

**PENGEMBANGAN APLIKASI TASK TRACKER PLUS BERBASIS WEB
DENGAN MENGGUNAKAN REST API DAN GOLANG**

TUGAS AKHIR



FINNDY NOVERITA SYAFITRI

**PROGRAM STUDI DIII SISTEM INFORMASI
DEPARTEMEN TEKNIK
FAKULTAS VOKASI
UNIVERSITAS AIRLANGGA
2023**

**PENGEMBANGAN APLIKASI TASK TRACKER PLUS BERBASIS WEB
DENGAN MENGGUNAKAN REST API DAN GOLANG**

HALAMAN JUDUL



**FINNDY NOVERITA SYAFITRI
152011513011**

**PROGRAM STUDI DIII SISTEM INFORMASI
DEPARTEMEN TEKNIK
FAKULTAS VOKASI
UNIVERSITAS AIRLANGGA
2023**

HALAMAN PERNYATAAN TIDAK MELAKUKAN PLAGIAT

Bagian atau keseluruhan isi Tugas Akhir ini tidak pernah diajukan untuk mendapatkan gelar akademis pada bidang studi dan / atau Universitas lain dan tidak pernah dipublikasikan / ditulis oleh individu selain penyusun, kecuali bila dituliskan dengan format dalam isi Tugas Akhir.

Apabila ditemukan bukti bahwa pernyataan saya tidak benar, maka saya bersedia menerima sanksi sesuai ketentuan yang berlaku di Universitas Airlangga.

Surabaya, Juli 2023

Penulis



Finndy Noverita Syafitri

NIM. 152011513011

LEMBAR PERSETUJUAN TUGAS AKHIR

**PENGEMBANGAN APLIKASI *TASK TRACKER PLUS* BERBASIS WEB
DENGAN MENGGUNAKAN *REST API* DAN *GOLANG***

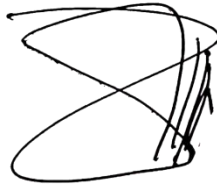
TUGAS AKHIR

Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Ahli Madya
Bidang Sistem Informasi
Pada Departemen Teknik Fakultas Vokasi
Universitas Airlangga

Oleh:
Finndy Noverita Syafitri
152011513011



Disetujui oleh:
Pembimbing

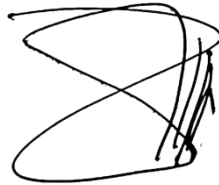


Drs, Eto Wuryanto, DEA.
NIP. 196609281991021001

**LEMBAR PENGESAHAN NASKAH
TUGAS AKHIR**

Judul : Pengembangan Aplikasi Task Tracker Plus Berbasis Web
Dengan Menggunakan Rest Api Dan Golang
Penulis : Finndy Noverita Syafitri
NIM : 152011513011
Pembimbing : Drs, Eto Wuryanto, DEA.
Tanggal Ujian : Senin, 24 Juli 2023

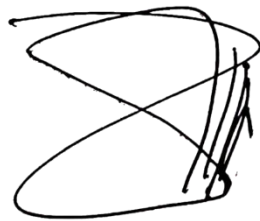
Disetujui Oleh :
Pembimbing,



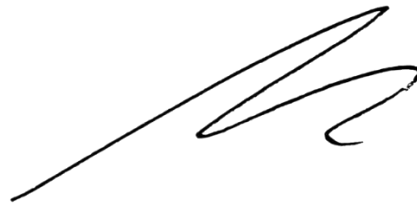
Drs, Eto Wuryanto, DEA.
NIP. 196609281991021001

Ketua Departemen Teknik

Koordinator Program Studi D3
Sistem Informasi



Drs, Eto Wuryanto, DEA.
NIP. 196609281991021001



Fitri Retrialisca S.Kom., M.Kom
NIP. 199303302018083201

LEMBAR PENGESAHAN PANITIA PENGUJI

Tugas Akhir ini Telah Diujikan dan Disahkan Dihadapan Komisi Penguji

Program Studi : DIII Sistem Informasi
Departemen : Teknik
Fakultas : Vokasi Universitas Airlangga
Hari/Tanggal : Senin, 24 Juli 2023
Pukul : 11.00 – 13.30
Tempat : Rapat Pimpinan 2 Fakultas Vokasi kampus B Universitas
Airlangga

Panitia Penguji terdiri dari:

Ketua Penguji



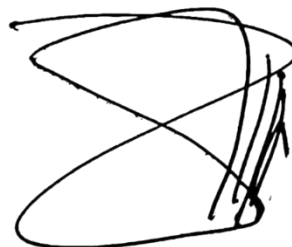
Endah Purwanti, S.Si., M.Kom.
NIP. 197812172005012001

Anggota Penguji I



Tesa Eranti Putri S.Kom., M.Kom.
NIP. 198904232016033201

Anggota Penguji II



Drs. Eto Wuryanto, DEA.
NIP. 196609281991021001

PEDOMAN PENGGUNAAN TUGAS AKHIR

Naskah Tugas Akhir ini tidak dipublikasikan, tetapi tersedia di perpustakaan dalam lingkungan Universitas Airlangga. Diperkenankan untuk dipakai sebagai referensi kepustakaan, tetapi pengutipan seizin penulis dan harus menyebutkan sumbernya sesuai kebiasaan ilmiah

Dokumen Tugas Akhir ini merupakan hak milik Universitas Airlangga

KATA PENGANTAR

Puji syukur saya panjatkan kehadirat Allah SWT atas segala nikmat dan karuniaNya, sehingga penulis dapat menyelesaikan Tugas Akhir dengan Judul “PENGEMBANGAN APLIKASI TASK TRACKER PLUS BERBASIS WEB DENGAN MENGGUNAKAN REST API DAN GOLANG” dengan tepat waktu.

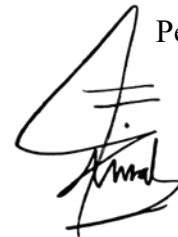
Dalam pembuatan laporan ini tidak lepas dari bantuan dari berbagai pihak, terima kasih diantaranya :

1. Allah SWT yang telah memberikan rahmat dan hidayah-Nya.
2. Bapak Prof. Dr. Anwar Ma’aruf, drh., M.Kes. selaku Dekan Fakultas Vokasi Universitas Airlangga.
3. Bapak Drs. Eto Wuryanto, DEA. selaku Ketua Departemen Teknik Fakultas Vokasi Universitas Airlangga.
4. Ibu Fitri Retrialisca S.Kom., M.Kom selaku Koordinator Program Studi D-III Sistem Informasi Fakultas Vokasi Universitas Airlangga.
5. Bapak Drs. Eto Wuryanto, DEA. selaku Dosen Pembimbing yang telah membimbing dan memberikan arahan saat melaksanakan Tugas Akhir.
6. Kedua orang tua saya yang telah mendukung secara lahir dan batin atas semua kegiatan yang telah saya lakukan dalam pelaksanaan Tugas Akhir.
7. Semua pihak yang telah membantu dalam proses penyelesaian Tugas Akhir.

Akhir kata dengan segala keterbatasan penulis berharap Tugas Akhir ini dapat bermanfaat bagi semua terutama di Program Studi D-III Sistem Informasi Fakultas Vokasi Universitas Airlangga.

Surabaya, 7 Juli 2023

Penulis



Finndy Noverita Syafitri

Finndy Noverita Syafitri, 2023. Pengembangan Aplikasi Task Tracker Plus Berbasis Web Dengan Menggunakan Rest Api Dan Golang. Tugas akhir ini dibimbing oleh Drs. Eto Wuryanto, DEA. Program Studi Diploma Sistem Informasi, Fakultas Vokasi, Universitas Airlangga

ABSTRAK

Perkembangan teknologi pada era modern saat ini berkembang begitu pesat, sehingga membuat pengelolaan tugas dan informasi menjadi lebih mudah. Pada organisasi atau individu dalam tuntutan dan kebutuhan yang kompleks, pengelolaan tugas menjadi semakin penting. Mengelola tugas dengan baik dengan memantau kemajuan secara *real-time* untuk mendapatkan informasi yang akurat tentang status tugas dibutuhkan suatu sistem yang memadai untuk mempermudah pengolahan tugas. Sehingga penelitian kali ini mengusulkan aplikasi web Task Tracker berbasis REST API dan MVC menggunakan bahasa pemrograman Golang untuk membantu dalam pengelolaan tugas. Tujuan penelitian ini adalah merancang dan mengimplementasikan arsitektur REST API dan pola MVC menggunakan pemrograman Golang untuk mencapai performa yang tinggi dan kinerja yang responsif. Program dibangun menggunakan alat visual studio code dengan database PostgreSQL dan pengujian sistem menggunakan *black box testing*. Proses-proses yang dilalui untuk penelitian ini yang pertama adalah melakukan identifikasi masalah pada PT.Ruang Raya Indonesia dan merumuskan masalah kemudian melakukan studi literature mengenai topik atau masalah yang sesuai dengan identifikasi masalah dan solusi yang akan digunakan nanti. Setelah studi literature selesai lanjut pada analisis dan perancangan sistem, selanjutnya dilakukan implementasi sistem dan melakukan pengujian sistem, kemudian dihasilkan sebuah kesimpulan dari penelitian yang dilakukan. Hasil dari penelitian ini berupa rancangan dan implementasi aplikasi Task Tracker Plus dengan menerapkan arsitektur REST API dan pola MVC menggunakan pemrograman Golang.

Kata Kunci: Web Task Tracker, Rest Api, MVC, Pemrograman Golang

Finndy Noverita Syafitri, 2023. Web-Based Task Tracker Plus Application Development Using Rest Api And Golang. This final project war supervised by Information Systems Diploma Study Program, Faculty of Vocational Studies, Airlangga University

ABSTRACT

Technological developments in modern era are developing so rapidly, thus making managing tasks and information easier. In organizations or individuals with complex demands and needs, task management becomes increasingly important. Managing tasks properly by monitoring progress in real-time to obtain accurate information about task status requires an adequate system to facilitate task processing. This research proposes a web application Task Tracker based on REST API and MVC using the Golang programming language to assist in task management. The purpose of this research is to design and implement a REST API architecture and MVC pattern using Golang programming to achieve high performance and responsive performance. The program was built using the visual studio code tool with the PostgreSQL database and system testing using black box testing. The processes that were passed for this research were to first identify problems at PT. Ruang Raya Indonesia and formulate problems then conduct literature studies on topics or problems that are in accordance with problem identification and solutions that will be used later. After the literature study is completed, proceed to system analysis and design, then carry out system implementation and system testing, then a conclusion is produced from the research conducted. The results of this study are the design and implementation of the Task Tracker Plus application by implementing the REST API architecture and the MVC pattern using the Golang programming

Keywords: Web Task Tracker, Rest Api, MVC, Golang Programming

DAFTAR ISI

HALAMAN PERNYATAAN TIDAK MELAKUKAN PLAGIAT	ii
LEMBAR PERSETUJUAN TUGAS AKHIR.....	iii
LEMBAR PENGESAHAN NASKAH.....	iv
LEMBAR PENGESAHAN PANITIA PENGUJI	v
PEDOMAN PENGGUNAAN TUGAS AKHIR	vi
KATA PENGANTAR	vii
ABSTRAK.....	viii
ABSTRACT.....	ix
DAFTAR ISI.....	x
DAFTAR GAMBAR	xii
DAFTAR TABEL	xiv
DAFTAR LAMPIRAN.....	xv
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	3
1.3 Tujuan.....	3
1.4 Manfaat.....	3
1.5 Batasan Masalah.....	4
BAB II TINJAUAN PUSTAKA.....	5
2.1 Task Tracker.....	5
2.2 Application Programming Interface (API).....	5
2.3 Rest API	6
2.4 Golang	7
2.5 Konsep Model, View Dan Controller (MVC).....	8
2.6 PostgreSQL	9
2.7 Tailwind CSS	10
BAB III METODE PENELITIAN.....	11
3.1 Tempat dan Waktu Pelaksanaan.....	11
3.2 Bahan dan Alat	11
3.3 Analisis Kebutuhan Penelitian	12

3.4	Prosedur Penelitian	12
BAB IV HASIL DAN PEMBAHASAN		15
4.1	Mind Mapping	15
4.2	Desain Proses	15
4.3	Desain Database	22
4.4	Desain Input dan Output.....	32
4.5	Flowchart Diagram.....	36
4.6	Hasil Implementasi Desain.....	40
4.7	Hasil Pengujian Sistem.....	46
BAB V KESIMPULAN DAN SARAN.....		54
5.1	Kesimpulan.....	54
5.2	Saran	54
DAFTAR PUSTAKA		55
LAMPIRAN.....		58
Lampiran 1 <i>Log Activity</i>		58
Lampiran 2 Dokumen Teknik MSIB PT. Ruang Raya Indonesia		63

DAFTAR GAMBAR

Gambar 2. 1 Konsep Model-View-Controller	9
Gambar 3. 1 Flowchart Penelitian.....	12
Gambar 4. 1 Mind Mapping.....	15
Gambar 4. 2 Entity Relationship Diagram.....	17
Gambar 4. 3 Use Case Diagram.....	18
Gambar 4. 4 Arsitektur Sistem.....	21
Gambar 4. 5 Conceptual Data Model.....	22
Gambar 4. 6 Physical Data Model	23
Gambar 4. 7 Input From Login	32
Gambar 4. 8 Input Form Register	33
Gambar 4. 9 Input Form Category	33
Gambar 4. 10 Input Form Task.....	34
Gambar 4. 11 Desain Output Dashboard	35
Gambar 4. 12 Desain Output Task.....	35
Gambar 4. 13 Desain Output Category	35
Gambar 4. 14 Flowchart Proses Register.....	36
Gambar 4. 15 Flowchart Proses Login.....	37
Gambar 4. 16 Flowchart Proses Tambah Kategori	38
Gambar 4. 17 Flowchart Tambah Task.....	39
Gambar 4. 18 Implementasi Desain Halaman Home.....	40
Gambar 4. 19 Implementasi Desain Halaman Fitur Register	41
Gambar 4. 20 Implementasi Desain Halaman Fitur login.....	41
Gambar 4. 21 Implementasi Desain Halaman Dashboard	42
Gambar 4. 22 Implementasi Desain Halaman Tambah Kategori	42
Gambar 4. 23 Implementasi Desain Halaman List Kategori	43
Gambar 4. 24 Implementasi Desain Modal Success Add Category	43
Gambar 4. 25 Implementasi Desain Modal Delete Category	43
Gambar 4. 26 Implementasi Desain Halaman Tambah Task.....	44
Gambar 4. 27 Implementasi Desain Halaman Daftar Task.....	44
Gambar 4. 28 Implementasi Desain Halaman Update Status Task	45

Gambar 4. 29 Implementasi Desain Modal Success Add Task	45
Gambar 4. 30 Implementasi Desain Modal Delete Task	46
Gambar 4. 31 Implementasi Desain Modal Update Status Task	46

