistem akan lakukan adalah menghapus semua/spesifik data yang berkaitan pada redis dan biarkan nanti melakukan *caching* ulang dari MySQL ke Redis.

4. Skenario Pengujian

Pada tahap ini dijelaskan skenario pengujian sistem dengan jumlah data yang selalu bertambah sebanyak 200 Ribu. Masing-masing pengujian dilakukan sebanyak 3 kali *request* dari sisi *client*. Data yang disimpan pada redis menggunakan 2 jenis tipe data, yaitu *strings* dan *hashes*.

Pengujian pertama dilakukan tanpa menggunakan redis. Selanjutnya, Pengujian kedua dilakukan dengan menggunakan redis (tipe data *strings*. Terakhir, Pengujian ketiga dilakukan dengan menggunakan redis (tipe data *hashes*.

Semua pengujian dilakukan dari mulai *request* pertama sampai ketiga yang kemudian akan diteruskan ke sisi *server* dan dilanjutkan ke *database* MySQL/Redis. Data kembaliannya akan diteruskan kembali atau ditampilkan ke *client*. Tujuan pengujian ini adalah untuk mengetahui berapa lama waktu yang dibutuhkan sistem untuk menangani data yang semakin banyak.

B. *Development* (Pengembangan)

Pada tahap *Development* dilakukan proses pengembangan sistem seperti, penginstalan redis pada sistem operasi, instalasi *extensions* redis pada PHP, Instalasi *library* redis pada laravel, konfigurasi redis pada laravel, dan pengkodean redis sebagai sistem *caching* data.

3. Hasil dan Pembahasan

A. Implementasi (Hasil Pengujian)

Tahap selanjutnya, dilakukan implementasi pengujian sistem untuk memastikan bahwa redis berfungsi dengan baik dan sanggup dalam menangani data dalam jumlah besar. Pengujian dilakukan dengan jumlah data yang dimulai dari 200 Ribu sampai dengan 5 Juta dan dilakukan sebanyak tiga kali *request*. Adapun hasil yang diharapkan adalah bahwa dengan menggunakan redis sebagai sistem *caching* dapat meningkatkan kinerja sistem agar lebih cepat. Berikut ini adalah data yang akan disimpan pada redis

Tabel 3 Data Pengujian Yang Diambil dan Ditampilkan (Jumlah Data 200 Ribu)

Area	Jumlah									Jumlah Berpendidikan							
	Angg. Keluarga	Pria	Wanita	Kawin	Blm Kawin	Janda	Duda	Bekerja	Blm Bekerja	TK	SD	SMP	SMA	D3	S1	S2	S3
Ogan Kom ering Ulu	10.480	10.480	10.480	10.480	10.480	10.480	10.480	10.480	10.480	10.480	10.480	10.480	10.480	10.480	10.480	10.480	10.480
Ogan Kom ering Ilir	17.902	17.902	17.902	17.902	17.902	17.902	17.902	17.902	17.902	17.902	17.902	17.902	17.902	17.902	17.902	17.902	17.902
Muara Enim	13.558	13.558	13.558	13.558	13.558	13.558	13.558	13.558	13.558	13.558	13.558	13.558	13.558	13.558	13.558	13.558	13.558
Lahat	21.774	21.774	21.774	21.774	21.774	21.774	21.774	21.774	21.774	21.774	21.774	21.774	21.774	21.774	21.774	21.774	21.774

Musi Rawas	14.626	14.626	14.626	14.626	14.626	14.626	14.626	14.626	14.626	14.626	14.626	14.626	14.626	14.626	14.626	14.626	14.626	14.626
Musi Banyuasin	12.980	12.980	12.980	12.980	12.980	12.980	12.980	12.980	12.980	12.980	12.980	12.980	12.980	12.980	12.980	12.980	12.980	12.980
Banyuasin	21.852	21.852	21.852	21.852	21.852	21.852	21.852	21.852	21.852	21.852	21.852	21.852	21.852	21.852	21.852	21.852	21.852	21.852
Ogan Komering Ulu Timur	15.542	15.542	15.542	15.542	15.542	15.542	15.542	15.542	15.542	15.542	15.542	15.542	15.542	15.542	15.542	15.542	15.542	15.542
Ogan Komering Ulu Selatan	20.574	20.574	20.574	20.574	20.574	20.574	20.574	20.574	20.574	20.574	20.574	20.574	20.574	20.574	20.574	20.574	20.574	20.574
Ogan Ilir	13.458	13.458	13.458	13.458	13.458	13.458	13.458	13.458	13.458	13.458	13.458	13.458	13.458	13.458	13.458	13.458	13.458	13.458
Empat Lawang	10.338	10.338	10.338	10.338	10.338	10.338	10.338	10.338	10.338	10.338	10.338	10.338	10.338	10.338	10.338	10.338	10.338	10.338
Penukal Abab Lematang Ilir	7.978	7.978	7.978	7.978	7.978	7.978	7.978	7.978	7.978	7.978	7.978	7.978	7.978	7.978	7.978	7.978	7.978	7.978
Musi Rawas Utara	3.686	3.686	3.686	3.686	3.686	3.686	3.686	3.686	3.686	3.686	3.686	3.686	3.686	3.686	3.686	3.686	3.686	3.686
Palembang	5.326	5.326	5.326	5.326	5.326	5.326	5.326	5.326	5.326	5.326	5.326	5.326	5.326	5.326	5.326	5.326	5.326	5.326
Prabumulih	3.534	3.534	3.534	3.534	3.534	3.534	3.534	3.534	3.534	3.534	3.534	3.534	3.534	3.534	3.534	3.534	3.534	3.534
Pagar Alam	1.924	1.924	1.924	1.924	1.924	1.924	1.924	1.924	1.924	1.924	1.924	1.924	1.924	1.924	1.924	1.924	1.924	1.924
Lubuk Linggau	4.418	4.418	4.418	4.418	4.418	4.418	4.418	4.418	4.418	4.418	4.418	4.418	4.418	4.418	4.418	4.418	4.418	4.418

Data pada **Tabel 3.** Merupakan data yang akan menjadi data uji. Data tersebut diperoleh dari pengambilan beberapa kolom yang ada pada database MySQL lalu melalui proses perhitungan, yang kemudian data balikannya akan ditampilkan pada sisi *client* dan disimpan juga di redis.

1. Pengujian Sistem Tanpa Redis

Berikut ini hasil waktu respon yang dibutuhkan dari *request* pertama, kedua dan ketiga serta dilanjutkan perhitungan rata-rata respon waktunya jika tanpa redis.

Tabel 4 Hasil Pengujian Tanpa Redis

Jumlah Data	Req. ke-1 (s)	Req. ke-2 (s)	Req. ke-3 (s)	Rata-rata (s)
200 Ribu	1,13	0,825	0,735	0,90
400 Ribu	1,67	1,70	1,65	1,67

600 Ribu	7,74	4,70	3,85	5,43
800 Ribu	31,05	21,04	17,90	23,33
1 Juta	39,61	30,4	24,44	31,48
1,2 Juta	57,59	41,50	39,66	46,25
1,4 Juta	74,40	59,21	55,08	62,90
1,6 Juta	85,02	69,70	66,25	73,66
1,8 Juta	94,36	77,84	70,45	80,88
2 Juta	103,88	82,66	80,36	88,97
2,2 Juta	116,91	91,49	91,73	100,04
2,4 Juta	117,47	99,54	97,68	104,90
2,6 Juta	126,25	117,32	108,54	117,37
2,8 Juta	134,38	128,89	117,63	126,97
3 Juta	158,98	153,86	145,72	152,85
3,2 Juta	167,66	150,97	142,93	153,85
3,4 Juta	179,04	161,67	153,12	164,61
3,6 Juta	191,01	173,92	164,89	176,61
3,8 Juta	201,61	185,34	175,94	187,63
4 Juta	211,04	195,76	185,99	197,60
4,2 Juta	221,48	206,73	196,37	208,19
4,4 Juta	231,31	217,47	206,11	218,30
4,6 Juta	241,75	228,15	216,28	228,73
4,8 Juta	252,98	239,76	227,15	239,96
5 Juta	264,43	251,62	238,37	251,47

Dari tabel diatas, terlihat bahwa setiap kali ada *request*, waktu respon yang didapatkan selalu tinggi. Yang artinya *server* akan selalu mengambil datanya dari *database* utama (MySQL).

2. Pengujian Sistem Dengan Redis (Tipe Data *Strings*)

Berikut ini merupakan hasil waktu respon yang dibutuhkan dari *request* kedua, ketiga dan keempat serta dilanjutkan dengan perhitungan rata-rata respon waktunya jika menggunakan tipe data *strings*.

Tabel 5 Hasil Pengujian Redis dengan Jenis Tipe Data Strings

Jumlah Data	Req. ke-2 (s)	Req. ke-3 (s)	Req. ke-4 (s)	Rata-rata (s)
200 Ribu	0,430	0,453	0,464	0,449
400 Ribu	0,439	0,467	0,481	0,462
600 Ribu	0,485	0,519	0,459	0,488
800 Ribu	0,465	0,509	0,477	0,484
1 Juta	0,502	0,477	0,483	0,487
1,2 Juta	0,475	0,409	0,466	0,450
1,4 Juta	0,505	0,452	0,475	0,478
1,6 Juta	0,512	0,429	0,473	0,471
1,8 Juta	0,514	0,439	0,481	0,478
2 Juta	0,526	0,457	0,476	0,486
2,2 Juta	0,530	0,440	0,476	0,482
2,4 Juta	0,545	0,456	0,481	0,494
2,6 Juta	0,548	0,453	0,480	0,494
2,8 Juta	0,557	0,461	0,482	0,500
3 Juta	0,534	0,462	0,481	0,493
3,2 Juta	0,552	0,463	0,484	0,500
3,4 Juta	0,553	0,470	0,485	0,503
3,6 Juta	0,551	0,471	0,485	0,503
3,8 Juta	0,551	0,476	0,486	0,504
4 Juta	0,552	0,478	0,487	0,506
4,2 Juta	0,558	0,482	0,488	0,509
4,4 Juta	0,556	0,485	0,489	0,510
4,6 Juta	0,557	0,488	0,490	0,512
4,8 Juta	0,559	0,492	0,491	0,514
5 Juta	0,560	0,495	0,492	0,516

Pada *request* pertama tidak dimasukkan pada tabel karena data belum ada pada redis, sehingga *server* melakukan perintahnya ke *database* utama (MySQL). Dan pada saat *request* ke-2 sampai ke-4, waktu yang didapatkan sangatlah rendah atau bisa diartikan cepat.

3. Pengujian Sistem Dengan Redis (Tipe Data *Hashes*)

Berikut ini merupakan hasil waktu respon yang dibutuhkan dari *request* kedua, ketiga dan keempat serta nilai rata-rata respon waktunya jika menggunakan tipe data *hashes*.

Tabel 6 Hasil Pengujian Redis dengan Jenis Tipe Data Hashes

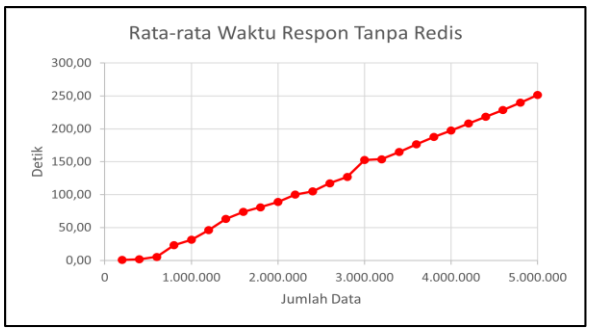
Jumlah Data	Req. ke-2 (s)	Req. ke-3 (s)	Req. ke-4 (s)	Rata-rata (s)
200 Ribu	0,450	0,478	0,455	0,461
400 Ribu	0,435	0,437	0,459	0,444
600 Ribu	0,466	0,513	0,468	0,482
800 Ribu	0,502	0,476	0,47	0,483
1 Juta	0,437	0,445	0,473	0,452
1,2 Juta	0,472	0,484	0,471	0,476
1,4 Juta	0,476	0,474	0,478	0,476
1,6 Juta	0,480	0,478	0,481	0,480
1,8 Juta	0,475	0,464	0,482	0,474
2 Juta	0,473	0,473	0,485	0,477
2,2 Juta	0,488	0,478	0,488	0,485
2,4 Juta	0,484	0,471	0,492	0,482
2,6 Juta	0,486	0,472	0,493	0,484
2,8 Juta	0,488	0,473	0,496	0,486
3 Juta	0,493	0,478	0,499	0,490
3,2 Juta	0,495	0,475	0,502	0,491
3,4 Juta	0,495	0,475	0,505	0,492
3,6 Juta	0,499	0,477	0,507	0,495
3,8 Juta	0,502	0,478	0,510	0,497
4 Juta	0,504	0,479	0,513	0,499
4,2 Juta	0,506	0,479	0,516	0,500
4,4 Juta	0,509	0,485	0,518	0,504
4,6 Juta	0,511	0,484	0,521	0,506
4,8 Juta	0,514	0,486	0,524	0,508
5 Juta	0,516	0,488	0,527	0,510

Sama seperti saat menggunakan tipe data *strings*, Pada *request* pertama tidak dimasukkan pada tabel karena data belum ada pada redis, sehingga *server* melakukan perintahnya ke *database* utama (MySQL). Dan pada saat request ke-2 sampai ke-4, waktu yang didapatkan sangatlah rendah atau bisa diartikan sangat cepat

B. Evaluasi

Tahap ini merupakan informasi hasil rata-rata waktu respon yang disajikan salam bentuk grafik.

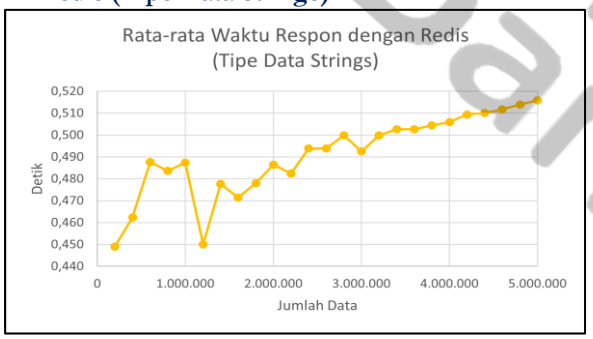
1. Grafik Rata-rata Waktu Respon Jika Tanpa Redis



Gambar 6. Grafik Rata-rata Waktu respon Jika Tanpa Redis

Dari grafik sebelumnya terlihat bahwa waktu respon akan selalu meningkat seiring dengan jumlah data yang selalu bertambah banyak dikarenakan balikan pada MySQL tidak ditangani dengan baik ke redis sehingga menyebabkan proses pada *server* dan *database* utama akan terus besar.

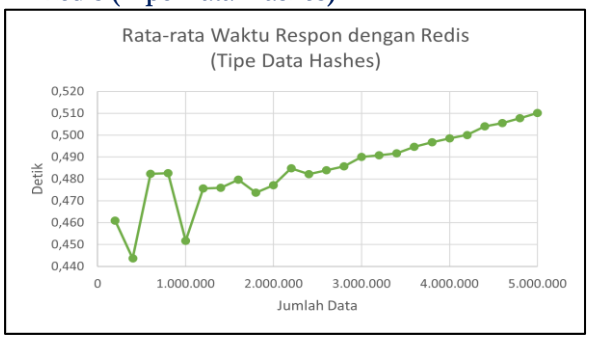
2. Grafik Rata-rata Waktu Respon Jika Dengan Redis (Tipe Data *Strings*)



Gambar 7. Grafik Rata-rata Waktu Respon Jika dengan Redis (Tipe Data *Strings*)

Terlihat bahwa terjadi penurunan waktu respon yang didapat dalam melakukan pengambilan data pada redis dengan menggunakan jenis tipe data *strings*. Ini dikarenakan datanya diambil langsung ke memori, pengambilan data ke memori lebih cepat dari pada ke jenis *storage* lain.

3. Grafik Rata-rata Waktu Respon Jika Dengan Redis (Tipe Data *Hashes*)



Gambar 8. Grafik Rata-rata Waktu Respon jika dengan Redis (Tipe Data *Hashes*)

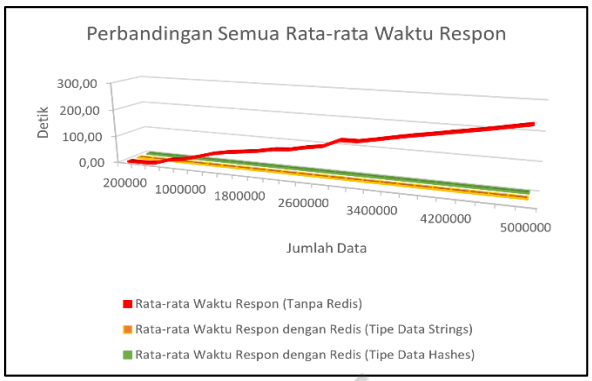
Terlihat juga bahwa dengan menggunakan redis dengan jenis tipe data *hashes* juga bisa menghasilkan waktu respon yang tidak kalah cepat dengan menggunakan tipe data *strings* dan tentunya lebih cepat dari tanpa menggunakan redis.

Secara keseluruhan dari tiga grafik sebelumnya, menunjukkan bahwa implementasi Redis dengan benar, dapat meningkatkan waktu respon sistem dalam memuat data besar, terutama untuk operasi baca. Redis tidak hanya mempercepat waktu respon secara keseluruhan tetapi juga lebih efektif dalam menangani data berukuran besar, menjaga performa yang baik seiring pertambahan data. Berikut ini merupakan tabel ringkasan rata-rata waktu respon yang dibutuhkan sistem dalam memuat data berdasarkan jumlah data yang tersedia.

Tabel 7 Hasil Semua Waktu Respon

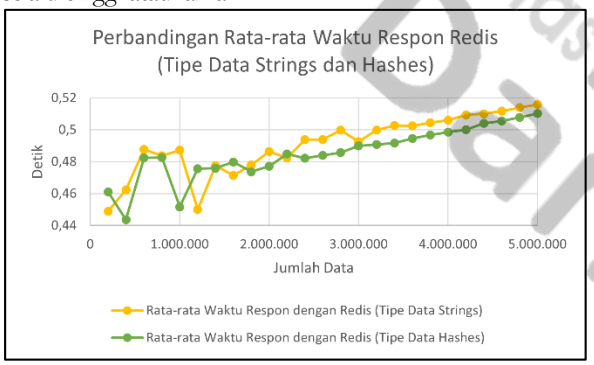
Jumlah Data	Rata-rata Waktu Respon Tanpa Redis (s)	Rata-rata Waktu Respon Dengan Redis *Strings* (s)	Rata-rata Waktu Respon Dengan Redis *Hashes* (s)
200 Ribu	0,90	0,449	0,461
400 Ribu	1,67	0,462	0,444
600 Ribu	5,43	0,488	0,482
800 Ribu	23,33	0,484	0,483
1 Juta	31,48	0,487	0,452
1,2 Juta	46,25	0,450	0,476
1,4 Juta	62,90	0,478	0,476
1,6 Juta	73,66	0,471	0,480
1,8 Juta	80,88	0,478	0,474
2 Juta	88,97	0,486	0,477
2,2 Juta	100,04	0,482	0,485
2,4 Juta	104,90	0,494	0,482
2,6 Juta	117,37	0,494	0,484
2,8 Juta	126,97	0,500	0,486
3 Juta	152,85	0,493	0,490
3,2 Juta	153,85	0,500	0,491
3,4 Juta	164,61	0,503	0,492
3,6 Juta	176,61	0,503	0,495
3,8 Juta	187,63	0,504	0,497
4 Juta	197,60	0,506	0,499
4,2 Juta	208,19	0,509	0,500
4,4 Juta	218,30	0,510	0,504
4,6 Juta	228,73	0,512	0,506
4,8 Juta	239,96	0,514	0,508
5 Juta	251,47	0,516	0,510

Dapat dilihat hasil pada kolom ke-1, rata-rata waktu respon yang didapat selalu tinggi atau lambat karena sistem belum mengimplementasikan redis. Sedangkan pada 2 kolom terakhir, dengan menggimplementasikan redis, bisa menghasilkan rata-rata waktu respon yang sangat signifikan menurun atau cepat dan selalu stabil. Jika dibuat dalam 1 grafik maka akan tampil seperti berikut.