DAFTAR TABEL

Tabel 4. 1 Entity Relationship Diagram.....	17
Tabel 4. 2 Users.....	23
Tabel 4. 3 Sessions.....	24
Tabel 4. 4 Categories.....	24
Tabel 4. 5 Tasks	25
Tabel 4. 6 Rancangan Request and Response Register.....	25
Tabel 4. 7 Rancangan Request and Response Login	26
Tabel 4. 8 Rancangan Request and Response Logout	27
Tabel 4. 9 Rancangan Request and Response Task Add	27
Tabel 4. 10 Rancangan Request and Response Task List.....	28
Tabel 4. 11 Rancangan Request and Response Delete Task.....	29
Tabel 4. 12 Rancangan Request and Response Category Add	30
Tabel 4. 13 Rancangan Request and Response Category List.....	31
Tabel 4. 14 Rancangan Request and Response Delete Category.....	31
Tabel 4. 15 Perangkat Uji	47
Tabel 4. 16 Pengujian Fitur register	47
Tabel 4. 17 Pengujian Fitur Login	48
Tabel 4. 18 Pengujian Fitur Logout	49
Tabel 4. 19 Pengujian Fitur Tambah Kategori.....	50
Tabel 4. 20 Pengujian Fitur Hapus Kategori.....	51
Tabel 4. 21 Pengujian Fitur Tambah Task.....	52
Tabel 4. 22 Pengujian Fitur Hapus Task.....	52

DAFTAR LAMPIRAN

Lampiran 1 Log Activity MSIB PT. Ruang Raya Indonesia

Lampiran 2 Dokumen Teknik MSIB PT. Ruang Raya Indonesia

Lampiran 3 Dokumen Kegiatan MSIB PT. Ruang Raya Indonesia

BAB I PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi pada era modern saat ini berkembang begitu pesat, sehingga membuat pengelolaan tugas dan informasi menjadi lebih mudah. Pada organisasi atau individu dalam tuntutan dan kebutuhan yang kompleks, pengelolaan tugas menjadi semakin penting. Pengelolaan tugas seringkali kompleks dan melibatkan banyak detail, langkah, dan ketergantungan antara satu tugas dengan yang lain. Pengelolaan tugas ini dengan cara manual atau menggunakan alat yang kurang efektif dapat menyebabkan kebingungan, kehilangan informasi penting, dan penundaan.

Untuk mengelola tugas dengan baik pengelolaan tugas juga dapat memantau kemajuan tugas secara *real-time* untuk mendapatkan informasi yang akurat tentang status tugas. Tanpa sistem atau alat yang tepat, organisasi atau individu akan kesulitan dalam mengelola tugas dan dapat kehilangan fokus pada tugas-tugas yang penting. Memiliki alat yang untuk pengelolaan tugas yang mudah dan efisien menjadi kunci untuk mengatasi tantangan ini.

Untuk itu pada penelitian kali ini mengusulkan aplikasi web Task Tracker menjadi solusi yang efektif dalam pengelolaan tugas. Dengan menggunakan aplikasi web Task Tracker, organisasi atau individu dapat dengan mudah mengelola tugas-tugas, mengatur jadwal, dan melacak kemajuan tugas secara efektif. Hal ini membantu meningkatkan produktivitas, mengoptimalkan penggunaan sumber daya, dan mencapai hasil yang lebih baik dalam proyek yang dijalankan.

Task Tracker adalah alat yang dirancang khusus untuk membantu pengguna dalam mengelola tugas, mengatur jadwal, melacak kemajuan tugas. Melalui Task Tracker, pengguna dapat membuat daftar tugas, menetapkan tenggat waktu, mengatur prioritas, mengalokasikan sumber daya, dan melacak kemajuan proyek secara *real-time*.

Aplikasi Web Task Tracker Plus yang akan dibangun berbasis REST API dan MVC menggunakan bahasa pemrograman Golang. Aplikasi ini

menggunakan arsitektur REST API (*Representational State Transfer Application Programming Interface*) dan pola *Model-View-Controller* (MVC) sebagai kerangka kerjanya.

Arsitektur REST API dipilih karena dapat membangun antarmuka yang terhubung antara aplikasi web dan sumber daya server dan dapat menyediakan *endpoint* yang mewakili tindakan spesifik, seperti membuat tugas, memperbarui status tugas, atau menghapus tugas. Selain itu, REST API juga memungkinkan integrasi dengan aplikasi lain melalui permintaan HTTP yang dapat dilakukan secara terstruktur dan konsisten.

Pola *Model-View-Controller* (MVC) dipilih karena memudahkan untuk mengorganisir kode dan logika aplikasi. Dalam pola ini, Model bertanggung jawab untuk mengelola data dan logika bisnis, View bertanggung jawab untuk menampilkan informasi kepada pengguna, dan Controller bertanggung jawab untuk menerima input dari pengguna, memprosesnya, dan mengubah model atau view yang sesuai. Dengan menggunakan pola MVC dapat memisahkan tugas yang berbeda dan memungkinkan pengembangan yang lebih modular dan mudah dipelihara.

Bahasa pemrograman Golang dipilih sebagai bahasa utama dalam mengimplementasikan aplikasi Task Tracker Plus karena memiliki performa yang tinggi, sederhana, dan dapat diandalkan. Bahasa ini juga mendukung pemrograman konkuren dengan rutin (*goroutine*) yang memungkinkan pengelolaan operasi paralel secara efisien. Go juga memiliki manajemen memori yang baik dan mudah dikembangkan. Dengan menggunakan Go, aplikasi Task Tracker Plus dapat memberikan kinerja yang responsif dan efisiensi dalam penggunaan sumber daya.

Dengan menggunakan bahasa pemrograman Go, pengembang dapat membangun aplikasi web Task Tracker yang tangguh, cepat, dan aman. Go juga menyediakan paket-paket yang kuat untuk mengimplementasikan RESTful API dan mendukung pengembangan dengan pola MVC. Dengan demikian, menggunakan Go sebagai bahasa pemrograman dalam

pengembangan aplikasi web Task Tracker dapat memberikan manfaat signifikan dalam hal efisiensi, performa, dan kemudahan pengembangan.

1.2 Rumusan Masalah

Berdasarkan uraian latar belakang diatas, penelitian ini dapat merumuskan beberapa pertanyaan seperti:

1. Bagaimana merancang dan mengimplementasikan arsitektur REST API yang efektif untuk mendukung fungsi utama dari aplikasi Task Tracker Plus?
2. Bagaimana menerapkan pola MVC dalam pengembangan aplikasi Task Tracker Plus untuk memisahkan logika bisnis, tampilan pengguna, dan manipulasi data?
3. Bagaimana mengoptimalkan aplikasi web Task Tracker Plus menggunakan bahasa pemrograman Go untuk mencapai performa yang tinggi dan kinerja yang responsif?

1.3 Tujuan

Berikut adalah tujuan dari penelitian ini:

- 1) Merancang dan mengimplementasikan arsitektur REST API yang efektif untuk mendukung fungsi utama dari aplikasi Task Tracker Plus
- 2) Menerapkan menerapkan pola MVC dalam pengembangan aplikasi Task Tracker Plus untuk memisahkan logika bisnis, tampilan pengguna, dan manipulasi data
- 3) Mengoptimalkan aplikasi web Task Tracker Plus menggunakan bahasa pemrograman Go untuk mencapai performa yang tinggi dan kinerja yang responsif.

1.4 Manfaat

Berikut adalah manfaat dari penelitian ini:

1. Pengguna dapat dengan mudah membuat, mengedit, dan menghapus tugas, serta mengatur tenggat waktu, prioritas, dan status tugas.
2. Pengguna dapat memantau kemajuan proyek secara *real-time*
3. Pengguna dapat mengakses aplikasi dari berbagai perangkat dan lokasi

1.5 Batasan Masalah

Berikut batasan masalah pada penelitian ini:

1. Proses mengelola task
2. Proses mengelola kategori
3. Proses menyimpan data task dan kategori
4. Proses menandai tugas sebagai selesai

BAB II

TINJAUAN PUSTAKA

2.1 Task Tracker

Task tracker, atau sistem pelacakan tugas, adalah alat atau aplikasi yang digunakan untuk melacak dan memantau status serta perkembangan tugas-tugas dalam suatu proyek. Tujuannya adalah memberikan visibilitas yang jelas terkait dengan siapa yang bertanggung jawab atas tugas tertentu, di mana tugas berada dalam alur kerjanya, dan kapan tugas harus selesai. Dengan fitur-fitur seperti daftar tugas, penugasan dan penjadwalan membantu tim mengorganisir dan mengelola tugas secara efisien, memastikan transparansi, dan menghindari tugas yang terlupakan atau terlewat (Tawfik, 2023).

2.2 Application Programming Interface (API)

API (*Application Programming Interface*) adalah antarmuka aplikasi yang berisi kumpulan instruksi yang disimpan dalam sebuah library dan menjelaskan sebuah aplikasi dapat saling berkomunikasi untuk berbagi data aplikasi lainnya (Firdaus et al., 2019). (Firdaus et al., 2019).

Menurut Hradil dan Sklenak (2017), API adalah suatu jembatan komunikasi yang memungkinkan sebuah aplikasi berinteraksi dengan aplikasi lain dan bertukar data. Dengan adanya banyak jenis sistem aplikasi dan variasi kebutuhan pengguna, meskipun memiliki tujuan serta informasi yang sama diperlukan dalam proses tersebut. Untuk mencapai keseragaman pengolahan data ini, API dapat menjadikan mekanisme standar guna menyederhanakan layanan pertukaran antara kedua belah pihak. Melalui metode ini, para pengembang bisa memberikan berbagai macam dataset kepada developer lain agar nantinya mereka bisa menggunakannya secara mumpuni di dalam platform-aplikasinya sendiri.

Dalam penggunaan API, terdapat intruksi terperinci dengan penjelas untuk pemanggilan fungsi parameter, format dan response. API terdiri dari tiga elemen penting yaitu *tools*, *function*, *tools* dan *protocol* dimana mendukung metode *create*, *update*, *read* dan *delete* yang diakses melalui *protocol* HTTP.

API mempunyai banyak manfaat, seperti menggabungkan system aplikasi dengan layanan pihak ketigs, mengakses data dari server diluar aplikasi sistem dan memberikan akses kepada developer untuk mengembangkan aplikasi dengan menggunakan fitur-fitur yang sudah tersedia.

2.3 Rest API

REST API (*Representational State Transfer Application Programming Interface*) adalah pendekatan pengembangan API yang mengimplementasikan arsitektur REST (*Representational State Transfer*). REST API adalah standar dari sebuah gaya arsitektur pada pengembangan sistem berbasis web yang digunakan sebagai protokol untuk komunikasi data. Pada REST API, REST server menyediakan sebuah data yang dapat diakses oleh REST client, data akan diidentifikasi oleh URLs (*Universal Resource Identifiers*) dan umumnya ditampilkan dalam bentuk JSON.

Tujuan dari penggunaan REST API adalah untuk menyederhanakan metode dalam transmisi data melalui jaringan. REST mengadopsi prinsip-prinsip desain yang memungkinkan sistem terdistribusi dan terkoneksi secara efisien (Kristanto, 2020). Saat mentransmisikan data, API ditransmisikan melalui antarmuka standar yaitu HTTP (Akbar, 2018).

Pada arsitektur REST terdapat beberapa komponen yaitu sebagai berikut (Perdana, 2018):

1. URI design

REST API menggunakan protokol HTTP dalam mengakses sumber daya. Untuk mengakses sumber daya pada API, dibutuhkan *Uniform Resource Identifier* (URI) atau endpoint.

2. HTTP verbs

Bagian ini dapat digambarkan juga sebagai pendekatan yang dilakukan oleh client saat melakukan permintaan pada server. Terdapat berbagai metode dalam HTTP, meskipun beberapa metode yang sering dipakai mencakup GET, POST, PUT, dan DELETE.

3. HTTP response code

Komponen ini adalah kode standar dalam protokol HTTP saat client membuat permintaan. Ada tiga jenis kode yang biasa digunakan di REST API yaitu 200 menandakan request yang diminta client berhasil, 400 request yang diminta client gagal dan 500 menandakan request gagal pada sisi server.

4. Format response

Setiap permintaan yang dilakukan oleh client, server akan mengembalikan respon berupa data atau informasi. Ada dua format respons umum yang digunakan dalam REST API: *Extensible Markup Language* (XML) dan *JavaScript Object Notation* (JSON).

2.4 Golang

Bahasa pemrograman Golang (*Google Language*) merupakan hasil pengembangan oleh Google pada tahun 2009 dengan kolaborasi dari Ken Thompson, Robert Griesemer dan Rob Pike. Dalam proses pengembangannya, tujuannya adalah menciptakan bahasa yang unggul dalam aspek kecepatan, kehandalan, skalabilitas dan kesederhanaan (Sari & Hidayat ,2022). Sebagai sebuah bahasa program, Golang mengintegrasikan antara kemampuan keamanan dan efisiensi untuk sistem open source. Selain itu, juga memberi dukungan terhadap konkurensi melalui implementasi sederhananya. Garansi kualitas sistem ini ditunjang dengan adanya fitur *Garbage Collection* berupa *perocess built-in collector* (Goroutines). Oleh karena alasan tersebut, Golang menjadi andal serta performa tinggi pada saat digunakan dalam aplikasi skala besar (Ari Kristanto et.al.,2020.).

Berikut ini merupakan alasan mengapa memilih Golang:

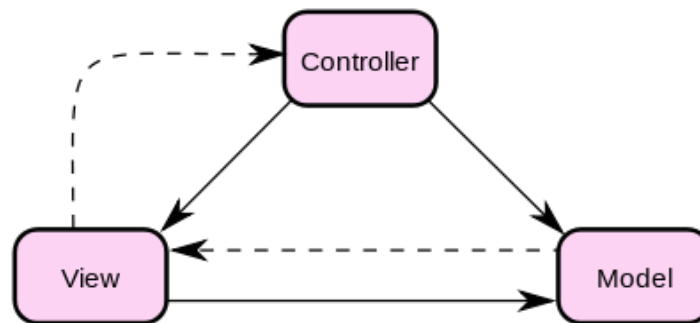
2. Kinerja tinggi: Golang dirancang untuk memberikan performa tinggi dengan dukungan konkurensi dan garbage collector yang efisien.
3. Sederhana dan mudah dipelajari: Golang sederhana dan mudah dipelajari dengan sintaks langsung, dokumentasi lengkap, dan komunitas aktif.