Gambar 9. Grafik Perbandingan Semua Waktu Respon

Dari hasil akhir dari perbandingan waktu respon yang ditampilkan pada satu grafik menunjukkan bahwa jika tanpa menggunakan redis ketika data bertambah banyak, maka waktu yang di butuhkan untuk memuat data akan selalu tinggi atau lama.



Gambar 10. Grafik Perbandingan Waktu Respon antara Tipe Data *Strings* dengan *Hashes*

Sedangkan jika sistem telah mengimplementasikan redis, baik itu dengan tipe data *strings* atau *hashes* walaupun data bertambah banyak, hasil rata-rata waktu respon pada *request* ke-2 dan seterusnya selama belum ada perubahan data, hanya dibutuhkan waktu yang singkat dan tentunya stabil dalam memuat jumlah data yang semakin besar. Dan jika dilihat pada gambar sebelumnya antara manakah yang lebih baik antara tipe data *strings* dengan *hashes* untuk kasus masalah ini?. jawabannya adalah pada kasus penelitian ini, tipe data *hashes* mempunyai kinerja yang cepat.

4. Kesimpulan

Dari hasil pengujian dan evaluasi mengenai waktu respon yang didapat dalam memuat data, dapat disimpulkan bahwa dengan menggunakan redis untuk jumlah data sebanyak 5 juta diperoleh rata-rata waktu respon 0,516 detik untuk jenis struktur data *strings*, dan 0,510 detik untuk struktur data *hashes*. Ini lebih cepat dibandingkan tanpa menggunakan redis diperoleh rata-rata waktu respon 251,47 detik. Maka terjadi penurunan sebesar 99,7%. Yang artinya terdapat peningkatan kinerja sebesar 489,05%. Redis cocok untuk digunakan khususnya ketika butuh untuk proses *caching* data. Dan jika dilihat perbandingan dua respon waktu dari dua penggunaan tipe data pada redis untuk sistem *caching*, tipe data *hashes* adalah yang paling efisien, dikarenakan pada saat pengambilan datanya langsung bisa ditampilkan dalam bentuk paginasi karena respon data pada redisnya sudah berupa *array* sehingga waktu yang dibutuhkan untuk memuat datanya walaupun dengan *volume* yang semakin besar, rata-rata waktu yang dihasilkan tetap stabil. Sedangkan untuk tipe data *strings* harus diubah dulu dari *json* ke bentuk *array* agar bisa diproses untuk dipaginasi. Dan rata-rata waktu yang dihasilkan berdasarkan *volume* data yang terus bertambah pun tetap bisa mendapatkan nilai yang stabil. Dengan demikian, jika redis salah satunya digunakan sebagai solusi untuk masalah pada suatu sistem yang sudah mengalami penurunan kinerjanya ini sangat cocok jika di implementasikan dengan benar. Dan tidak lupa untuk selalu memantau kapasitas memori yang masih tersedia guna memastikan redis bisa bekerja secara optimal.

Akan tetapi masih terdapat beberapa kekurangan, sehingga diharapkan dapat dikembangkan lagi. Adapun yang disarankan oleh peneliti yaitu, Untuk sistem sekarang hanya menggunakan 2 GB RAM, Sehingga jika nanti batas maksimal RAM sudah penuh maka perlu adanya penambahan ukuran kapasitas RAM pada sisi *server* agar data yang ditampung dengan *limit* besar bisa tersimpan dengan baik. Dan jika pada sisi *database* atau kode program masih bisa dilakukan normalisasi dan optimasi sangat disarankan untuk dibuat lebih efisien lagi, Sehingga performanya akan lebih meningkat juga.

5. Daftar Pustaka

[1] D. Oktaviani, F. S. Papilaya, dan P. F. Tanaem, "Perancangan Aplikasi E-Menu Restaurant dengan Menggunakan Cloud Computing dan Serverless Architecture Lambda," *Explor. Sist. Inf. Dan Telematika*, vol. 12, no. 1, hlm. 1, Apr 2021, doi: 10.36448/jsit.v12i1.1887.

[2] Aziz dan Yaya Mulyana Abdul, "Model Kebijakan Peningkatan Laporan Kematian dalam Administrasi Kependudukan dan Catatan Sipil di Kabupaten Bandung Barat," *Sosiohumaniora*, vol. 19, no. 2, hlm. 140–148, 2019.

[3] D. Irawan dan A. T. Hidayat, "RANCANG BANGUN DASHBOARD KEPEGAWAIAN SEKOLAH TINGGI ILMU EKONOMI MUSI RAWAS (STIE MURA) LUBUKLINGGAU," vol. 9, 2019.

[4] W. F. Ramadhan, W. N. Dewi, dan C. Nas, "APLIKASI WEB PORTAL MANAJEMEN INFORMATIKA BERBASIS WEBSITE DENGAN MENGGUNAKAN FRAMEWORK CODEIGNITER DAN MYSQL PADA UNIVERSITAS CATUR INSAN CENDEKIA," *J. Digit*, vol. 10, no. 2, hlm. 124, Des 2020, doi: 10.51920/jd.v10i2.164.

[5] A. Frisdayanti, "PERANAN BRAINWARE DALAM SISTEM INFORMASI MANAJEMEN," vol. 1, 2019.

[6] A. M. Rosad, "IMPLEMENTASI PENDIDIKAN KARAKTER MELALUI MANAGEMEN SEKOLAH," *Tarbawi J. Keilmuan*

Manaj. Pendidik., vol. 5, no. 02, hlm. 173, Des 2019, doi: 10.32678/tarbawi.v5i02.2074.

[7] G. H. Irawan dan N. Ramdhani, "Analisa Performa Cache Database Redis dan MySQL dengan PHP".

[8] F. Febriyani, E. S. Pramukantoro, dan F. A. Bakhtiar, "Perbandingan Kinerja Redis, Mosquitto, dan MongoDB sebagai Message Broker pada IoT Middleware".

[9] M. G. Khoshkholgh, K. Navaie, K. G. Shin, V. C. M. Leung, dan H. Yanikomeroglu, "Caching or No Caching in Dense HetNets?," dalam *2019 IEEE Wireless Communications and Networking Conference (WCNC)*, Marrakesh, Morocco: IEEE, Apr 2019, hlm. 1–7. doi: 10.1109/WCNC.2019.8885724.

[10] R. Somya dan T. M. E. Nathanael, "PENGEMBANGAN SISTEM INFORMASI PELATIHAN BERBASIS WEB MENGGUNAKAN TEKNOLOGI WEB SERVICE DAN FRAMEWORK LARAVEL," *J. Techno Nusa Mandiri*, vol. 16, no. 1, hlm. 51–58, Mar 2019, doi: 10.33480/techno.v16i1.164.

[11] M. I. Zulfa, A. Fadli, dan A. W. Wardhana, "DESAIN MODEL VIEW CONTROLLER DALAM IMPLEMENTASI REDIS SERVER UNTUK MEMPERCEPAT AKSES DATA RELASIONAL," 2019.

[12] A. P. Meriani, D. R. Asih, dan S. Annisa, "Pengujian Distributed Cached Database Dengan Menggunakan Redis Pada Aplikasi MaBaUS".

[13] M. I. Zulfa, A. Fadli, dan A. W. Wardhana, "Application caching strategy based on in-memory using Redis server to accelerate relational data access," *J. Teknol. Dan Sist. Komput.*, vol. 8, no. 2, hlm. 157–163, Apr 2020, doi: 10.14710/jtsiskom.8.2.2020.157-163.

[14] F. A. Isfahani dan F. Nugraha, "Implementasi Load Balancing NGINX dan Mongodb Cluster serta Mekanisme Redis Caching".

[15] S. Rohaeni, "PENGEMBANGAN SISTEM PEMBELAJARAN DALAM IMPLEMENTASI KURIKULUM 2013 MENGGUNAKAN MODEL ADDIE PADA ANAK USIA DINI," *Instruksional*, vol. 1, no. 2, hlm. 122, Apr 2020, doi: 10.24853/instruksional.1.2.122-130.

[16] H. Yulianti, "Pemanfaatan Sistem Pelatihan E-Learning Pada Pengembangan Kinerja Karyawan di Masa Pandemi Covid-19 Dengan Pengujian ISO 9126," *MULTINETICS*, vol. 7, no. 1, hlm. 65–81, Okt 2021, doi: 10.32722/multinetics.v7i1.3769.

[17] J. Budiraharjo, M. Milisani, dan Y. Marosani, "PENGARUH PROSES REKRUTMEN DAN SELEKSI TERHADAP KUALITAS KERJA PEGAWAI BADAN PERENCANAAN PEMBANGUNAN DAERAH DKI JAKARTA," *J. Manaj.*, 2022.

[18] S. M. Hadi dan A. Samad, "Sistem Informasi Pengolahan Data Bantuan Beasiswa Siswa Miskin (BSM) Pada Kantor Wilayah Kementerian Agama Provinsi Maluku Utara," *J. Ilm. Ilk. - Ilmu Komput. Inform.*, vol. 2, no. 1, Jan 2019, doi: 10.47324/ilkominfo.v2i1.15.

LAMPIRAN

```
1.  <?php
2.  $parentKey = 'data-recap:' . $areaName;
3.  $keyCache = $parentKey.":fm:regencies:page-".$page;
4.
5.  if (Redis::exists($keyCache)) {
6.    $cachedData = Redis::get($keyCache);
7.    $toArray = json_decode($cachedData, true);
8.    $familyBuildings = LengthPager::paginate($toArray);
9.  } else {
10.   $familyBuildings = $this->getData()
11.    ->addSelect('regencies.id', 'regencies.name')
12.    ->join('regencies',
13.           'dasawismas.regency_id', '=', 'regencies.id')
14.    ->where('dasawismas.province_id', '=', 16)
15.    ->groupBy('regencies.id')
16.    ->orderBy('dasawismas.regency_id', 'ASC')
17.    ->when($user->role_id == 2 &&
18.        $user->admin->village_id != NULL,
19.        function (Builder $query) use ($user) {
20.          $query->where('dasawismas.village_id', '=',
21.            $user->admin->village_id);
22.        }
23.    )
24.    ->when($user->role_id == 2 &&
25.        $user->admin->district_id != NULL,
26.        function (Builder $query) use ($user) {
27.          $query->where('dasawismas.district_id', '=',
28.            $user->admin->district_id);
29.        }
30.    )
31.    ->when($user->role_id == 2 &&
32.        $user->admin->regency_id != NULL,
33.        function (Builder $query) use ($user) {
34.          $query->where('dasawismas.regency_id', '=',
35.            $user->admin->regency_id);
36.        }
37.    )
38.    ->when($user->role_id == 2 &&
39.        $user->admin->province_id != NULL,
40.        function (Builder $query) use ($user) {
41.          $query->where('dasawismas.province_id', '=',
42.            $user->admin->province_id);
43.        }
44.    )
```

```
45.    ->simplePaginate($this->perPage);
46.
47.   if ($familyBuildings->isNotEmpty()) {
48.     Redis::set($keyCache,json_encode($familyBuildings),
49.       'EX', config('database.redis.options.ttl')
50.      );
51.   }
52. }
53. ?>
```

Lampiran 1 Implementasi Tipe Data *Strings* ke Kode Program

```
1. <?php
2. $parentKey = 'data-recap:' . $areaName;
3. $keyCache = $parentKey .":fm:regencies:page-" . $page;
4. $keys = Redis::keys("$keyCache:*");
5.
6. if (!empty($keys)) {
7.   foreach ($keys as $value) {
8.     $data[] = Redis::hGetAll($value);
9.   }
10.    // Extract all next_page_url values
11.    $nextPage = array_column($data, 'next_page_url');
12.    // Check if all next_page_url values are empty
13.    $allEmpty = array_reduce($nextPageUrls,
14.      fn($carry, $url) => $carry && empty($url), true);
15.
16.    $toPaging = [
17.      'data' => $data,
18.      'total' => count($data),
19.      'per_page' => $this->perPage,
20.      'current_page' => $page,
21.      'hasMore' => $allEmpty ? false : true,
22.    ];
23.    $familyMembers = LengthPager::paginate($toPaging);
24.  } else {
25.    $familyMembers = $this->getData()
26.      ->addSelect('regencies.id', 'regencies.name')
27.      ->join('regencies',
28.          'dasawismas.regency_id', '=', 'regencies.id')
29.      ->where('dasawismas.province_id', '=', 16)
30.      ->groupBy('regencies.id')
```

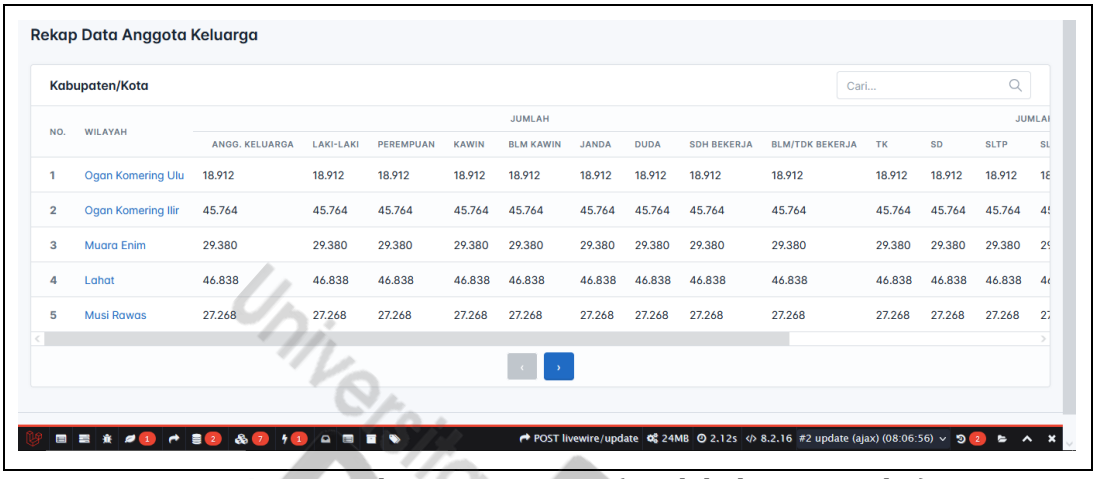
```
31.      ->orderBy('dasawismas.regency_id', 'ASC')
32.      ->when($user->role_id == 2 &&
33.          $user->admin->village_id != NULL,
34.          function (Builder $query) use ($user) {
35.            $query->where('dasawismas.village_id', '=',
36.              $user->admin->village_id);
37.          })
38.      ->when($user->role_id == 2 &&
39.          $user->admin->district_id != NULL,
40.          function (Builder $query) use ($user) {
41.            $query->where('dasawismas.district_id','=',
42.              $user->admin->district_id);
43.          })
44.      ->when($user->role_id == 2 &&
45.          $user->admin->regency_id != NULL,
46.          function (Builder $query) use ($user) {
47.            $query->where('dasawismas.regency_id', '=',
48.              $user->admin->regency_id);
49.          })
50.      ->when($user->role_id == 2 &&
51.          $user->admin->province_id != NULL,
52.          function (Builder $query) use ($user) {
53.            $query->where('dasawismas.province_id','=',
54.              $user->admin->province_id);
55.          })
56.      ->simplePaginate($this->perPage);
57.
58.    if ($familyMembers->isNotEmpty()) {
59.      $pageData = $familyMembers->toArray();
60.      Redis::transaction(function ($redis) use
61.        ($pageData, $keyCache) {
62.        foreach ($pageData['data'] as $value) {
63.          $redis->hMSet("$keyCache:{$value['id']}",
64.            [...$value, ...['next_page_url' =>
65.               $pageData['next_page_url']]
66.            ]);
67.          $redis->expire("$keyCache:{$value['id']}",
68.            config('database.redis.options.ttl'));
69.        }
70.      });
71.    }
72.  }
73.  ?>
```

Lampiran 2 Implementasi Tipe Data *Hashes* ke Kode Program

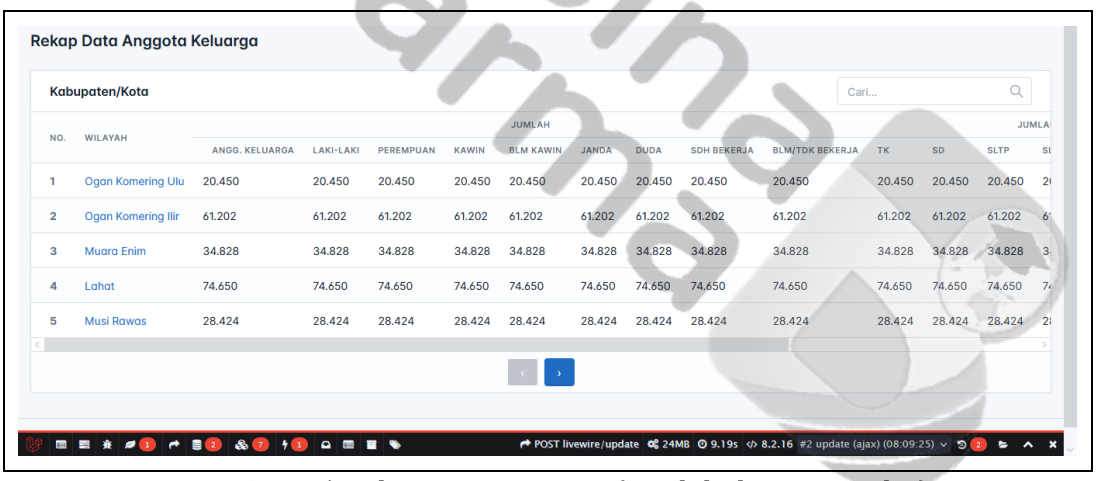
```
1.  <?php
2.  private function getData() {
3.    return FamilyMember::query()
4.      ->selectRaw("
5.          COUNT(family_members.family_head_id)
6.             AS family_members_count,
7.          COUNT(CASE WHEN
8.            family_members.gender = 'Laki-laki'
9.            THEN 1 ELSE 0 END)
10.             AS gender_males_count,
11.         COUNT(CASE WHEN
12.           family_members.gender = 'Perempuan'
13.           THEN 1 ELSE 0 END)
14.             AS gender_females_count,
15.         COUNT(CASE WHEN
16.           family_members.marital_status = 'Kawin'
17.           THEN 1 ELSE 0 END)
18.             AS marries_count,
19.         COUNT(CASE WHEN
20.           family_members.marital_status = 'Belum Kawin'
21.           THEN 1 ELSE 0 END)
22.             AS singles_count,
23.         COUNT(CASE WHEN
24.           family_members.marital_status = 'Janda'
25.           THEN 1 ELSE 0 END)
26.             AS widows_count,
27.         COUNT(CASE WHEN
28.           family_members.marital_status = 'Duda'
29.           THEN 1 ELSE 0 END)
30.             AS widowers_count,
31.         COUNT(CASE WHEN
32.           family_members.profession !=
33.             'Belum/Tidak Bekerja'
34.           THEN 1 ELSE 0 END)
35.             AS workings_count,
36.         COUNT(CASE WHEN
37.           family_members.profession LIKE
38.             '%Tidak Bekerja%'
39.           THEN 1 ELSE 0 END)
40.             AS not_workings_count,
41.         COUNT(CASE WHEN
42.           family_members.last_education = 'TK/PAUD'
43.           THEN 1 ELSE 0 END)
44.             AS kindergartens_count,
```

```
45.         COUNT(CASE WHEN
46.           family_members.last_education = 'SD/MI'
47.           THEN 1 ELSE 0 END)
48.             AS elementary_schools_count,
49.         COUNT(CASE WHEN
50.           family_members.last_education = 'SLTP/SMP/MTS'
51.           THEN 1 ELSE 0 END)
52.             AS middle_schools_count,
53.         COUNT(CASE WHEN
54.           family_members.last_education =
55.             'SLTA/SMA/MA/SMK'
56.           THEN 1 ELSE 0 END)
57.             AS high_schools_count,
58.         COUNT(CASE WHEN
59.           family_members.last_education = 'Diploma'
60.           THEN 1 ELSE 0 END)
61.             AS associate_degrees_count,
62.         COUNT(CASE WHEN
63.           family_members.last_education = 'S1'
64.           THEN 1 ELSE 0 END)
65.             AS bachelor_degrees_count,
66.         COUNT(CASE WHEN
67.           family_members.last_education = 'S2'
68.           THEN 1 ELSE 0 END)
69.             AS master_degrees_count,
70.         COUNT(CASE WHEN
71.           family_members.last_education = 'S3'
72.           THEN 1 ELSE 0 END)
73.             AS post_degrees_count
74.     ")
75.     ->join('family_heads',
76.            'family_members.family_head_id', '=',
77.            'family_heads.id')
78.     ->join('dasawismas',
79.            'family_heads.dasawisma_id', '=',
80.            'dasawismas.id');
81. }
82. ?>
```