4. Golang dikembangkan untuk mencapai efisiensi penggunaan sumber daya seperti memori dan pemrosesan. Oleh karena itu, bahasa ini cocok digunakan dalam lingkungan dengan keterbatasan sumber daya.
5. Pemrograman konkuren : Golang menawarkan fitur-fitur konkurensi yang sederhana, seperti goroutine dan channel, untuk mempermudah pengembangan aplikasi secara efisien dalam aspek konkurensi.
6. Skalabilitas dan pemeliharaan : Golang memberikan dukungan untuk pemrograman modular yang memungkinkan aplikasi dengan ukuran besar dapat dikembangkan dan dipelihara tanpa kesulitan.
7. Ekosistem : Waktu yang relatif muda, Golang tetap memiliki ekosistem yang terus berkembang dengan berbagai macam pustaka, kerangka kerja, dan alat pengembangan yang tersedia.

Dengan keunggulan-kegunaan ini, telah menjadi pilihan populer bagi para pengembang yang mencari bahasa pemrograman yang mudah digunakan, efisien serta dapat mengatasi tuntutan dalam pengembangan aplikasi modern.

2.5 Konsep Model, View Dan Controller (MVC)

Dalam sebuah aplikasi, MVC adalah pola desain yang membagi menjadi tiga komponen. Tujuannya adalah untuk memisahkan representasi internal informasi dari tampilannya kepada pengguna. Komponen Model mengelola data aplikasi dan meringkas data yang disediakan oleh Controller. Di sisi lain, komponen Tampilan merepresentasikan informasi keluaran dalam bentuk teks atau grafik kepada pengguna, menampilkan laporan dan informasi umum sesuai kebutuhan. View biasanya adalah template JSP. Controller berguna untuk meminta Model untuk menjalankan logika bisnis dan kemudian memperbaharui View berdasarkan output Model (Ismail, 2019).



Gambar 2. 1 Konsep Model-View-Controller

Gambar 2.1 menggambarkan jalur data dalam pola desain MVC. Ketika pengguna membuat permintaan, Controller akan menerima dan berkomunikasi dengan Model untuk mendapatkan atau memperbarui data yang dibutuhkan. Setelah itu, Controller akan memperbarui tampilan melalui View menggunakan data relevan, kemudian respons dikirim kembali kepada pengguna.

Pola desain MVC membagi tanggung jawab dan menjaga kebersihan kode dengan baik. Ini membolehkan pengembang untuk mengelola dan merubah setiap komponen secara terpisah tanpa menyebabkan gangguan pada komponen lainnya.

2.6 PostgreSQL

PostgreSQL adalah sistem database objek relasional *open source* yang terkenal dan kuat. Ini menggabungkan bahasa SQL dengan banyak fitur untuk menyimpan dan menskalakan data kompleks dengan aman. PostgreSQL berawal mula dari tahun 1986 sebagai bagian dari proyek POSTGRES di the University of California, Berkeley, dan sampai saat ini sudah lebih dari 30 tahun aktif dalam mengembangkan platform utamanya (Prayogo, 2019).

Praba dan Savitri (2020) dalam jurnal mereka menarik kesimpulan berdasarkan hasil percobaan bahwa performansi respon time PostgreSQL lebih baik daripada MySQL. Oleh karena itu, dalam pembuatan database dalam skala besar dan dengan jumlah record yang banyak, PostgreSQL menjadi solusi yang lebih baik karena terbukti lebih unggul dalam beberapa query dalam penelitian ini.

2.7 Tailwind CSS

Tailwind CSS adalah framework CSS (*Cascading Style Sheets*) yang memberikan alternatif, dibandingkan dengan gaya CSS tradisional, dibandingkan dengan framework populer lainnya, Tailwind CSS saat ini memperoleh rasio kepuasan terbesar di antara pengembang dengan trend yang masih meningkat yang memberikan kelas CSS utility (Klimm, 2021).

Salah satu keunggulan utama dari Tailwind CSS adalah tingkat fleksibilitas yang sangat tinggi. Dengan adanya berbagai macam kelas utilitas yang tersedia, pengembang dapat dengan mudah menyusun tampilan dan gaya elemen sesuai dengan kebutuhan mereka sendiri. Berkat deskripsi nama-nama kelas utilitas tersebut, para pengembang juga bisa memperoleh pemahaman dan manfaat langsung secara cepat. Keuntungan ini juga membantu meningkatkan kolaborasi tim karena setiap anggota tim mampu mengerti serta menggunakan kelompok-kelompok kelas utilitas tertentu tanpa kesulitan apapun.

BAB III

METODE PENELITIAN

3.1 Tempat dan Waktu Pelaksanaan

Tempat pelaksanaan dari pengembangan aplikasi *Task Tracker Plus* berbasis *Web* dengan menggunakan *Rest Api* dan *Golang* adalah PT. Ruang Raya Indonesia (Ruang Guru). Sedangkan untuk waktu pelaksanaan selama 1 bulan sesuai dengan waktu pelaksanaan tugas akhir pada program studi independent kampus merdeka.

3.2 Bahan dan Alat

Bahan dan alat yang digunakan dalam penelitian Aplikasi Web Task Tracker Plus adalah

1. Perangkat keras komputer (*Hardware*)

Berikut merupakan kebutuhan non fungsional *hardware* yang digunakan dalam sistem ini:

- 1) Processor: Intel(R) Core (TM) i7-8565U CPU @ 1.80GHz 1.99 GHz
- 2) Memory: 8.00 GB (7.85 GB usable)
- 3) Harddisk: 240GB

2. Perangkat lunak komputer (*Software*)

Berikut merupakan kebutuhan non fungsional *hardware* yang digunakan dalam sistem ini:

- 1) OS: Windows 10
- 2) Golang
- 3) Microsoft Office
- 4) PostgreSQL
- 5) Visual Studio Code

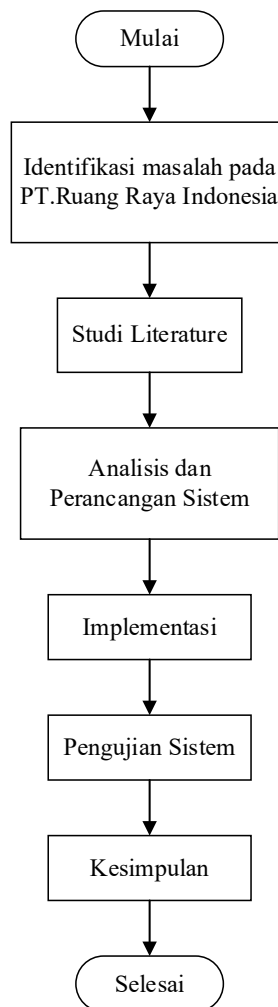
3.3 Analisis Kebutuhan Penelitian

Berikut merupakan analisis kebutuhan yang akan digunakan pada pembuatan sistem:

- Sistem dapat mengelola task
- Sistem dapat mengelola kategori
- Sistem dapat menyimpan data task dan kategori
- Sistem dapat menandai tugas sebagai selesai

3.4 Prosedur Penelitian

Prosedur penelitian yang akan dilakukan pada penelitian ini adalah seperti pada gambar *flowchart* penelitian berikut:



Gambar 3. 1 *Flowchart* Penelitian

Pada Gambar 3.1 menjelaskan alur dari penelitian yang akan dilakukan, pertama yang dilakukan adalah identifikasi masalah pada PT.Ruang Raya Indonesia dan merumuskan masalah kemudian melakukan studi *literature* mengenai topik atau masalah yang sesuai dengan identifikasi masalah dan solusi yang akan digunakan nanti. Setelah studi *literature* selesai lanjut pada analisis dan perancangan sistem, selanjutnya dilakukan implementasi sistem dan melakukan pengujian sistem, kemudian dihasilkan sebuah kesimpulan dari penelitian yang dilakukan.

Pada analisis dan perancangan sistem, implementasi dan pengujian sistem terdapat beberapa tahap yang harus dilakukan, sebagai berikut.

3.4.1 Mind Mapping

Mind map merupakan salah satu metode pembelajaran yang memanfaatkan instrumen yang dapat membantu memetakan isi atau materi sehingga lebih mudah dipelajari dan dianalisis (Kustian, 2021). *Mind mapping* pada penelitian ini menggambarkan struktur dan fitur utama dari aplikasi task tracker.

3.4.2 Desain Proses

Desain proses pada penelitian ini akan menggambarkan kebutuhan sistem yang pertama fungsional yaitu tindakan atau proses yang dilakukan oleh sistem perangkat lunak dan informasi yang dihasilkan oleh sistem, kedua non fungsional yaitu persyaratan yang tidak terkait langsung dengan fitur atau perilaku utama dari sistem atau aplikasi.

Desain proses juga menggambarkan *entity relationship diagram* (ERD) dari *database* yang menunjukkan relasi antar entitas *database* pada aplikasi yang akan dibuat, *use case diagram* digunakan untuk mengetahui aliran aktivitas dari sebuah pelaku atau aktor yang terlibat didalam sistem. Arsitektur sistem merupakan komponen sebuah sistem yang terdiri dari jaringan, perangkat keras dan lunak yang distrukturkan dan pada penelitian ini akan menggambarkan kebutuhan yang diperlukan sistem (Subaeki, 2022).

3.4.3 Desain Database

Desain *database* pada penelitian ini diperlukan untuk merancang kebutuhan *database* yang akan digunakan dalam sistem dari pembuatan

conceptual data model yang kemudian akan di generate menjadi *physical data model* yang mana akan menghasilkan model entitas beserta relasi yang nanti untuk membuat database, menjabarkan struktur tabel dan perancangan *request* dan *response* API digunakan sebagai aturan permintaan data dan pengembalian dari layanan.

3.4.4 Desain *Input* dan *Output*

Desain *input* dan *output* pada penelitian ini digunakan untuk merancang desain awal dari sistem yang akan dibangun. Desain *input* merupakan bentuk dari suatu tampilan dari suatu program. pada sistem Aplikasi Task Tracker Plus terdapat beberapa *input interface*. Sedangkan desain *output* merupakan perancangan desain laporan yang merupakan hasil dari data dari proses yang terjadi, yang tersimpan pada *database*.

3.4.5 *Flowchart* Diagram

Flowchart diagram merupakan aliran proses yang menggambarkan urutan langkah-langkah dan pengambil keputusan dalam suatu proses dari sebuah program. *Flowchart* diagram pada penelitian ini akan ada beberapa langkah yaitu *flowchart register*, *login*, tambah kategori, tambah task.

3.4.6 Implementasi Desain

Implementasi merupakan tahapan yang menunjukkan hasil dari perancangan yang telah dilakukan pada tahap sebelumnya. Implementasi desain pada penelitian ini merupakan implementasi antarmuka atau *interface* dari aplikasi yang akan dibangun yaitu Task Tracker berbasis web.

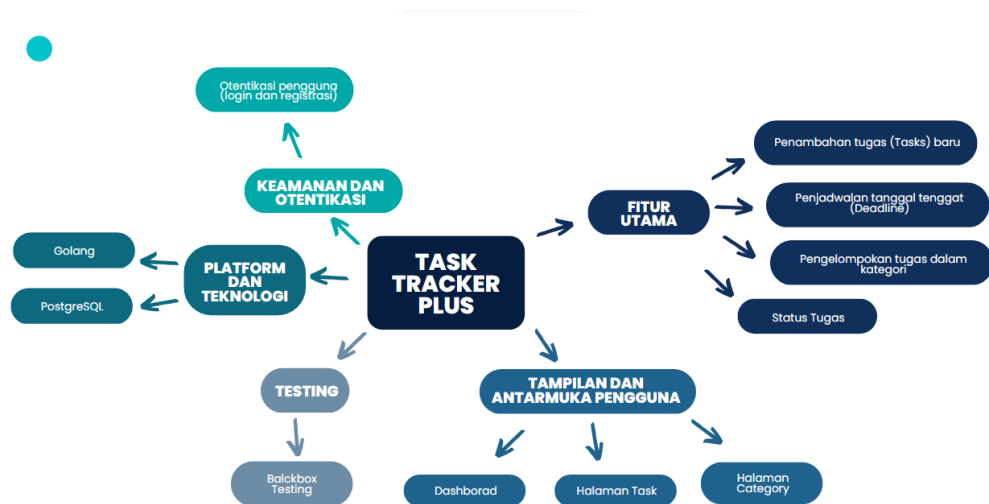
3.4.7 Pengujian Sistem

Pengujian sistem ini merupakan prosedur penelitian tahap terakhir dimana akan menunjukan hasil dari pengujian sistem pada aplikasi task tracker yang menggunakan metode *black box testing* yang mana akan dilakukan pengujian dari *interface* yang telah terimplementasi pada implementasi desain.

BAB IV

HASIL DAN PEMBAHASAN

4.1 Mind Mapping



Gambar 4. 1 *Mind Mapping*

Pada gambar 4.1 merupakan *mind mapping* untuk aplikasi "Task Tracker Plus" adalah representasi visual yang digunakan untuk menggambarkan struktur dan fitur utama dari aplikasi tersebut.

4.2 Desain Proses

Dengan desain proses yang baik, aplikasi task tracker dapat membantu dalam mengelola tugas dengan lebih terstruktur dan mengoptimalkan produktivitas.

4.2.1 Fungsional

Kebutuhan fungsional adalah jenis kebutuhan yang menggambarkan tindakan atau proses yang dilakukan oleh sistem perangkat lunak dan informasi yang dihasilkan oleh sistem tersebut. Dalam hal ini, kebutuhan fungsional berfokus pada apa yang sistem dapat lakukan dan hasil yang diharapkan dari tindakan-tindakan tersebut. Adapun proses dan informasi yang dihasilkan adalah:

1. Pengguna dapat membuat task baru, melihat daftar task, mengedit task, dan menghapus task yang sudah tidak diperlukan.
2. Pengguna dapat membuat kategori baru, melihat daftar kategori, dan menghapus kategori.
3. Sistem dapat menyimpan data task dan kategori yang dimasukkan oleh pengguna dalam database untuk pengelolaan yang lebih lanjut.
4. Pengguna dapat menandai tugas sebagai selesai (completed) ketika task telah selesai dikerjakan.

4.2.2 Non Fungsional

Kebutuhan non-fungsional adalah persyaratan yang tidak terkait langsung dengan fitur atau perilaku utama dari sistem atau aplikasi, tetapi merupakan aspek penting dari desain yang mempengaruhi performa, dan pengalaman pengguna secara keseluruhan

1. Perangkat Lunak

Perangkat yang diperlukan dalam pembuatan sistem dan pemrosesan data dari sistem yang diusulkan adalah:

- a. Microsoft Windows 10 sebagai sistem operasi.
- b. Golang untuk bahasa pemrograman.
- c. PostgreSQL sebagai program RDBMS.
- d. Microsoft Office sebagai pembuatan laporan-laporan.

2. Perangkat Keras

Perangkat keras yang dibutuhkan sebagai penunjang berupa seperangkat Laptop dengan spesifikasi sebagai berikut:

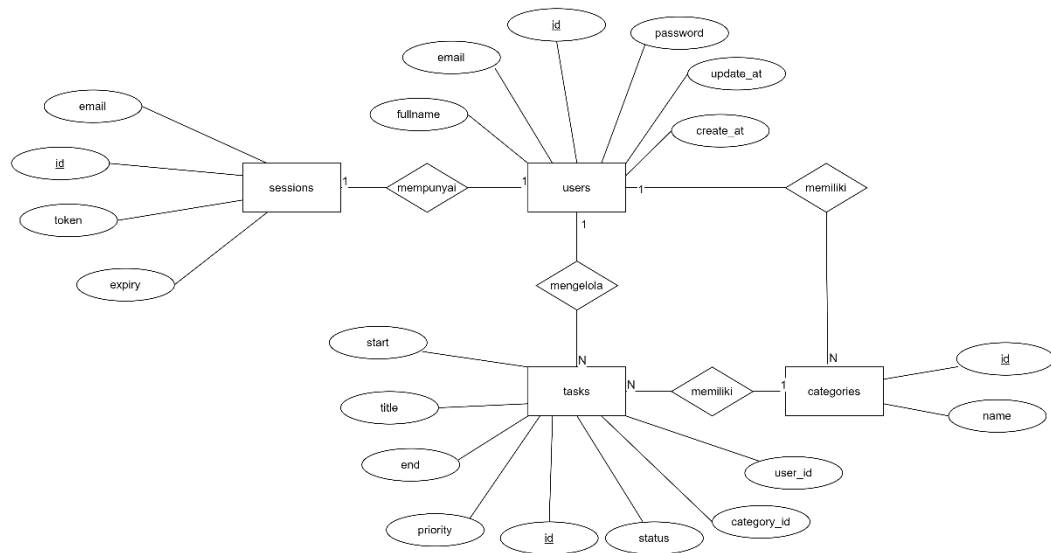
- a. AMD A9-9425 (2 cores at 3.1 Ghz / up to 3.7 Ghz / 1MB Cache)
keatas atau yang sekelas
- b. RAM minimal 4 GB
- c. SSD 240GB

3. Brainware

Sebagai pelaksana personal atau pegawai yang bertugas menggunakan aplikasi task tracker. Sebagai pengguna aplikasi task

tracker, tugas seorang pengguna akan berfokus pada mengelola dan melaksanakan tugas-tugas yang telah ditetapkan oleh sistem.

4.2.3 ERD (*Entity Relationship Diagram*)



Gambar 4. 2 *Entity Relationship Diagram*

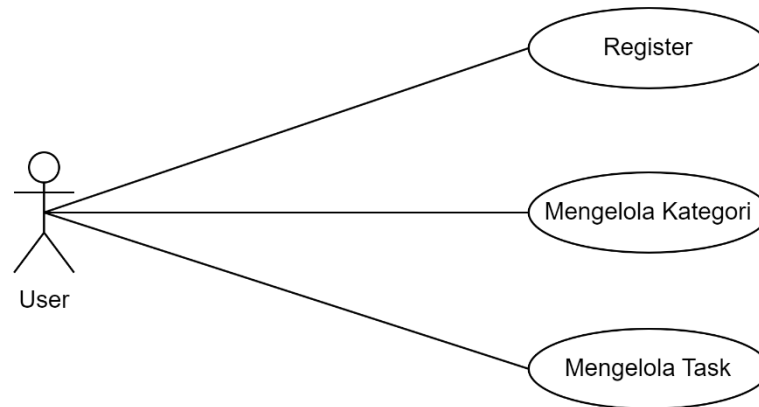
Gambar di atas merupakan gambaran hubungan antar entitas yang satu dengan yang lain yang saling berhubungan dan saling berinteraksi satu dengan yang lain. ERD adalah diagram yang digunakan untuk perancangan database dan menunjukkan relasi antar object beserta atributnya.

Tabel 4. 1 *Entity Relationship Diagram*

No	Entitas	Atribut
1	users	id, fullname, email, password, created_at, update_at
2	sessions	id, token, email, expiry
3	task	id, title, priority, status, category_id, user_id, start, end
4	categories	id, name, usser_id

Tabel 4.1 menunjukkan entitas dan atribut dari ERD yang memiliki empat entitas yaitu user, sessions, task, dan categories. Dan setiap entitas memiliki masing-masing id.

4.2.4 Use case diagram



Gambar 4. 3 *Use Case Diagram*

Diagram *Use Case* adalah diagram yang menggambarkan aliran aktivitas dari sebuah pelaku atau aktor yang terlibat didalam sistem. Pada gambar dibawah ini merupakan diagram use case dengan actor user. User dapat melakukan register, mengelola kategori dan mengelola task. adapun penjelasannya sebagai berikut ini:

1) User

Orang yang dapat mengakses atau menggunakan aplikasi Task Tracker Plus, mulai dari register ke aplikasi hingga melakukan aksi terhadap aplikasi seperti mengelola kategori dan mengelola task.

2) Register

Register merupakan langkah pertama yang dilakukan user ketika ia tidak mempunyai akses pada aplikasi Task Tracker Plus. Mendaftarkan data diri ke dalam aplikasi agar dikenali.

3) Mengelola Kategori

Pada tahap ini mencakup aktivitas-aktivitas yang dapat dilakukan oleh pengguna dalam mengelola kategori pada sistem.

a. Create Kategori

- Deskripsi: Pengguna dapat membuat kategori baru dengan mengisi informasi seperti nama, dan pengguna user_id.

- Aksi: Pengguna memasukkan informasi kategori.
- Reaksi Sistem: Sistem menyimpan kategori baru ke dalam tabel "categories" dengan informasi yang dimasukkan pengguna.

b. Delete Kategori

- Deskripsi: Pengguna dapat menghapus kategori yang ada dari sistem.
- Aksi: Pengguna memilih kategori yang ingin dihapus.
- Reaksi Sistem: Sistem menghapus kategori yang dipilih dari tabel "categories".

c. View Kategori

- Deskripsi: Pengguna dapat melihat daftar kategori yang ada dalam sistem.
- Aksi: Pengguna meminta daftar kategori.
- Reaksi Sistem: Sistem menampilkan daftar kategori yang ada dalam tabel "categories".

4) Mengelola Task

Pada tahap ini mencakup aktivitas-aktivitas yang dapat dilakukan oleh pengguna dalam mengelola task pada sistem.

a. Create Task

- Deskripsi: Pengguna dapat membuat task baru dengan mengisi informasi seperti judul (title), tanggal mulai (start), tanggal berakhir (end), prioritas (priority), status (status), kategori (category_id), dan pengguna yang bertanggung jawab (user_id).
- Aksi: Pengguna memasukkan informasi task.
- Reaksi Sistem: Sistem menyimpan task baru ke dalam tabel "tasks" dengan informasi yang dimasukkan pengguna.

b. Delete Task

- Deskripsi: Pengguna dapat menghapus task yang ada dari sistem.
- Aksi: Pengguna memilih task yang ingin dihapus.
- Reaksi Sistem: Sistem menghapus task yang dipilih dari tabel "tasks".

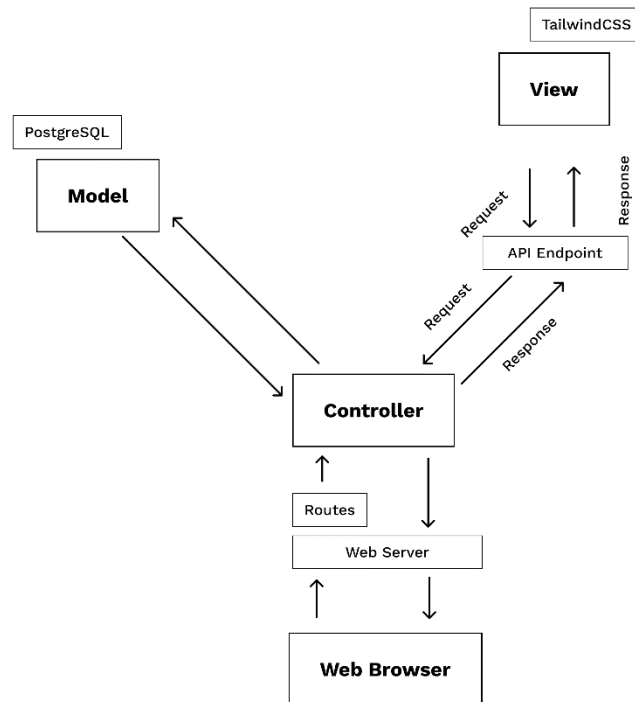
c. View Task

- Deskripsi: Pengguna dapat melihat daftar task yang ada dalam sistem.
- Aksi: Pengguna meminta daftar task.
- Reaksi Sistem: Sistem menampilkan daftar task yang ada dalam tabel "tasks" beserta informasi detailnya seperti judul, tanggal mulai, tanggal berakhir, prioritas, status, kategori, dan pengguna yang bertanggung jawab.

d. Update Task

- Deskripsi: Pengguna dapat memperbarui informasi task yang sudah ada dalam sistem.
- Aksi: Pengguna memilih task yang ingin diperbarui dan mengisi informasi yang ingin diubah yaitu kolom status.
- Reaksi Sistem: Sistem menyimpan perubahan informasi pada task yang dipilih ke dalam tabel "tasks".

4.2.5 Arsitektur sistem



Gambar 4. 4 Arsitektur Sistem

Berikut ini merupakan penjelasan dari gambar arsitektur sistem yang digunakan:

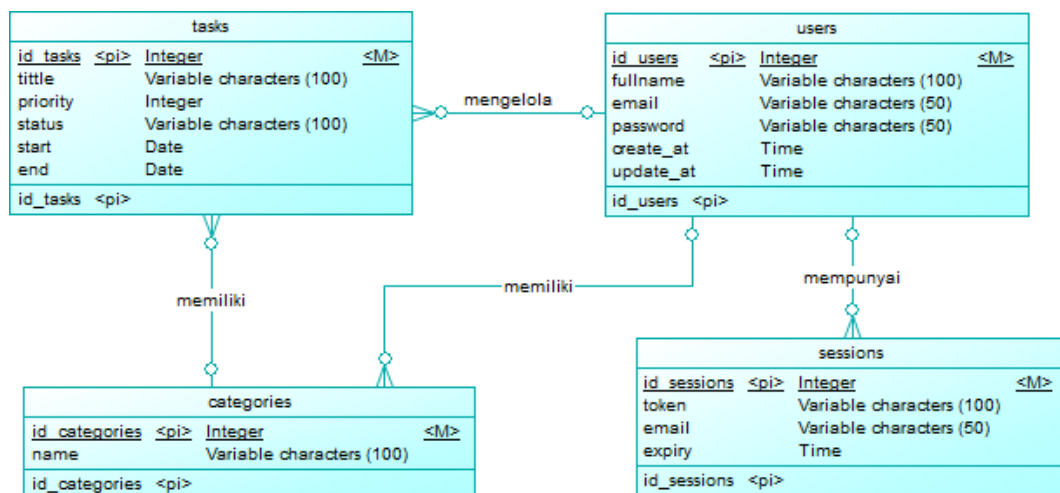
1. **View:** Ini adalah komponen MVC yang digunakan untuk menampilkan kepada pengguna sistem, yang berinteraksi langsung dengan API melalui API. Pada view menggunakan tailwind css sebagai framework CSS yang berbasis utility untuk membuat UI atau tampilan dari aplikasi web
2. **API Endpoints:** Ini adalah endpoint-endpoint API yang dihasilkan oleh sistem monolith. Setiap endpoint mewakili operasi yang ingin dilakukan oleh sistem. Permintaan API akan diteruskan ke Controllers yang sesuai.
3. **Controllers:** Controllers berfungsi sebagai penghubung antara API Endpoints dan Models. Controllers menerima permintaan dari API, memprosesnya, dan memanggil fungsi-fungsi yang sesuai dalam Models untuk memperoleh atau memperbarui data.

4. Models: Models mewakili entitas dalam sistem. Mereka bertanggung jawab untuk logika bisnis dan akses ke Data Store. Models mengambil data yang diperlukan, memanipulasi data, dan menyimpannya ke Data Store.
5. PostgreSQL: PostgreSQL digunakan sebagai database. Models berkomunikasi dengan PostgreSQL untuk mengambil, menyimpan, atau memperbarui data yang diperlukan

4.3 Desain Database

4.3.1 Conceptual Data Model (CDM)

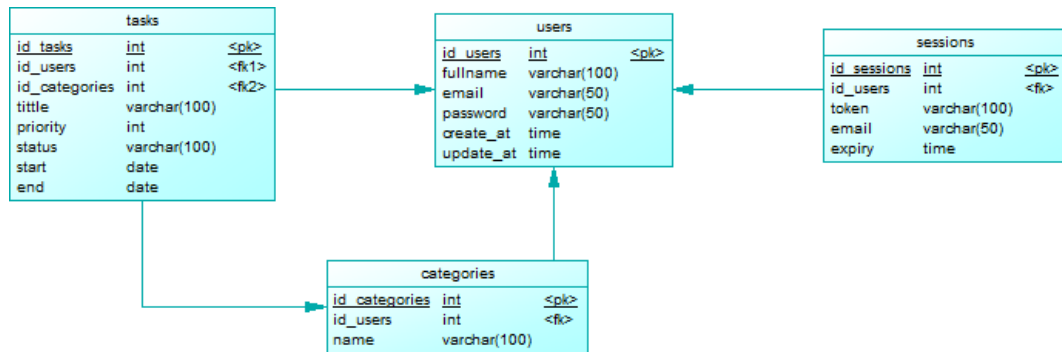
Conceptual data model merupakan gambaran secara keseluruhan struktur *database* yang telah dirancang seperti gambar 4.5, terdapat 4 tabel yaitu tabel *users*, tabel *sessions*, tabel *categories*, dan tabel *tasks*.



Gambar 4. 5 Conceptual Data Model

4.3.2 Physical Data Model (PDM)

Pada *physical data model* merupakan gambaran secara detail konsep rancangan struktur *database* dan merupakan hasil *generate* dari CDM dan gambar berikut merupakan hasil dari *generate* tabel *users*, tabel *sessions*, tabel *categories*, dan tabel *tasks*.

Gambar 4. 6 *Physical Data Model*

4.3.3 Struktur Tabel

Berikut adalah struktur tabel dari Database Aplikasi Tracker Plus:

a. Nama tabel: users

Tabel 4.2 merupakan tabel user untuk hak akses user pengguna sistem aplikasi tracker plus.

Tabel 4. 2 *Users*

No	Field	Type	Keterangan
1	id_users	Integer	Primary key dan auto increment
2	Fullname	Varchar 100	nama lengkap pengguna
3	Email	Varchar 50	email pengguna
4	Password	Varchar 50	Password login
5	create_at	time	waktu kapan dibuat
6	update_at	time	waktu kapan diubah

Tabel 4.2 merupakan stuktur tabel untuk users yang memiliki 6 atribut yaitu id_users, fullname, email, password, create_at, dan update_at.

b. Nama Tabel: sessions

Tabel 4.3 merupakan tabel session dimana data akan ditambahkan ketika user melakukan login.