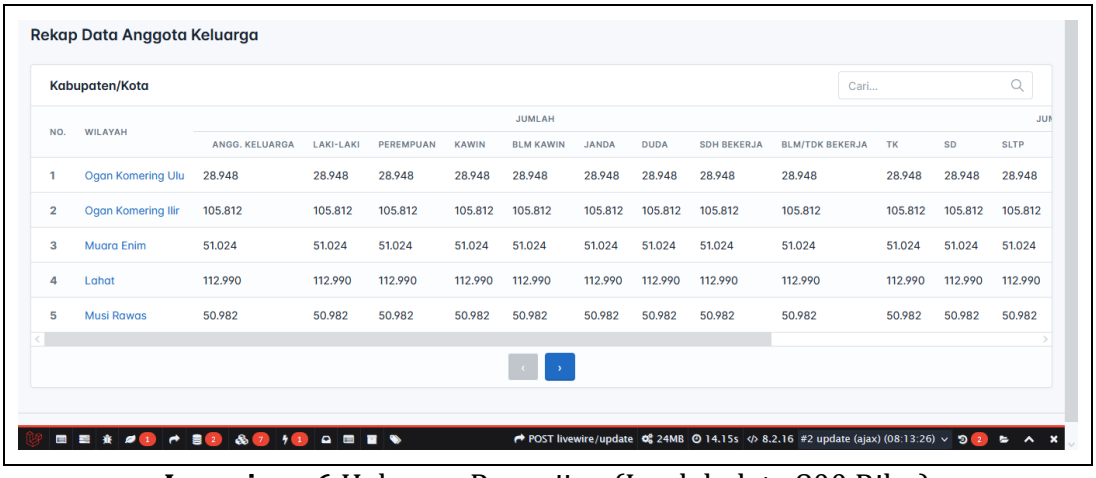
Lampiran 3 Kode Program Untuk Melakukan Perhitungan



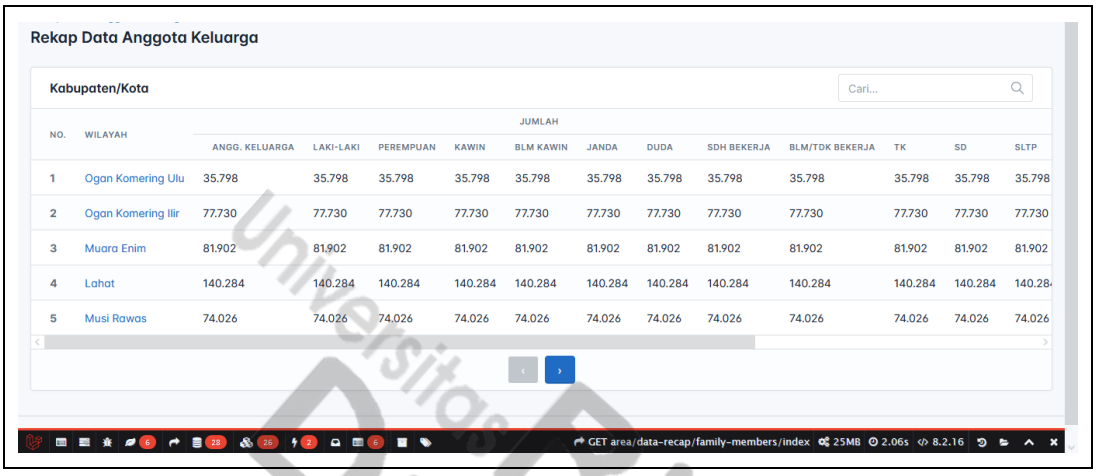
Lampiran 4 Halaman Pengujian (Jumlah data 400 Ribu)



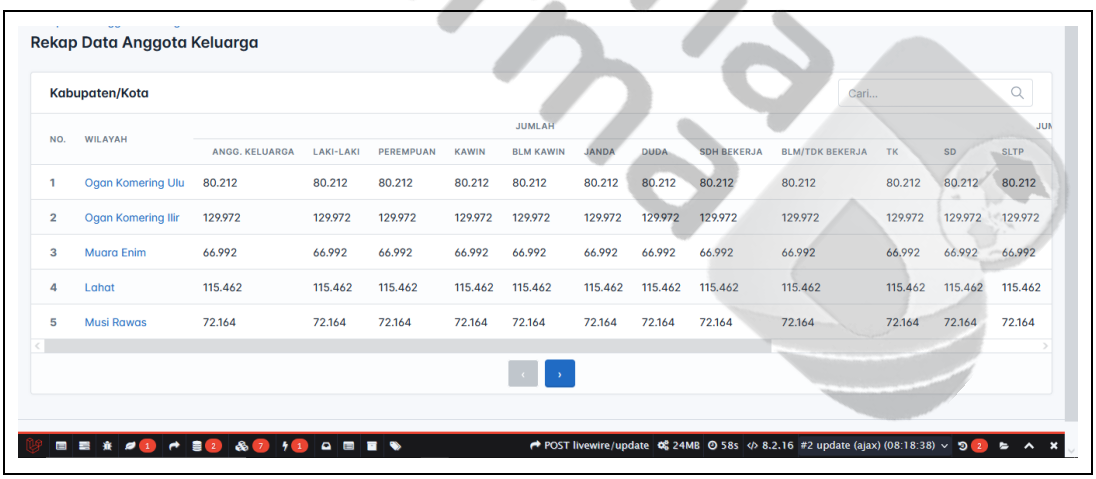
Lampiran 5 Halaman Pengujian (Jumlah data 600 Ribu)



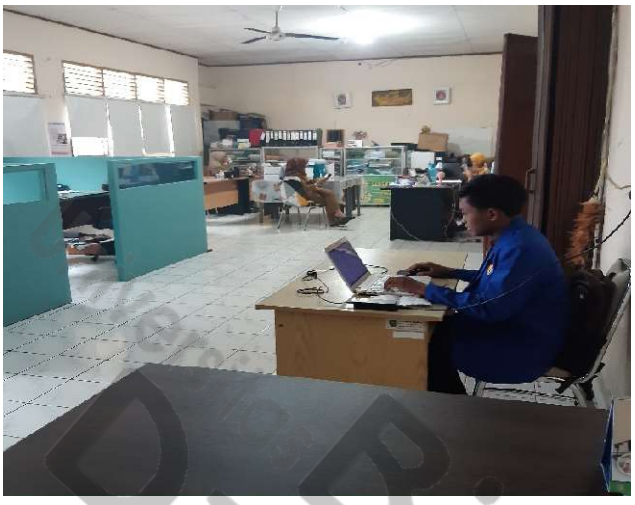
Lampiran 6 Halaman Pengujian (Jumlah data 800 Ribu)

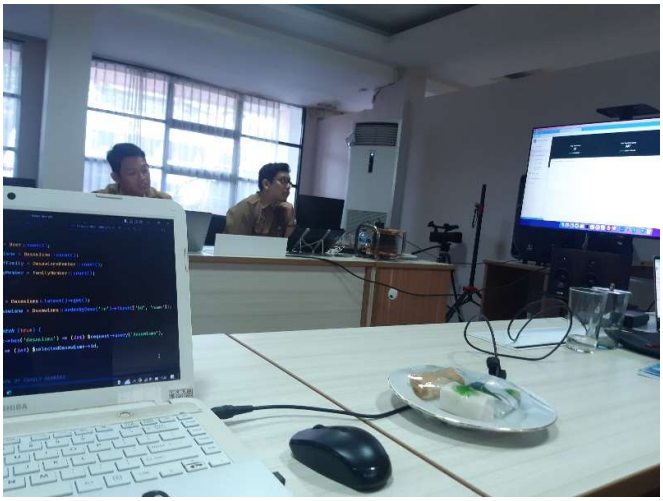


Lampiran 7 Halaman Pengujian (Jumlah Data 1 Juta)



Lampiran 8 Halaman Pengujian (Jumlah Data 1,200 Juta)





PEMERINTAH PROVINSI SUMATERA SELATAN
BADAN KESATUAN BANGSA DAN POLITIK
Jl. Kapten F. **Tendean No. 1059** Telp/Fax.(0711) 354715
Palembang 31129

SURAT KETERANGAN PENELITIAN

Nomor: 070/ 8698 /Ban. KBP/2022

Dasar :
1. Peraturan Menteri Dalam Negeri Republik Indonesia Nomor 3 Tahun 2018 Tentang Penerbitan Surat Keterangan Penelitian.
2. Peraturan Gubernur Sumatera Selatan Nomor 56 Tahun 2014 Tentang Pedoman Penerbitan Rekomendasi Penelitian.
3. Surat Wakil Rektor Bidang Akademik, Universitas Bina Darma Palembang.
   Nomor : 515/MBKM/Univ-BD/X/2022
   Tanggal : 1 Oktober 2022
   Hal : Permohonan Riset /Penelitian

DENGAN INI MEMBERIKAN REKOMENDASI KEPADA :

Nama /NIM	:	**DWI ROBBI PRASETYO, Dkk Terlampir / 191420064**
Pekerjaan	:	Mahasiswa
Alamat	:	Desa Marga Mulya, Kelurahan Marga Mulya Kecamatan Sinar Peninjauan, Kabupaten Ogan Komering Ulu.
Lokasi Penelitian	:	**Badan Penelitian dan Pengembangan Daerah Provinsi Sumatera Selatan.**
Jangka Waktu	:	3 Bulan
Penanggungjawab	:	Wakil Rektor Bidang Akademik, Universitas Bina Darma Palembang.
Tujuan	:	Mengadakan penelitian dalam rangka pelaksanaan Riset.
Judul Penelitian	:	**Sistem Website di Kantor Badan Penelitian dan Pengembangan Daerah Provinsi Sumatera Selatan.**
Catatan	:	1. Rekomendasi ini diterbitkan untuk kepentingan penelitian 2. Tidak dibenarkan melakukan penelitian/survei yang tidak sesuai/tidak ada kaitannya dengan judul kegiatan penelitian/survei diatas 3. Melaporkan hasil penelitian/survei kepada Gubernur Sumatera Selatan cq. Kepala Badan Kesatuan Bangsa dan Politik Provinsi Sumatera Selatan. 4. Surat rekomendasi ini dapat dicabut kembali apabila pemegang tidak mentaati ketentuan tersebut diatas.

Dikeluarkan di : Palembang
Pada tanggal : 10 Oktober 2022

Plt. KEPALA BADAN KESATUAN BANGSA DAN POLITIK
PROVINSI SUMATERA SELATAN

DR. H. SUNARTO, Sos, M.Si
Pembina TK.I / IV.B
NIP. 196906081990031006

Tembusan :
1. Gubernur Sumatera Selatan di Palembang (sebagai laporan)
2. Wakil Rewktor Bidang Akademik, Universitas Bina Darma Palembang

Lampiran Surat Pengantar
Nomor : 070/ /Ban.KBP/2022
Tanggal : Oktober 2022

Anggota Tim Riset/Penelitian

No.	Nama	NIM
1.	Muhammad Ade Syaifudin	
2.	Muhammad Zainul Iqbal	
3.	Dwi Robbi Parsetyo	
4.	Metta Setiana	

KEPUTUSAN
DEKAN FAKULTAS SAINS TEKNOLOGI

NOMOR :

TENTANG

PEMBIMBING PENELITIAN MAHASISWA
FAKULTAS SAINS TEKNOLOGI UNIVERSITAS BINA DARMA

Menimbang :

a. Bahwa mahasiswa semester akhir diharuskan melaksanakan penelitian dan menyusun skripsi sebagai salah satu syarat untuk menyelesaikan studi pada Program Strata 1 (S-1) Fakultas Sains Teknologi Universitas Bina Darma;
b. Bahwa untuk kelancaran dalam pelaksanaan penelitian dan penyusunan skripsi dimaksud, dipandang perlu untuk menunjuk dan menugaskan Pembimbing Skripsi bagi setiap mahasiswa;
c. Bahwa untuk memenuhi butir-butir di atas perlu diterbitkan Surat Keputusan sebagai landasan hukumnya.

Mengingat :

1. Undang-undang Nomor 20 tahun 2003;
2. Peraturan Pemerintah Nomor 60 tahun 1999;
3. Akte Pendirian Yayasan Nomor 95 tanggal 28 Desember 1993;
4. Surat Keputusan Menteri Pendidikan Nasional Republik Indonesia Nomor : 112/D/O/2002;
5. Statuta Universitas Bina Darma;
6. Surat Keputusan Rektor Universitas Bina Darma Nomor : 165/SK/UNIV-BD/XI/ 2008 tanggal 03Nopember 2008.

MEMUTUSKAN

Menetapkan :

PERTAMA : Menunjuk dan menugaskan saudara-saudara
1. Fatoni, M.M., M.Kom.
2.

berturut- turut sebagai Pembimbing Utama dan Pembimbing Pendamping dalam menyusun Skripsi mahasiswa di bawah ini :

Nama	: Dwi Robbi Prasetyo
Nim	: 191420064
Fakultas	: Sains Teknologi
Program Studi	: Teknik Informatika
Judul Penelitian	: Implementasi In-Memory Database (Redis) Untuk Mempercepat Operasi Relasional Dalam Menampilkan Rekap Data Pada Dashboard Sistem Dasawisma PKK Provinsi Sumatera Selatan

KEDUA : keputusan ini berlaku mulai tanggal ditetapkan sampai dengan yang bersangkutan menyelesaikan skripsi dan tugas akhir;

KETIGA : keputusan ini diberikan kepada yang bersangkutan untuk dilaksanakan sebagaimana mestinya, apabila di kemudian hari terdapat kekeliruan dalam penetapan ini akan diperbaiki sebagaimana mestinya.

Ditetapkan di Palembang
pada tanggal
Dekan,

Dr. Tata Sutabri, S.Kom., MMSI., MKM.

Tembusan disampaikan kepada Yth.
1. Pembimbing Utama dan Pendamping;
2. Ketua Program Studi;
3. Mahasiswa yang bersangkutan.

	FORMULIR	Nomor Dok :	FRM/TA/04/05
		Nomor Revisi	05
	Permohonan Pengajuan Judul Karya Akhir	Tgl. Berlaku	06 Maret 2023
		Standar SPMI	-

Perihal : **Permohonan Judul Karya Akhir**

Kepada Yth.
Ketua Program Studi Teknik Informatika
Fakultas Ilmu Komputer.
Universitas Bina Darma
Palembang

Dengan hormat,
Saya yang bertanda tangan dibawah ini, mahasiswa Program Studi Teknik Informatika Universitas Bina Darma Palembang.

Nama : Dwi Robbi Prasetyo
Nim : 191420064
Semester : 8
Program Studi : Teknik Informatika
Kelompok Riset : Software Engineering, Mobile Apps and Cloud Computing

Sehubungan dengan akan berakhirnya studi saya, maka dengan ini bermaksud mengajukan permohonan judul tugas akhir, Adapun judul yang saya ajukan sebagai berikut.

1. Implementasi *in-Memory Database (Redis)* Untuk Mempercepat Operasi Relasional Dalam Menampilkan Rekap Data Pada *Dashboard* Sistem Dasawisma PKK Provinsi Sumatera Selatan.
2. Penerapan *Design Pattern (Service* dan *Repository Layer)* Untuk Menyelesaikan Permasalahan Tertentu Dalam Pengembangan *Dashboard* Sistem Dasawisma PKK Provinsi Sumatera Selatan.

Atas perhatiannya, Saya ucapkan terima kasih.

Ketua Kelompok Riset,

Rasmila, S.Kom, M.Kom

Hormat saya,

Dwi Robbi Prasetyo

Mengetahui
Ketua Program Studi,

Alex Wijaya, S.Kom., M.IT.

Pembimbing Karya Akhir : **Fatoni, M.M., M.Kom** 06/03-2023

Syarat Pengajuan Judul :

- ***Formulir di isi lengkap dengan melampirkan jurnal atau paper***
- ***Fotocopy lembar PA yang sudah di acc oleh Pembimbing Akademik untuk mengajukan Skripsi (Khusus Program Sudi Sistem Informasi)***
- ***Formulir Nota Dinas (Khusus Fakultas Ekonomi dan Bisnis)***
- ***Fotocopy KRS yang tercantum Skripsi***
- ***Berkas dimasukan dalam Map Plastik Transparan warna*** *(Fak. Ilmu Komputer = Merah), (Fak. Ekonomi dan Bisnis = Kuning), (Fak.Psikologi, dan Fak. Komunikasi = Biru) (Fak. Teknik = Hijau), (Fak. Ilmu Keguruan, Ilmu Pendidikan dan Bahasa = Merah Maroon), (Fak. Vokasi =Orange muda).*

Universitas Bina Darma	FORMULIR PENILAIAN SKRIPSI/ KARYA AKHIR TEKNIK INFORMATIKA	Nomor Dok :	
		Nomor Revisi :	
		Tgl. Berlaku :	
		Kluasa ISO :	

LEMBAR KONSULTASI
SKRIPSI/KARYA AKHIR

NIM : **191420064**
NAMA MAHASISWA : **DWI ROBBI PRASETYO**
PROGRAM STUDI : **TEKNIK INFORMATIKA**
FAKULTAS : **SAINS DAN TEKNOLOGI**
JUDUL : **IMPLEMENTASI *IN-MEMORY DATABASE (REDIS)* UNTUK MEMPERCEPAT OPERASI RELASIONAL DALAM MENAMPILKAN REKAP DATA PADA *DASHBOARD* SISTEM DASAWISMA PKK PROVINSI SUMATERA SELATAN**
PEMBIMBING : **FATONI, M.M., M.Kom**

TANGGAL	KETERANGAN	TANDA TANGAN DOSEN PEMBIMBING
17/06 2023	proposal revisi — Metode penelitian — format penulisan — penomoran tabel dan rumus	
21/06-2023	Acc ujian proposal	

HALAMAN PENGESAHAN

IMPLEMENTASI *IN-MEMORY DATABASE* (*REDIS*) UNTUK MEMPERCEPAT OPERASI RELASIONAL DALAM MENAMPILKAN REKAP DATA PADA *DASHBOARD* SISTEM DASAWISMA PKK PROVINSI SUMATERA SELATAN

OLEH :

DWI ROBBI PRASETYO
191420064

PROPOSAL KARYA AKHIR

Disusun Sebagai Salah Satu Syarat Untuk Melakukan Penelitian

Disetujui,
Program Studi Teknik Informatika
Universitas Bina Darma

Dosen Pembimbing	Ketua Program Studi,
Fatoni , M.M., M.Kom	Alex Wijaya, S.Kom., MIT

Universitas Bina Darma	**FORMULIR Berita Acara Ujian Seminar Proposal Penelitian**	Nomor Dok :	FRM/TA/ 07
		Nomor Revisi :	04
		Tgl. Berlaku :	1 Jan 2019
		Klausa ISO :	

FORMULIR PERBAIKAN PROPOSAL PENELITIAN

Fakultas..Sains...Teknologi....
Program Studi Teknik..Informatika
Universitas Bina Darma

Nama : Dwi Robbi Prasetyo
NIM : 19142064
Program Studi : Teknik Informatika
Judul : Implementasi In-memory Database (Redis) Dalam Menampilkan Rekap Data Pada Dashboard sistem Pasawisma PKK Provinsi Sumatera Selatan

Catatan Perbaikan :
Perlihatkan Kodingan Redis ke mysql, komparasi antara kecepatan antara sebelum menggunakan redis (mysql) dengan sesudah ~~me~~ pakai redis.