Tabel 4. 3 *Sessions*

No	Field	Type	Keterangan
1	id_sessions	Integer	Primary key dan auto increment
2	id_users	Integer	Foreign key dari tabel users
3	Token	Varchar 100	nama lengkap pengguna
4	Email	Varchar 50	email pengguna dimana foreign key dengan tabel users
5	Expiry	time	menampilkan tanggal kadaluarsa token

Tabel 4.3 merupakan struktur tabel untuk sessions yang memiliki 5 atribut dimana salah satunya merupakan foreign key yaitu id_sessions, id_users sebagai foreign key, token, email, dan expiry.

c. Nama Tabel: categories

Tabel 4.4 merupakan tabel categories untuk penambahan data kategori yang dientrikan.

Tabel 4. 4 *Categories*

No	Field	Type	Keterangan
1	id_categories	Integer	Primary key dan auto increment
2	id_users	Integer	Foreign key dengan tabel users
3	Name	Varchar 100	nama kategori

Tabel 4.4 merupakan struktur tabel dari categories yang memiliki 2 atribut dan satu foreign key yaitu id_categories, id_users dari tabel users dan name.

d. Nama Tabel: tasks

Tabel 4.5 merupakan tabel tasks untuk penambahan data task yang dientrikan.

Tabel 4. 5 *Tasks*

No.	Field	Type	Keterangan
1	id_tasks	Integer	Primary key dan auto increment
2	id_users	Integer	Foreign key dengan tabel users
3	id_categories	Integer	Foreign key dengan tabel categories
4	Title	Varchar 100	judul task
5	End	Time	Tanggal akhir task
6	Priority	Integer	prioritas task
7	Status	Varchar 100	status dari task
8	Start	Date	Tanggal start task

Tabel 4.5 merupakan struktur tabel dari tasks yang memiliki 9 atribut dengan 2 atribut foreign key yaitu id_tasks, id_users foreign key dari tabel users, id_categories foreign key dari tabel categories, title, deadline, priority, status, start, dan end.

4.3.4 Perancangan Request dan Response API

Pada tahap ini penulis membuat sebuah rancangan untuk *request* dan *response* yang digunakan sebagai aturan permintaan data dan pengembalian dari layanan, berikut ini merupakan rancangan *request* dan *response* API yang akan dibuat:

a. Request dan Response Register

Adapun rancangan *request* dan *response* register ditunjukkan pada tabel 4.6.

Tabel 4. 6 Rancangan Request and Response Register

Method	POST
URL	localhost:8080/api/v1/user/register
Headers	-
Body	{ "fullname": "John Doe", "email": "johndoe@example.com", "password": "password123" }
Keterangan	Pada kolom body, terdapat beberapa isian yaitu fullname berisi nama calon pengguna, email berisi

	email calon pengguna dan password berisi password calon pengguna.
Response 200	{ "message": "register success" }
Response 400	{ "error": "invalid data" }
Response 500	{ "error": "internal server error" }

Tabel 4.6 merupakan rancangan *request* dan *response* register berfungsi untuk membuat akun pengguna baru ke dalam sistem.

b. Request dan Response Login

Adapun rancangan *request* dan *response* login ditunjukkan pada tabel 4.7.

Tabel 4. 7 Rancangan *Request and Response Login*

Method	POST
URL	localhost:8080/api/v1/user/login
Headers	-
Body	{ "email": "johndoe@example.com", "password": "password123" }
Keterangan	Pada kolom body, terdapat beberapa isian yaitu email berisi email pengguna dan password berisi password pengguna yang terdaftar di dalam sistem.
Response 200	{ "message": "login success" }
Response 400	{ "error": "email or password is empty" }
Response 500	{ "error": "internal server error" }

Tabel 4.7 merupakan rancangan *request* dan *response* login berfungsi untuk pengguna masuk ke dalam sistem, sehingga pengguna dapat memperoleh hak akses ke sistem.

c. Request dan Response Logout

Adapun rancangan *request* dan *response* logout ditunjukkan pada tabel 4.8.

Tabel 4. 8 Rancangan *Request* and *Response* Logout

Method	POST
URL	localhost:8080/api/v1/user/logout/{id}
Headers	Authorizon
Body	-
Keterangan	Pada kolom URL terdapat parameter id berisi dari nilai id dari user yang akan di logout. Pada kolom header wajib disertakan yaitu authorization, authorization berisi bareer token. Headers authorization dibuat sebagai bukti bahwa pengguna telah melakukan login sehingga dapat mengakses sumber data
Response 200	{ "message": "logout success" }
Response 400	{ "error": "invalid ID user" }
Response 500	{ "error": "internal server error" }

Tabel 4.8 merupakan rancangan *request* dan *response* logout berfungsi untuk pengguna keluar dari otoritas sistem, setelah melakukan logout pengguna tidak memperoleh hak akses ke sistem.

d. Request dan Response Task Add

Adapun rancangan *request* dan *response* add task ditunjukkan pada tabel dibawah 4.9.

Tabel 4. 9 Rancangan *Request* and *Response* Task Add

Method	POST
URL	localhost:8080/api/v1/task/add
Headers	Authorization
Body	{ "title": "Task 1", "start": "2023-06-20", "end": "2023-06-30", "priority": 2, "category id": "1", }

	"status": "In Progress" }
Keterangan	Pada kolom body, terdapat beberapa isian yaitu title yang berisi judul task, start yang berisi tanggal dengan format tanggal, end yang berisi tanggal dengan format tanggal, priority yang berisi prioritas task dengan format integer, category_id yang berisi id dari kategori yang dipilih dan status yaitu berisi status dari task pengguna. Pada kolom header wajib disertakan yaitu authorization, authorization berisi bareer token. Headers authorization dibuat sebagai bukti bahwa pengguna telah melakukan login sehingga dapat mengakses sumber data
Response 201	{ "message": "add task success" }
Response 400	{ "error": "invalid data" }
Response 500	{ "error": "internal server error" }

Tabel 4.9 merupakan rancangan *request* add task berfungsi untuk pengguna membuat task baru.

e. Request dan Response Task List

Adapun rancangan *request* dan *response* task list ditunjukkan pada tabel dibawah 4.10.

Tabel 4. 10 Rancangan *Request* and *Response* Task List

Method	GET
URL	localhost:8080/api/v1/task/list
Headers	Authorization
Body	-
Keterangan	Pada kolom header wajib disertakan yaitu authorization, authorization berisi bareer token. Headers authorization dibuat sebagai bukti bahwa pengguna telah melakukan login sehingga dapat mengakses sumber data
Response 201	[{ "id": 1, "title": "Task 1",

	<pre> "start": "2023-06-20", "end": "2023-06-30", "priority": 2, "category_id": "1", "status": "In Progress" }, { "id": 2, "title": "Task 2", "start": "2023-06-20", "end": "2023-06-30", "priority": 1, "category_id": "2", "status": "Completed" },] </pre>
Response 500	<pre> { "error": " [error messages]" } </pre>

Tabel 4.10 merupakan rancangan *request* task list berfungsi untuk menampilkan daftar task yang tersimpan pada sistem.

f. Request dan Response Delete Task

Adapun rancangan *request* dan *response* delete task ditunjukkan pada tabel dibawah 4.11.

Tabel 4. 11 Rancangan *Request* and *Response* Delete Task

Method	DELETE
URL	localhost:8080/api/v1/task/delete/{id}
Headers	Authorization
Body	-
Keterangan	Pada kolom URL terdapat parameter id berisi dari nilai id dari task yang akan di hapus. Pada kolom header wajib disertakan yaitu authorization, authorization berisi bareer token. Headers authorization dibuat sebagai bukti bahwa pengguna telah melakukan login sehingga dapat mengakses sumber data
Response 201	<pre> { "message": "delete Task success" } </pre>
Response 400	<pre> { "error": "Invalid task ID" } </pre>

Response 500	{ "error": "[error messages]" }
---------------------	---------------------------------------

Tabel 4.11 merupakan rancangan *request* delete task berfungsi untuk pengguna menghapus task yang dipilih berdasarkan id task.

g. Request dan Response Category Add

Adapun rancangan *request* dan *response* add kategori ditunjukkan pada tabel dibawah 4.12.

Tabel 4. 12 Rancangan *Request* and *Response* Category Add

Method	POST
URL	localhost:8080/api/v1/category/add
Headers	Authorization
Body	{ "name": "Category 1", "user_id": "1", }
Keterangan	Pada kolom body, terdapat beberapa isian yaitu name yang berisi nama kategori dan user_id. Pada kolom header wajib disertakan yaitu authorization, authorization berisi bareer token. Headers authorization dibuat sebagai bukti bahwa pengguna telah melakukan login sehingga dapat mengakses sumber data
Response 201	{ "message": "add category success" }
Response 400	{ "error": "invalid data" }
Response 500	{ "error": "internal server error" }

Tabel 4.12 merupakan rancangan *request* add category yang berfungsi untuk pengguna membuat kategori baru.

h. Request dan Response Category List

Adapun rancangan *request* dan *response* category list ditunjukkan pada tabel dibawah 4.13.

Tabel 4. 13 Rancangan *Request and Response Category List*

Method	GET
URL	localhost:8080/api/v1/category/list
Headers	Authorization
Body	-
Keterangan	Pada kolom header wajib disertakan yaitu authorization, authorization berisi bareer token. Headers authorization dibuat sebagai bukti bahwa pengguna telah melakukan login sehingga dapat mengakses sumber data
Response 201	[<pre> { "id": 1, "title": "Category 1", "user_id": "1", }, { "id": 2, "title": " Category 2", "user_id": "1", },] </pre>
Response 500	<pre> { "error": " [error messages]" } </pre>

Tabel 4.13 merupakan rancangan *request category list* yang berfungsi untuk menampilkan daftar category yang tersimpan pada sistem.

i. Request dan Response Delete Categeory

Adapun rancangan *request* dan *response delete category* ditunjukkan pada tabel dibawah 4.14.

Tabel 4. 14 Rancangan *Request and Response Delete Category*

Method	DELETE
URL	localhost:8080/api/v1/category/delete/{id}
Headers	Authorization
Body	-
Keterangan	Pada kolom URL terdapat parameter id berisi dari nilai id dari catgeory yang akan di hapus. Pada kolom header wajib disertakan yaitu authorization, authorization berisi bareer token.

	Headers authorization dibuat sebagai bukti bahwa pengguna telah melakukan login sehingga dapat mengakses sumber data
Response 201	{ "message": "delete Task success" }
Response 400	{ "error": "Invalid task ID" }
Response 500	{ "error": "[error messages]" }

Tabel 4.14 merupakan rancangan *request delete category* yang berfungsi untuk pengguna menghapus *category* yang dipilih berdasarkan id *category*.

4.4 Desain Input dan Output

4.4.1 Desain Input

Desain input adalah gambaran tentang bentuk dari suatu tampilan dari suatu program. pada sistem Aplikasi Task Tracker Plus terdapat beberapa *input interface*.

Login to your account

email

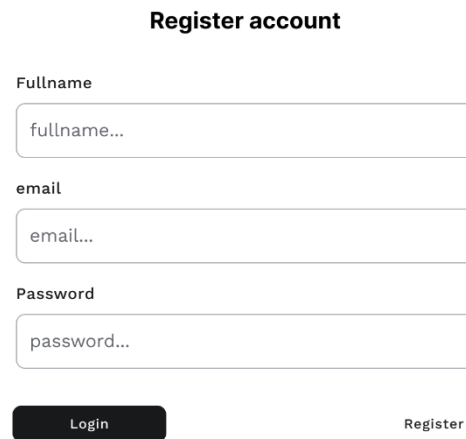
Password

Register
Login

Gambar 4. 7 *Input From Login*

Pada gambar 4.7 merupakan tampilan input formulir login, pada form ini digunakan jika user ingin masuk ke dalam program aplikasi task tracker plus. User harus menginputkan email dan password yang dimiliki,

Jika email dan password benar maka user dapat masuk ke dalam sistem sehingga user dapat mengakses menu-menu yang ada di program.



Register account

Fullname

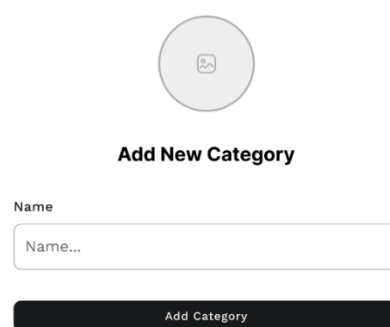
email

Password

Login Register

Gambar 4. 8 *Input Form Register*

Pada gambar 4.8 merupakan tampilan input formulir register, pada form ini digunakan jika user ingin melakukan pendaftaran ke dalam program aplikasi task tracker plus. User harus menginputkan fullname, email dan password yang dimiliki. Jika fullname, email dan password benar maka user berhasil terdaftar pada sistem aplikasi task tracker plus sehingga user dapat melakukan proses login pada program.



Add New Category

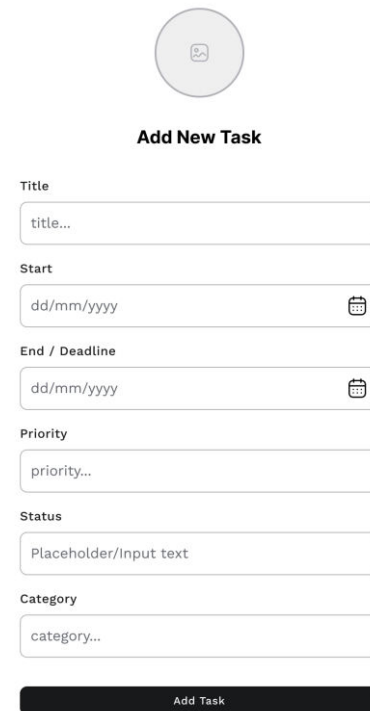
Name

Add Category

Gambar 4. 9 *Input Form Category*

Pada gambar 4.9 merupakan tampilan input formulir kategori, pada form ini digunakan jika user ingin menambah kategori task ke dalam program aplikasi task tracker plus, untuk mengakses form ini user harus login. User harus menginputkan nama kategori yang dimiliki. Jika nama

benar maka user berhasil menambah data kategori pada sistem aplikasi task tracker plus.



Add New Task

Title
title...

Start
dd/mm/yyyy

End / Deadline
dd/mm/yyyy

Priority
priority...

Status
Placeholder/Input text

Category
category...

Add Task

Gambar 4. 10 *Input Form Task*

Pada gambar 4.10 merupakan tampilan input formulir input task, pada form ini digunakan jika user ingin menambah task ke dalam program aplikasi task tracker plus. User harus menginputkan title, deadline, priority, status, kategori_id dan user_id yang dimiliki. Jika title, deadline, priority, status, kategori_id dan user_id berhasil diinputkan pada sistem aplikasi task tracker plus sehingga user dapat memiliki task baru pada program

4.4.2 Desain Output

Desain output merupakan perancangan desain laporan yang merupakan hasil dari data dari proses yang terjadi, yang tersimpan pada database. Pada aplikasi task tracker plus ini, terdapat 3 desain output, yaitu desain output dashboard, task dan kategori.