Tim Penguji:
Ketua : Fatoni
Anggota Penguji : Muhammad Nasir
Anggota Penguji : Irman Effendy

Palembang, 19 ~~September~~ Agustus 2023
Ketua Prog. Studi Teknik Informatika

Alek Wijaya, S.Kom., M.I.T

No. Revisi : 04	Tanggal : 1/01/2019

SURAT KETERANGAN LULUS
UJIAN SEMINAR PROPOSAL PENELITIAN SKRIPSI
PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS TEKNOLOGI
UNIVERSITAS BINA DARMA

Nomor Dok. : FRM/TA/09
Tanggal : 1 Mei 2006 Rev. 00

Nama : **Dwi Robbi Prasetyo**

NIM/NIRM : **191420064**

Judul : Implementasi In-memory Database (redis) Untuk Mempercepat Operasi Relasional Dalam Menampilkan Rekap Data Pada Dashboard Sistem Dasawisma Pkk Provinsi Sumatera Selatan

Pembimbing Utama : 1. Fatoni, M.m., M.kom.

ngikuti Ujian Seminar Proposal Penelitian Skripsi Program Studi Teknik Informatika Strata Satu Fakultas knologi Universitas Bina Darma pada :

ggal : Senin, 17 Juli 2023
: 1. Fatoni, M.M., M.Kom.

2. Muhammad Nasir, M.M., M.Kom.

3. Irman Effendy, M.Kom

yatakan Dinyatakan LAYAK Untuk Dilanjutkan Ke Tahap Penelitian. Dengan Ini Mohon Kiranya Agar Dapat ikan SK Pembimbing Penelitian Guna Melanjutkan Penelitian Sampai Ujian Komprehensif Kepada Mahasiswa t. Atas Perhatian Dan Kerjasamanya Kami Mengucapkan Terima Kasih.

Palembang, 30 Agustus 2023
Program Studi Teknik Informatika
Fakultas Sains Teknologi

re nilai : **LULUS 88 (A)**

Alek Wijaya, S.kom., M.i.t.

ikan Skripsi dilakukan paling lambat 1 minggu
Ikut ujian Program

buatan SK Pembimbing:
alus Seminar, dibuktikan dengan Surat keterangan lulus seminar yang telah di ACC penguji dan Kaprogstudi (Asli)
ormulir perbaikan seminar yang telah di ACC penguji dan Kaprogstudi (Asli)
urat pengajuan judul dan pembimbing Tugas Akhir yang telah di ACC pembimbing dan Kaprogstudi (Fotocopy)
ekap nilai yang telah di ACC/cek oleh PPM (fotocopy)
otocopy Kwitansi BPP Terbaru , Seminar dan Kwitansi Bimbingan Skripsi
oto copy surat balasan dari Perusahaan
emua Berkas dimasukan kedalam Map Kertas warna merah 1 buah dan diserahkan di PPM Lantai 1
Mahasiswa wajib mempunyai foto copy (Arsip) semua berkas persyaratan diatas

Universitas Bina Darma	FORMULIR PENILAIAN SKRIPSI/ KARYA AKHIR TEKNIK INFORMATIKA	Nomor Dok :	
		Nomor Revisi :	
		Tgl. Berlaku :	
		Kluasa ISO :	

LEMBAR KONSULTASI SKRIPSI/KARYA AKHIR

NIM : 191420064
NAMA MAHASISWA : Dwi Robbi Prasetyo
PROGRAM STUDI : TEKNIK INFORMATIKA
FAKULTAS : SAINS DAN TEKNOLOGI
JUDUL : Implementasi *In-Memory Database* (Redis) Untuk Mempercepat Operasi Relasional Dalam Menampilkan Rekap Data Pada *Dashboard* Sistem Dasawisma PKK Provinsi Sumatera Selatan
PEMBIMBING : Fatoni, M.M., M. Kom.

TANGGAL	KETERANGAN	TANDA TANGAN DOSEN PEMBIMBING
06/09/2023	Bab 1 Acc Bab 2 Revisi Bab 3 Studi literatur Sistem Rinci	
11/9/2023	Bab 2 Acc Bab 3 Revisi Alur sistem lagi Bab 4 tambah flow Client - server	
08/08/2024	Bab 3, tambah Desain sistem Pengujian Lanjutkan ke bab 4	

Universitas Bina Darma	FORMULIR PENILAIAN SKRIPSI/ KARYA AKHIR TEKNIK INFORMATIKA	Nomor Dok :	
		Nomor Revisi :	
		Tgl. Berlaku :	
		Kluasa ISO :	

LEMBAR KONSULTASI
SKRIPSI/KARYA AKHIR

NIM : 191420064
NAMA MAHASISWA : DWI ROBBI PRASETYO
PROGRAM STUDI : TEKNIK INFORMATIKA
FAKULTAS : SAINS DAN TEKNOLOGI
JUDUL : IMPLEMENTASI *IN-MEMORY DATABASE (REDIS)* UNTUK MEMPERCEPAT OPERASI RELASIONAL DALAM MENAMPILKAN REKAP DATA PADA *DASHBOARD* SISTEM DASAWISMA PKK PROVINSI SUMATERA SELATAN
PEMBIMBING : FATONI, M.M., M.Kom

TANGGAL	KETERANGAN	TANDA TANGAN DOSEN PEMBIMBING
12/8-2024	bab 3 acc bab 4 revisi penyajian & pembahasan lanjut bab 5	
15/8-2024	bab 4 & 5 acc lengkapi lainnya	

Bina Darma	FORMULIR Berita Acara Seminar Hasil Penelitian	Nomor Dok :	FRM/TA/04/11
		Nomor Revisi :	01
		Tgl. Berlaku :	12 Juli 2023

FORMULIR PERBAIKAN SEMINAR HASIL PENELITIAN

: DWI Robbi prasetyo
: 19142 0064
gram Studi : TEknik Informatika
ultas : FTI Sains Teknologi
: Implementasi In-Memory Database (Redis) untuk mempercepat operasi Relasional Dalam menampilkan Rekap Data Pada Dashboard Sistem Dasawisma PKK Provinsi Sumatera selatan

tan Perbaikan :
laskan dashboard, harus menggunakan metode ADDIE, ttlah jurnal tentang Optimasi Query, Ubah spesifik data da redis jika ada aksi Pengubahan data pada database ama (mySQL), Jelaskan teori-teori yang mendukung nelitian tanpa menggunakan penomoran

Penguji:
a : Fatoni
gota Penguji : M. Natsir
gota Penguji : Irman Alfandy

Palembang, 23 September 2024
Ketua Program Studi ,

Alex Wijaya

SURAT KETERANGAN LULUS
UJIAN SARJANA SKRIPSI
PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS TEKNOLOGI
UNIVERSITAS BINA DARMA

Nomor Dok : FRM/WDS/01
Tanggal 01 Mei 2006, Rev. 00

Nama Lengkap : **Dwi Robbi Prasetyo**

NIM : **191420064**

Judul : Implementasi In-memory Database (redis) Untuk Mempercepat Operasi Relasional Dalam Menampilkan Rekap Data Pada Dashboard Sistem Dasawisma Pkk Provinsi Sumatera Selatan

Pembimbing Utama : 1. Fatoni, M.m., M.kom.

Telah mengikuti Ujian Komprehensif / Tugas Akhir II Program Studi Teknik Informatika Strata Satu Sains Teknologi Universitas Bina Darma pada :

Tanggal : Jumat, 23 Agustus 2024

Dengan ini dinyatakan LULUS dengan score nilai 88 (A). Atas perhatian dan kerjasamanya Kami mengucapkan terima kasih.

Palembang, 06 September 2024
Program Studi Teknik Informatika
Fakultas Sains Teknologi

Alek Wijaya, S.kom., M.I.t.

NB :
1. Syarat untuk mendaftar Wisuda
2. Informasi Pendaftaran Wisuda Hubungi Pusat Pelayanan Mahasiswa
3. Wajib Ditanda tangani oleh Ka. Prog. Studi

Implementasi Redis Untuk Mempercepat Pengambilan Data (Studi Kasus: Sistem Dasawisma PKK Sumsel)

ORIGINALITY REPORT

15%	14%	7%	10%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Universitas Putera Indonesia YPTK Padang Student Paper	2%
2	putusan3.mahkamahagung.go.id Internet Source	2%
3	e-journal.uajy.ac.id Internet Source	1%
4	Mulki Indana Zulfa, Rudy Hartanto, Adhistya Erna Permanasari. "Performance Comparison of Swarm Intelligence Algorithms for Web Caching Strategy", 2021 IEEE International Conference on Communication, Networks and Satellite (COMNETSAT), 2021 Publication	1%
5	journal.eng.unila.ac.id Internet Source	1%
6	elibrary.nusamandiri.ac.id Internet Source	1%
7	journal.ugm.ac.id	

	Internet Source	1 %
8	131design.nl Internet Source	<1 %
9	jurnal.unidha.ac.id Internet Source	<1 %
10	publishing-widyagama.ac.id Internet Source	<1 %
11	rtcl.eecs.umich.edu Internet Source	<1 %
12	jurnal.untan.ac.id Internet Source	<1 %
13	garuda.kemdikbud.go.id Internet Source	<1 %
14	jurnal.stkippgritulungagung.ac.id Internet Source	<1 %
15	etheses.uin-malang.ac.id Internet Source	<1 %
16	repo.palcomtech.ac.id Internet Source	<1 %
17	digilib.unila.ac.id Internet Source	<1 %
18	ejournal.unesa.ac.id Internet Source	<1 %

19	repository.stimaimmi.ac.id Internet Source	<1 %
20	journal.amikindonesia.ac.id Internet Source	<1 %
21	Submitted to Universitas Brawijaya Student Paper	<1 %
22	eastdj.wordpress.com Internet Source	<1 %
23	eprints3.upgris.ac.id Internet Source	<1 %
24	sitemap.academia.edu Internet Source	<1 %
25	Submitted to University of Wollongong Student Paper	<1 %
26	digilibadmin.unismuh.ac.id Internet Source	<1 %
27	id.123dok.com Internet Source	<1 %
28	jurnal.ubl.ac.id Internet Source	<1 %
29	ejurnal.stmik-budidarma.ac.id Internet Source	<1 %
30	www.researchgate.net Internet Source	<1 %

31	1library.co Internet Source	<1%
32	elibrary.stipram.ac.id Internet Source	<1%
33	Asrean Hendi, Caswita Caswita, Een Yayah Haenilah. "Pengembangan Media Pembelajaran Interaktif Berbasis Strategi Metakognitif untuk Meningkatkan Kemampuan Berpikir Kritis siswa", Jurnal Cendekia : Jurnal Pendidikan Matematika, 2020 Publication	<1%
34	digilib.uinsa.ac.id Internet Source	<1%
35	ejournal.nusamandiri.ac.id Internet Source	<1%
36	eprints.uny.ac.id Internet Source	<1%
37	ojs.unikom.ac.id Internet Source	<1%
38	text-id.123dok.com Internet Source	<1%
39	"Final program", 2016 URSI Asia-Pacific Radio Science Conference (URSI AP-RASC), 2016 Publication	<1%

40	"Web, Artificial Intelligence and Network Applications", Springer Science and Business Media LLC, 2020 Publication	<1%
41	Asyifa Maulana, Intan Purnamasari, Iqbal Maulana. "RANCANG BANGUN WEBSITE LAYANAN JASA REPARASI ALAT ELEKTRONIK RUMAH TANGGA MENGGUNAKAN FRAMEWORK LARAVEL (STUDI KASUS: CV. XYZ)", Jurnal Informatika dan Teknik Elektro Terapan, 2024 Publication	<1%
42	Karina Pereira Lima. ">i/i<: preditores relativos à personalidade, habilidades sociais e saúde mental ao longo da residência médica", Universidade de São Paulo. Agência de Bibliotecas e Coleções Digitais, 2018 Publication	<1%
43	jtsiskom.undip.ac.id Internet Source	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		

HALAMAN PERSETUJUAN

Artikel Publikasi Ilmiah berjudul "Implementasi *In-Memory Database* (REDIS) Untuk Mempercepat Operasi Relasional Dalam Menampilkan Rekap Data Pada *Dashboard* Sistem Dasawisma PKK Provinsi Sumatera Selatan" oleh

Nama : Dwi Robbi Prasetyo

Nim 191420064

Program Studi : Teknik Informatika

Telahdisetujui untuk diajukan di Seminar Hasil Karya Akhir.

Mengetahui
Ketua Program Studi

Alek Wijaya, S.Kom., M.I.T

Palembang, 16 Agustus 2023
Menyetujui,
Pembimbing

Fatoni, M.M., M.Kom

EXPLORE
Jurnal Sistem Informasi dan Telematika
(Telekomunikasi, Multimedia dan Informatika)

Bandar Lampung, 19 September 2024

No : **109/Explore/PSTI/IX/2024**
Hal : **Letter of Acceptance**
Lampiran : **-**

LETTER OF ACCEPTANCE

Kepada Yth:

Sdr/i. **Dwi Robbi Prasetyo, Fatoni, Muhammad Nasir, Irman Effendy**

Di – Tempat

Melalui Surat ini kami pengelola jurnal **Explore: Jurnal Sistem Informasi dan Telematika (Telekomunikasi, Multimedia dan Informatika)** Mengucapkan Selamat atas artikel jurnal yang Bapak/ibu kirim Yang berjudul:

Implementasi Redis Untuk Mempercepat Pengambilan Data
(Studi Kasus: Sistem Dasawisma PKK Sumsel)

Telah melalui Proses Peer Review dan dinyatakan DITERIMA untuk di PUBLIKASIKAN di Explore: Jurnal Sistem Informasi dan Telematika (Telekomunikasi, Multimedia dan Informatika) pada Volume 15 No 2 Desember 2024.

Berkaitan dengan hal tersebut kami informasikan kepada Sdr/I:

1. Untuk mengirimkan Biaya administrasi Jurnal **Explore: Jurnal Sistem Informasi dan Telematika (Telekomunikasi, Multimedia dan Informatika) sebesar Rp.500.000 (Lima Ratus ribu rupiah) ke Rekening Bank Mandiri no rek: 1140010274184 an Robby Yuli Endra per artikel Jurnal.**
2. Untuk rincian tersebut pemakalah mendapatkan Url publikasi di website penerbit jurnal.
3. Bukti Transfer dapat dikirimkan ke email: explore@ubl.ac.id atau robby.yuliendra@ubl.ac.id

Demikian surat ini kami sampaikan, kami berharap kerja sama kembali dari saudara untuk dapat berpartisipasi pada edisi berikutnya dan surat ini dapat digunakan sebagaimana mestinya.

Ketua Redaksi Explore

JSITEXPLORE

Ahmad Cucus, S.Kom., M.Kom

Universitas Bina Darma	FORMULIR KELAYAKAN JILID SKRIPSI	Nomor Dok	:	FRM/TA/03/03
		Nomor Revisi	:	01
		Tgl. Berlaku	:	01 Februari 2017
		Klausa	:	7.5

CHECK LIST KESESUAIAN FORMAT TULISAN

NAMA : Dwi Robbi Prasetyo
NIM : 191420064
WARNA JILID :

NO	URAIAN	PEMBIMBING	PRODI
1	**COVER LUAR**	✓	
2	**COVER DALAM**	✓	
3	**LEMBAR PENGESAHAN**	✓	
4	**LEMBAR PERSETUJUAN**	✓	
5	**PERNYATAAN**	✓	
6	**DAFTAR ISI**	✓	
	6.1. INDENTASI, DOT LEADER	✓	
	6.2. UKURAN, JENIS, KAPITALISASI FONT	✓	
	6.3. CETAK MIRING/CETAK TEBAL	✓	
	6.4. JUDUL DAN PENOMORAN BAB/SUBBAB	✓	
	6.5. ISTILAH ASING/DAERAH	✓	
	6.6. KESESUAIAN NOMOR HALAMAN	✓	
7	**DAFTAR GAMBAR**	✓	
	7.1. INDENTASI, DOT LEADER	✓	
	7.2. UKURAN, JENIS, KAPITALISASI FONT	✓	
	7.3. CETAK MIRING/CETAK TEBAL	✓	
	7.4. JUDUL DAN PENOMORAN GAMBAR	✓	
	7.5. ISTILAH ASING/DAERAH	✓	
	7.6. KESESUAIAN NOMOR HALAMAN	✓	
8	**DAFTAR TABEL`**	✓	
	8.1. INDENTASI, DOT LEADER	✓	
	8.2. UKURAN, JENIS, KAPITALISASI FONT	✓	
	8.3. CETAK MIRING/CETAK TEBAL	✓	
	8.4. JUDUL DAN PENOMORAN TABEL	✓	
	8.5. ISTILAH ASING/DAERAH	✓	
	8.6. KESESUAIAN NOMOR HALAMAN	✓	
9	**KATA PENGANTAR**	✓	
10	**ABSTRAK**	✓	
	10.1. JUMLAH KATA (100-150 KATA)	✓	
	10.2. PARAGRAF TUNGGAL	✓	
	10.3. ISTILAH ASING	✓	
	10.4. KATA KUNCI	✓	
11	**PENULISAN ISI BAB**	✓	
	11.1. UKURAN FONT, SPASI PADA JUDUL BAB	✓	
	11.2. PENOMORAN SUBBAB MAKSIMAL LEVEL 4 (1.1.1.1)	✓	
	11.3. UKURAN, JENIS, KAPITALISASI FONT	✓	
	11.4. CETAK MIRING/CETAK TEBAL	✓	
	11.5. INDENTASI PARAGRAF DAN RINCIAN	✓	
	11.6. PENULISAN NOMOR DAFTAR RINCIAN	✓	
	11.7. BARIS KOSONG DI BAWAH HALAMAN MAX 3 BARIS	✓	
12	**GAMBAR**	✓	
	12.1. PENOMORAN GAMBAR	✓	
	12.2. RUJUKAN PADA NOMOR GAMBAR	✓	
	12.3 UKURAN MAX 1/2 HALAMAN	✓	
13	**TABEL**	✓	
	13.1. PENOMORAN TABEL	✓	
	13.2. RUJUKAN PADA NOMOR TABEL	✓	
	13.3. TABEL UTUH TIDAK TERPOTONG	✓	
	13.4. SPASI TUNGGAL 0 POINT BEFORE/AFTER	✓	
14	**DAFTAR PUSTAKA**	✓	
	14.1. SESUAI FORMAT BAKU PADA PEDOMAN	✓	
	14.2. TERCANTUM PADA BAB I/BAB II	✓	
	TELAH DIPERIKSA DAN SESUAI DENGAN PEDOMAN PENULISAN SKRIPSI FILKOM	✓	

Palembang, 25/05 20 24
Pemeriksa Kelayakan,

(Fatoni, M.M., M.Kom)