Dashboard



Lorem ipsum
 user1@mail.com

Task 1

Deadline 10 July 2023

 Completed



Lorem ipsum
 user1@mail.com

Task 2

Deadline 11 July 2023

 Completed



Lorem ipsum
 user1@mail.com

Task 3

Deadline 20 July 2023

 In Progress

Gambar 4. 11 Desain *Output Dashboard*

Task



Task 1
 Priority 2

Category: Desain dan Arsitektur

User ID:2

Start 10 July 2023

End 20 July 2023

 Completed



Task 1
 Priority 2

Category: Desain dan Arsitektur

User ID:2

Start 10 July 2023

End 20 July 2023

 Completed



Task 1
 Priority 2

Category: Desain dan Arsitektur

User ID:2

Start 10 July 2023

End 20 July 2023

 Completed

Gambar 4. 12 Desain *Output Task*

Category

No	Category Name
1	ID: 1 ID User : 1 Category 1
2	ID: 2 ID User : 1 Category 2
3	ID: 3 ID User : 1 Category 3

Gambar 4. 13 Desain *Output Category*

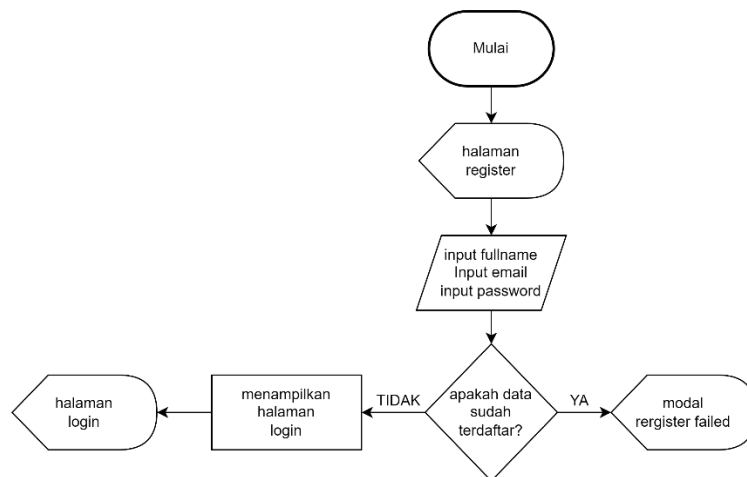
4.5 Flowchart Diagram

Flowchart diagram atau disebut bagan alur merupakan diagram yang menggambarkan urutan langkah-langkah dan pengambil keputusan dalam suatu proses dari sebuah program. Pada gambar dibawah ini merupakan bagan alur pada aplikasi Task Tracker Plus. Berikut ini merupakan penjelasan langkah-langkah dari flowchart pada Aplikasi Task Tracker Plus:

Berikut adalah penjelasan langkah-langkah dalam flowchart untuk aplikasi Task Tracker Plus terdiri menjadi:

a. Flowchart Register

Pada gambar 4.14 merupakan flowchart register :



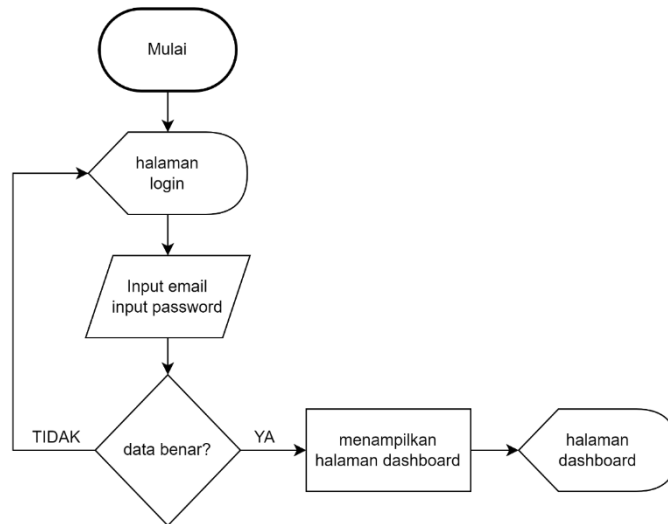
Gambar 4. 14 *Flowchart Proses Register*

Penjelasan:

- Pertama pengguna masuk ke halaman register
- Kemudian pengguna menginput data yang berupa fullname, email dan password
- Jika data yang terdaftar sudah valid maka menampilkan halaman login
- Jika data yang dimasukkan tidak valid maka akan muncul modal register gagal

b. Flowchart Login

Pada gambar 4.15 merupakan flowchart login :



Gambar 4. 15 *Flowchart Proses Login*

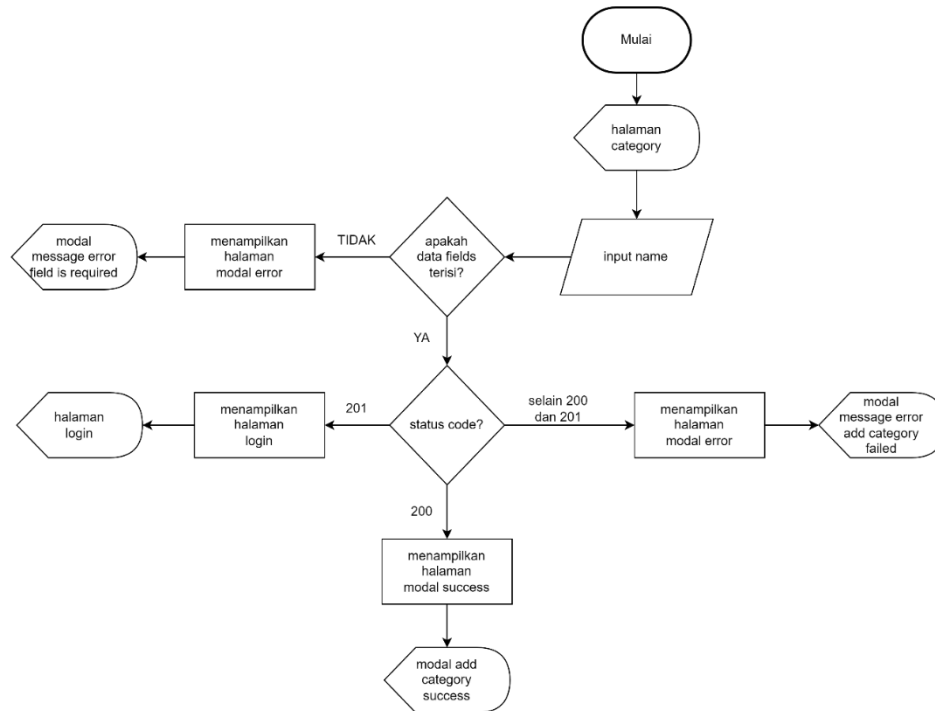
Penjelasan:

- Pertama pengguna masuk ke halaman login
- Selanjutnya pengguna menginputkan data email dan password
- Jika data login yang di inputkan valid, maka akan menampilkan halaman dashboard
- Jika data login tidak valid maka akan kembali ke halaman login
- Selesai

c. Flowchart Tambah Kategori

Pada gambar 4.16 merupakan flowchart tambah data kategori :

Penjelasan:

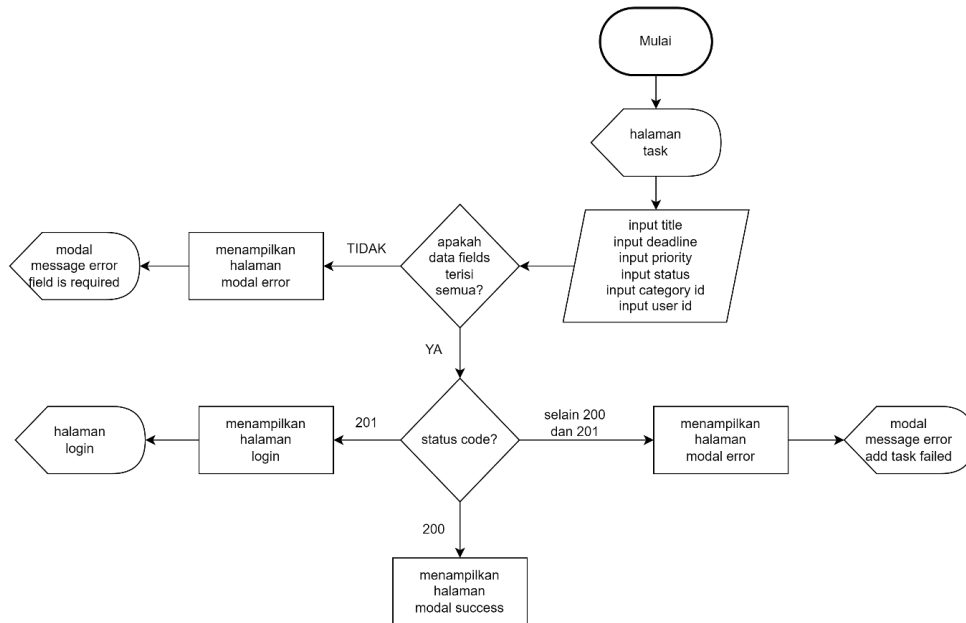


Gambar 4. 16 *Flowchart* Proses Tambah Kategori

- Pertama user masuk ke halaman tambah kategori
- Kemudian user menginputkan data nama ke form tambah kategori
- Jika data tidak terisi maka akan menampilkan modal error, dengan perintah untuk mengisi data form tambah kategori
- Jika data sudah terisi tapi status kode adalah 201, maka akan menampilkan halaman login. Ini menandakan anda belum melakukan autentikasi
- Jika status code adalah selain 200 dan 201 maka akan menampilkan modal error, dimana pesan errornya adalah gagal menambah data kategori
- Jika status kode adalah 200, maka akan menampilkan modal sukses dan data kategori berhasil ditambahkan

d. Flowchart Tambah Task

Pada gambar 4.17 merupakan flowchart tambah data task :



Gambar 4. 17 *Flowchart* Tambah Task

Penjelasan:

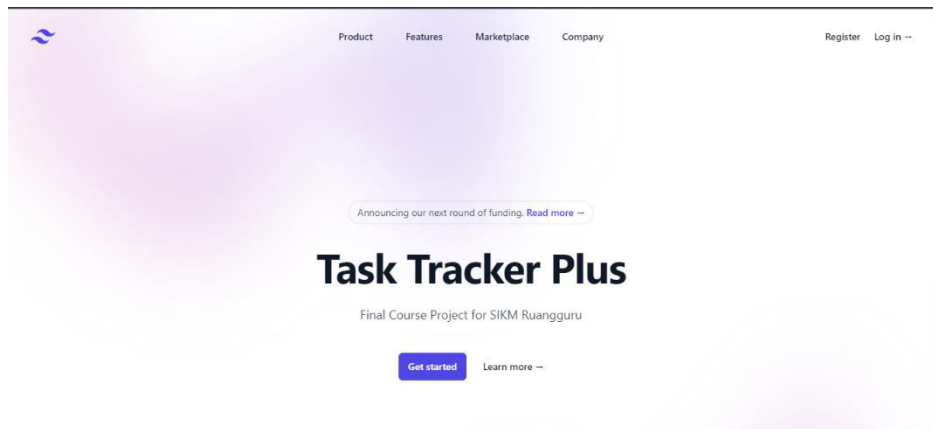
- Pengguna masuk ke halaman task
- Selanjutnya pengguna melakukan input judul, deadline, priority, status, category id dan user id ke form tambah task
- Jika data form belum terisi salah satu atau sama sekali, maka akan menampilkan modal eror dengan pesan inputan harus diisi
- Jika inputan sudah terisi semua tapi status kode adalah 201, maka akan ke halaman login. Hal ini karena pengguna belum melakukan autentikasi
- Jika status kode adalah selain 200 dan 201 maka, akan menampilkan modal error dengan pesan error yaitu tambah pesan gagal
- Jika status kode adalah 200, maka akan menampilkan modal sukses dan data task berhasil disimpan ke database
- Selesai

4.6 Hasil Implementasi Desain

Pada tahap ini, dilakukan implementasi antarmuka pada Aplikasi Task Tracker Plus ditunjukkan dengan hasil *screenshot* antarmuka yang sesuai dengan rancangan yang telah dibuat. Berikut ini merupakan hasil implementasi dari Aplikasi Task Tracker Plus adalah sebagai berikut:

a. Halaman Home

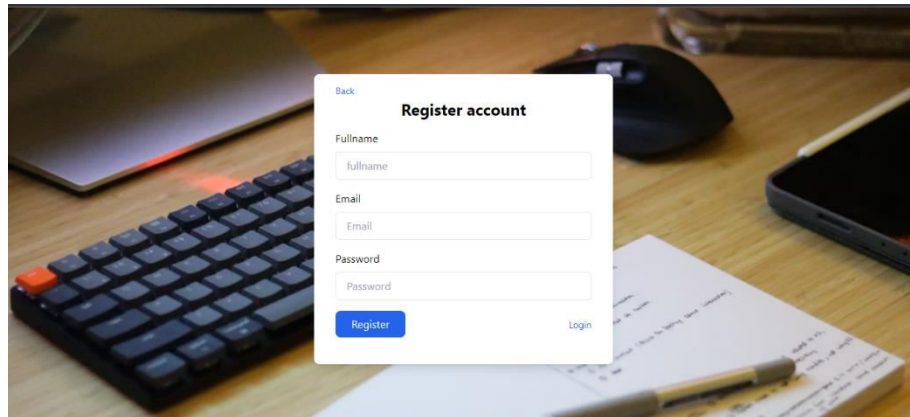
Halaman Home adalah halaman yang pertama kali muncul ketika website diakses oleh pengguna. Halaman home dapat diakses oleh pengguna tanpa melakukan autentikasi. Halaman home ditunjukkan seperti gambar 4.18.



Gambar 4. 18 Implementasi Desain Halaman Home

b. Halaman Fitur Register

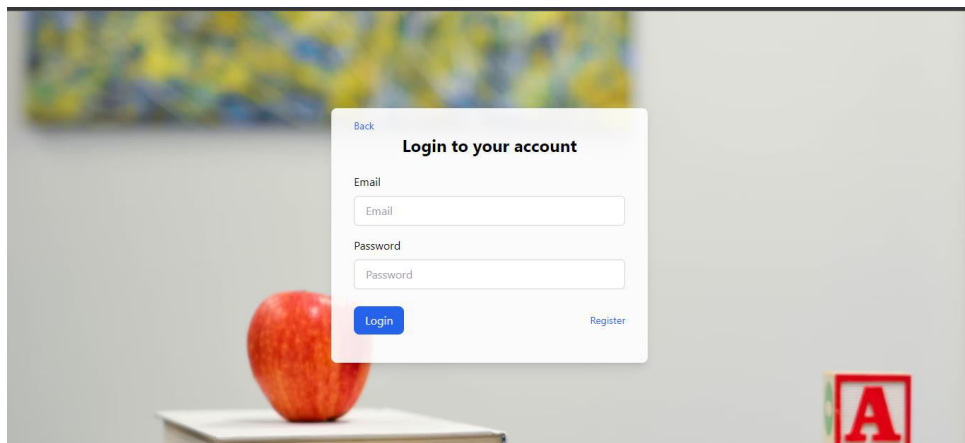
Halaman register memuat fitur pembuatan akun bagi pengguna. Halaman register akan tampil, ketika user menekan tombol register pada navbar. Terdapat formulir register yang digunakan pengguna untuk mengisi data seperti fullname, email, dan password. Kemudian pengguna ketika berhasil melakukan register maka akan mendapatkan akun sistem. Halaman register ditunjukkan seperti gambar 4.19.



Gambar 4. 19 Implementasi Desain Halaman Fitur Register

c. Halaman Fitur Login

Halaman login adalah halaman yang dapat diakses ketika menekan tombol login pada navbar di halaman home. Pada halaman login memuat fitur untuk masuk kedalam sistem. Pada halaman login terdapat form login yang digunakan untuk mengisi data akun seperti email dan password. Setelah berhasil pada proses login, maka sistem akan mengalihkan ke halaman home. Halaman login ditunjukkan seperti gambar 4.20.

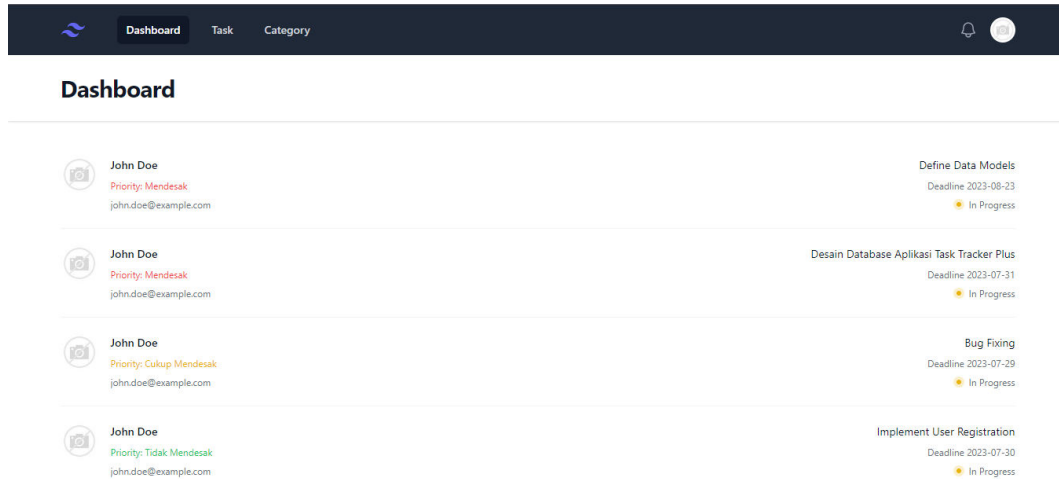


Gambar 4. 20 Implementasi Desain Halaman Fitur login

d. Halaman Dashboard

Halaman dashboard adalah halaman yang akan muncul ketika pengguna berhasil melakukan login. halaman ini memuat daftar data task yang dimiliki pengguna. Pada halaman ini juga menampilkan navbar yang berisi tombol dashboard, task, category dan tombol dropdown yang

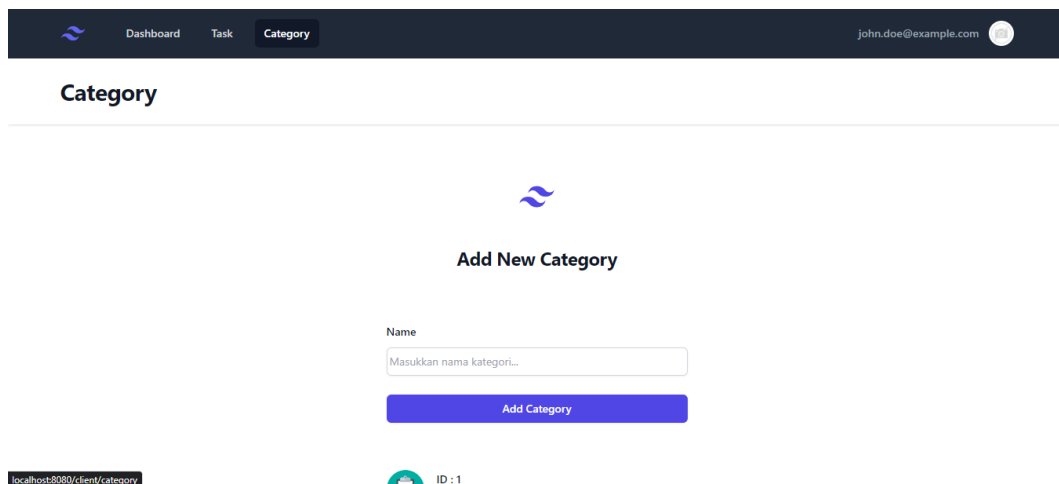
digunakan untuk logout. Halaman dashboard ditunjukkan seperti gambar 4.21.



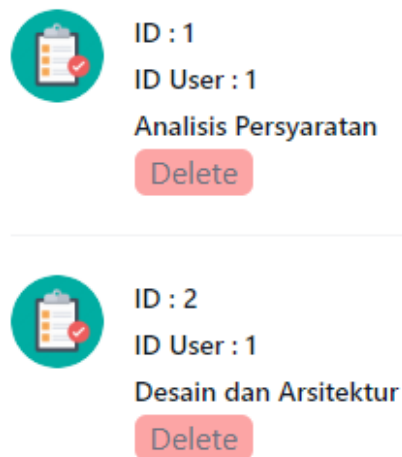
Gambar 4. 21 Implementasi Desain Halaman Dashboard

e. Halaman Kategori

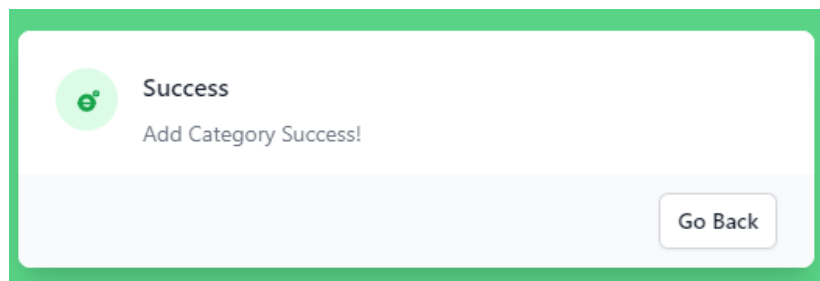
Pada halaman kategori memuat formulir tambah kategori, formulir ini berisi inputan nama. Selanjutnya dibawah formulir terdapat list data kategori, menampilkan nama kategori dan button delete yang digunakan untuk menghapus data kategori. Halaman kategori ditunjukkan seperti gambar 4.22.



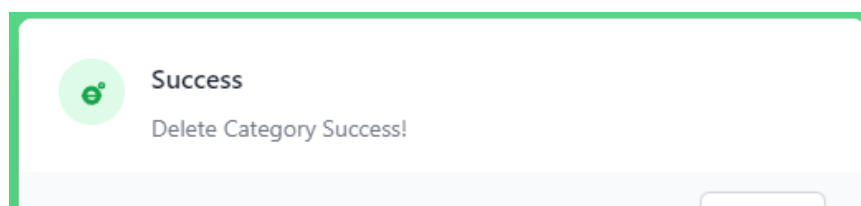
Gambar 4. 22 Implementasi Desain Halaman Tambah Kategori

Gambar 4. 23 Implementasi Desain Halaman *List Kategori***f. Modal Sukses Category**

Modal ini memuat informasi, bahwa pengguna telah berhasil menambah kategori baru. Modal tambah kategori ditunjukkan seperti gambar 4.24.

Gambar 4. 24 Implementasi Desain Modal *Success Add Category***g. Modal Delete Category**

Modal ini memuat informasi, bahwa pengguna telah berhasil menghapus kategori yang dipilih. Modal delete category ditunjukkan seperti gambar 4.25.



Gambar 4. 25 Implementasi Desain Modal Delete Category

h. Halaman Task

Pada halaman task menampilkan formulir tambah task, formulir ini berisi inputan title, inputan deadline, inputan priority, inputan status, inputan user_id, inputan category_id. Selanjutnya dibawah formulir terdapat list data task pengguna, menampilkan judul task, priority, category id, user_id, deadline, status task dan button delete yang digunakan untuk menghapus data task.

Add New Task

Title

Start

End/Deadline

Prioritas

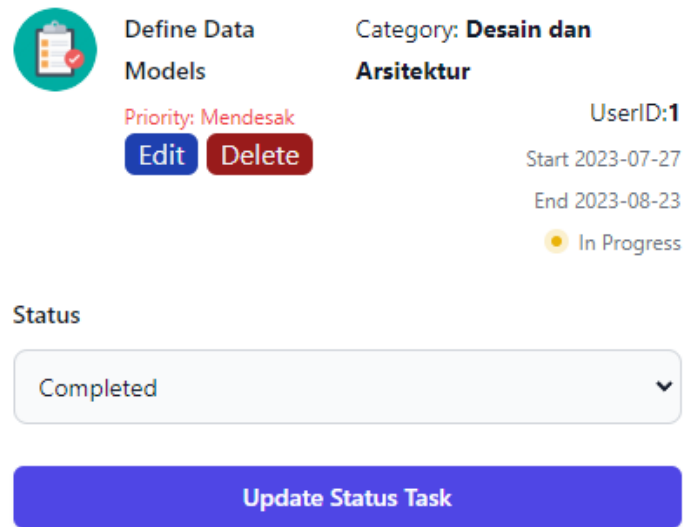
Status

Category

Gambar 4. 26 Implementasi Desain Halaman Tambah Task

 Define Data Models Priority: Mendesak Edit Delete	Category: Desain dan Arsitektur UserID: 1 Start 2023-07-27 End 2023-08-23 In Progress
 Desain Database Aplikasi Task Tracker Plus Priority: Mendesak Edit Delete	Category: Desain dan Arsitektur UserID: 1 Start 2023-07-28 End 2023-07-31 In Progress
 Bug Fixing Priority: Cukup Men... Edit Delete	Category: Analisis Persyaratan UserID: 1 Start 2023-07-27 End 2023-07-29 In Progress

Gambar 4. 27 Implementasi Desain Halaman Daftar Task



Define Data Models

Category: **Desain dan Arsitektur**

Priority: **Mendesak**

UserID: **1**

Start 2023-07-27

End 2023-08-23

In Progress

Status

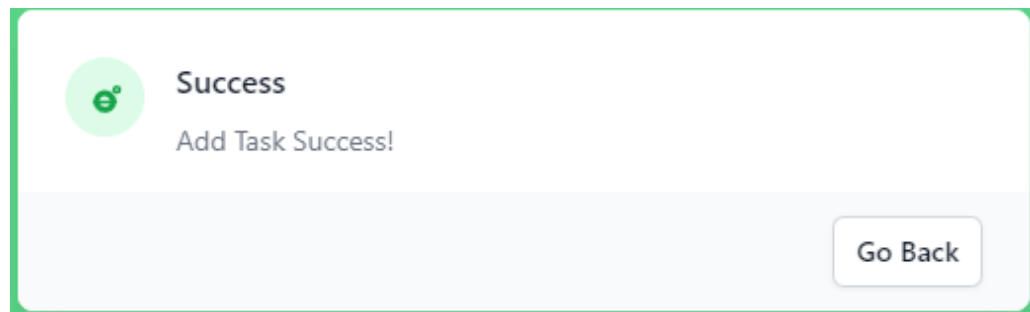
Completed

Update Status Task

Gambar 4. 28 Implementasi Desain Halaman Update Status Task

i. Modal Success Add Task

Modal ini memuat informasi, bahwa pengguna telah berhasil menambah task baru. Modal tambah task ditunjukkan seperti gambar 4.29.



Success

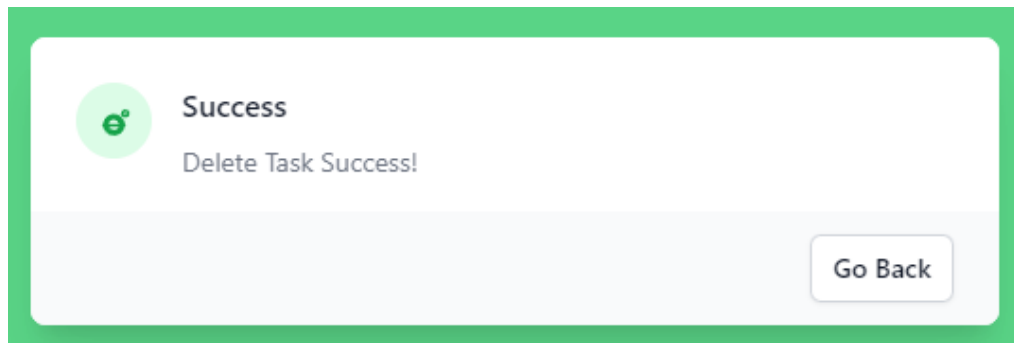
Add Task Success!

Go Back

Gambar 4. 29 Implementasi Desain Modal Success Add Task

j. Modal Delete Task

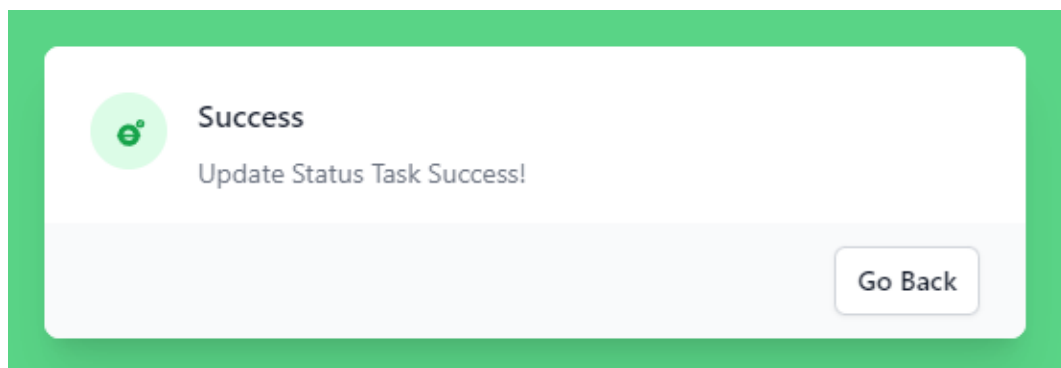
Modal ini memuat informasi, bahwa pengguna telah berhasil menghapus task yang dipilih. Modal delete task ditunjukkan seperti gambar 4.30.



Gambar 4. 30 Implementasi Desain Modal Delete Task

k. **Modal Update Status Task**

Modal ini memuat informasi, bahwa pengguna telah berhasil mengupdate status task yang dipilih. Modal update status task ditunjukkan seperti gambar 4.31.



Gambar 4. 31 Implementasi Desain Modal Update Status Task

4.7 Hasil Pengujian Sistem

Pada tahap ini, dilakukan pengujian sistem dengan melakukan pengujian pengujian pada sistem dengan metode *Blackbox Testing*. Metode *Blackbox Testing* adalah metode yang digunakan untuk menguji sebuah sistem tanpa harus memperhatikan detail dari perangkat lunak. Pengujian ini hanya memeriksa nilai keluaran berdasarkan nilai masukkan masing-masing (Latief 2015). Pada metode *Blackbox testing* penulis menggunakan teknik *Equivalence Partitioning (EP)*, teknik ini menguji dengan masukan serta membagi kedalam kelompok-kelompok berdasarkan fungsinya.

Untuk uji coba system pada aplikasi web "Task Tracker" menggunakan grader-cli, yaitu Command Line Interface (CLI) yang di sediakan oleh PT. Ruang Raya Indonesia untuk menguji coba dan menilai tugas.

1. Lingkungan Pelaksanaan Pengujian

Pada tahap ini, dilakukan pengujian dengan menggunakan perangkat seperti pada tabel 4.15.

Tabel 4. 15 Perangkat Uji

Perangkat Uji	Sisi Pengguna	Sisi Server
Aplikasi	Web	Localhost
Jenis Perangkat	Laptop	Laptop
Prosesor	AMD A9-9425 (2 cores at 3.1 Ghz / up to 3.7 Ghz / 1MB Cache)	AMD A9-9425 (2 cores at 3.1 Ghz / up to 3.7 Ghz / 1MB Cache)
Memori	240GB	240GB
Sistem Operasi	Windows	Windows
Jenis Sistem Operasi	Windows 10 (64x)	Windows (64x)

2. Pengujian Register

Pada tahap ini akan dijelaskan pengujian pada fitur register pada sisi pengembang untuk menguji fungsi register, pengujian fungsi register dilakukan sesuai dengan rancangan kasus uji dan juga hasil pengujian dapat dilihat dari hasil yang diperoleh seperti pada tabel 4.16.

Tabel 4. 16 Pengujian Fitur register

Test ID	TUC001			
Tujuan Test	Mengecek apakah sistem dapat melakukan register dengan tepat			
Kondisi Awal	Pengguna berada pada halaman register			
Data Input	Prosedur pengujian	Hasil yang diharapkan	Hasil yang diperoleh	Kesimpulan

Menginputkan data register pada form register, fullname: "tester" email: "test@mail.com" dan password: "testing123".	Pengguna memasukkan data register pada form masukkan register.	Pengguna berhasil melakukan register dan masuk pada halaman login.	Pengguna masuk pada halaman login.	Pengujian fitur register sesuai.
--	--	--	------------------------------------	----------------------------------

3. Pengujian Login

Pada tahap ini akan dijelaskan pengujian pada fitur login pada sisi pengembang untuk menguji fungsi login, pengujian fungsi login dilakukan sesuai dengan rancangan kasus uji dan juga hasil pengujian dapat dilihat dari hasil yang diperoleh seperti pada tabel 4.17.

Tabel 4. 17 Pengujian Fitur Login

Test ID	TUC002			
Tujuan Test	Mengecek apakah sistem dapat melakukan login dengan tepat			
Kondisi Awal	Pengguna berada pada halaman login			
Data Input	Prosedur pengujian	Hasil yang diharapkan	Hasil yang diperoleh	Kesimpulan
Menginputkan data login pada form login, email: "test@mail.com" dan password:	Pengguna memasukkan data login pada form masukkan login.	Pengguna berhasil melakukan login dan masuk pada halaman dashboard	Pengguna masuk pada halaman dashboard aplikasi task tracker plus.	Pengujian fitur login sesuai.

"testing123"		aplikasi task tracker plus.		
--------------	--	-----------------------------	--	--

4. Pengujian Logout

Pada tahap ini akan dijelaskan pengujian pada fitur logout pada sisi pengembang untuk menguji fungsi logout, pengujian fungsi logout dilakukan sesuai dengan rancangan kasus uji dan juga hasil pengujian dapat dilihat dari hasil yang diperoleh seperti pada tabel 4.18.

Tabel 4. 18 Pengujian Fitur Logout

Test ID	TUC003			
Tujuan Test	Mengecek apakah sistem dapat melakukan logout dengan tepat			
Kondisi Awal	Pengguna berada pada halaman dashboard			
Data Input	Prosedur pengujian	Hasil yang diharapkan	Hasil yang diperoleh	Kesimpulan
Data Logout "id"	Pengguna mengklik button logout pada navbar	Pengguna berhasil melakukan logout dan masuk pada halaman login.	Pengguna masuk pada halaman login.	Pengujian fitur logout sesuai.

5. Pengujian Tambah Kategori

Pada tahap ini akan dijelaskan pengujian pada fitur tambah kategori pada sisi pengembang untuk menguji fungsi tambah kategori, pengujian fungsi tambah kategori dilakukan sesuai dengan rancangan kasus uji dan juga hasil pengujian dapat dilihat dari hasil yang diperoleh seperti pada tabel 4.19.

Tabel 4. 19 Pengujian Fitur Tambah Kategori

Test ID	TUC004			
Tujuan Test	Mengecek apakah sistem dapat melakukan tambah kategori dengan tepat			
Kondisi Awal	Pengguna berada pada halaman kategori			
Data Input	Prosedur pengujian	Hasil yang diharapkan	Hasil yang diperoleh	Kesimpulan
Menginputkan data tambah kategori pada form tambah kategori, name: "Category 1"	Pengguna memasukkan data tambah kategori pada form masukkan kategori.	Data kategori masuk pada basis data dan tampil pada list kategori di halaman kategori	Data kategori tampil pada list kategori di halaman kategori dan tersimpan pada data base	Pengujian fitur tambah kategori sesuai.

6. Pengujian Hapus Kategori

Pada tahap ini akan dijelaskan pengujian pada fitur hapus kategori pada sisi pengembang untuk menguji fungsi hapus kategori, pengujian fungsi hapus kategori dilakukan sesuai dengan rancangan kasus uji dan juga hasil pengujian dapat dilihat dari hasil yang diperoleh seperti pada tabel 4.20.

Tabel 4. 20 Pengujian Fitur Hapus Kategori

Test ID	TUC005			
Tujuan Test	Mengecek apakah sistem dapat melakukan hapus kategori dengan tepat			
Kondisi Awal	Pengguna berada pada halaman kategori			
Data Input	Prosedur pengujian	Hasil yang diharapkan	Hasil yang diperoleh	Kesimpulan
Data Kategori “id”	Pengguna menghapus data kategori pada button delete di kategori list.	Dapat menghapus data kategori pada database	Data berhasil dihapus dan tidak tampil pada tampilan list kategori di halaman kategori dan dihapus pada basis data	Pengujian fitur hapus data kategori sesuai.

7. Pengujian Tambah Task

Pada tahap ini akan dijelaskan pengujian pada fitur tambah task pada sisi pengembang untuk menguji fungsi tambah task, pengujian fungsi tambah task dilakukan sesuai dengan rancangan kasus uji dan juga hasil pengujian dapat dilihat dari hasil yang diperoleh seperti pada tabel 4.21.

Tabel 4. 21 Pengujian Fitur Tambah Task

Test ID	TUC006			
Tujuan Test	Mengecek apakah sistem dapat melakukan tambah task dengan tepat			
Kondisi Awal	Pengguna berada pada halaman task			
Data Input	Prosedur pengujian	Hasil yang diharapkan	Hasil yang diperoleh	Kesimpulan
Menginputkan data tambah task pada form tambah tambah kategori, title: “Task 1”, Deadline:” 2023-06-02”, Priority: “1”, Status: “In Progress”, Category_Id: “1” dan User Id: “3”	Pengguna memasukkan data tambah task pada form masukkan task.	Data task masuk pada basis data dan tampil pada list task di halaman task	Data task tampil pada list task di halaman task dan tersimpan pada data base	Pengujian fitur tambah task sesuai.

8. Pengujian Hapus Task

Pada tahap ini akan dijelaskan pengujian pada fitur hapus task pada sisi pengembang untuk menguji fungsi hapus task, pengujian fungsi hapus task dilakukan sesuai dengan rancangan kasus uji dan juga hasil pengujian dapat dilihat dari hasil yang diperoleh seperti pada tabel 4.22.

Tabel 4. 22 Pengujian Fitur Hapus Task

Test ID	TUC007
----------------	--------

Tujuan Test		Mengecek apakah sistem dapat melakukan hapus task dengan tepat		
Kondisi Awal		Pengguna berada pada halaman task		
Data Input	Prosedur pengujian	Hasil yang diharapkan	Hasil yang diperoleh	Kesimpulan
Data task "id"	Pengguna menghapus data task pada button delete di task list.	Dapat menghapus data task pada database	Data berhasil dihapus dan tidak tampil pada tampilan list task di halaman task dan dihapus pada basis data	Pengujian fitur hapus data task berhasil sesuai.

Setelah dilakukan pengujian data dengan metode *blackbox testing* menggunakan teknik *Equivalence Partitioning (EP)* pada uji coba diatas, maka dapat diambil kesimpulan bahwa Aplikasi Task Tracker Plus setelah dilakukan pengujian dapat berjalan dengan baik.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan uraian pembahasan, analisis, dan pengujian yang telah dilakukan oleh penulis, maka dapat diambil beberapa kesimpulan dari Aplikasi Task Tracker Plus sebagai berikut:

1. Penelitian ini berhasil merancang dan mengimplementasikan struktur REST API yang efektif digunakan untuk mendukung layanan Aplikasi Task Tracker Plus.
2. Penelitian yang dikembangkan juga berhasil menerapkan pola Model-View-Controller dalam pengembangan Aplikasi Task Tracker Plus. Dengan menerapkan pola MVC mempunyai keunggulan untuk mengembangkan Aplikasi Task Tracker Plus menjadi lebih mudah.
3. Berdasarkan hasil pengujian sistem menggunakan *Blackbox Testing* dengan teknik *Equivalence Partitioning (EP)*, Aplikasi Task Tracker Plus dapat berjalan dengan baik.

5.2 Saran

Berdasarkan permasalahan dan pengamatan yang diamati setelah melakukan penelitian dan pembuatan Aplikasi Task Tracker Plus , terdapat beberapa saran yang dapat diberikan agar Aplikasi Task Tracker Plus dapat berkembang lebih baik, antara lain:

1. Aplikasi Task Tracker Plus bisa dikembangkan dengan perbaikan antar muka yang semakin baik dan mudah digunakan.
2. Penambahan fitur yang dapat meningkatkan fungsionalitas Aplikasi Task Tracker Plus.
3. Aplikasi Task Tracker Plus dapat dikembangkan berbasis perangkat mobile baik Android maupun IOS.

DAFTAR PUSTAKA

- Akbar, M. (2018). PENGEMBANGAN RESTFUL API UNTUK APPLICATION SPECIFIC HIGH LEVEL LOCATION SERVICE. Retrieved from <https://dspace.uui.ac.id/handle/123456789/9836>
- Ardian, D. P., & Maryanah, S. (2020). STUDI PERBANDINGAN PERFORMANSI ANTARA MYSQL DAN POSTGRESQL. *JURNAL KHATULISTIWA INFORMATIKA*. Retrieved from <https://ejournal.bsi.ac.id/ejurnal/index.php/khatulistiwa/article/download/8851/4533>
- FIRDAUS, A., WIDODO, S., SUTRISMAN, A., GADING, S., & MARDIANA, R. (2019). RANCANG BANGUN SISTEM INFORMASI PERPUSTAKAAN MENGGUNAKAN WEB SERVICE PADA JURUSAN TEKNIK KOMPUTER POLSRI. *Jurnal Informanika*. Retrieved from <https://poltekanika.ac.id/journal/index.php/inf/article/view/99/88>
- Hradil, J., & Sklenak, V. (2017). Practical Implementation of 10 Rules for Writing REST APIs. Retrieved from <https://www.proquest.com/openview/32a20d1d4fab9d366ff027dff785621f/1.pdf?pq-origsite=gscholar&cbl=2032453>
- Ismail, M. L. (2019). Perancangan Sistem Informasi Pengolahan Nilai Siswa Menggunakan Java Spring RESTful Web Service. Retrieved from <http://repository.uksw.edu/handle/123456789/19091>
- Jaidan, J. (2012). Studi Pendahuluan Untuk Pengembangan Model.
- Klimm, C. M. (2021). *Design Systems for Micro Frontends-An Investigation into the Development of Framework-Agnostic Design Systems using Svelte and Tailwind CSS*.
- Kristanto, A. A. (2020). PEMBANGUNAN SISTEM SANDBOX PADA PENGUJIAN TERHADAP PIHAK KETIGA BERBASIS REST API MENGGUNAKAN GOLANG DAN NSQ. Retrieved from <http://e-journal.uajy.ac.id/23329/>

- Kristanto, A. A., & Harjoseputro, Y. (2020). Implementasi Golang dan New Simpel Queue pada Sistem Sandbox Pihak Ketiga Berbasis REST API. *RESTI (Rekayasa Sist. Dan Teknol. Informasi)*(4), 745-750. doi:10.29207/resti.v4i4.2218
- Kustian, N. G. (2021). PENGGUNAAN METODE MIND MAPPING DALAM MENINGKATKAN HASIL BELAJAR SISWA. *Jurnal Inovasi Riset Akademik*, 30-37.
- Latif, A. (2015). IMPLEMENTASI KRIPTOGRAFI MENGGUNAKAN METODE ADVANCED ENCRYPTION STANDAR (AES) UNTUK PENGAMANAN DATA TEKS. *Mustek Jurnal Teknik*. Retrieved from <https://ejournal.unmus.ac.id/index.php/mustek/article/view/471>
- Perdana, M. A. (2018). Pengembangan REST API Layanan Penyimpanan menggunakan Metode Rapid Application Development (Studi kasus PT. XYZ). *InfoTekJar (Jurnal Nasional Informatika dan Teknologi Jaringan)*. doi:<https://doi.org/10.30743/infotekjar.v3i1.563>
- Prayogo, N. A. (2012). *Dasar Pemrograman Golang*. Retrieved Juli 12, 2023, from dasarpemrogramangolang.novalagung.com/
- Sari, A. S., & Hidayat, R. (2022). Designing website vaccine booking system using golang programming language and framework react JS. *Journal of Information System, Informatics and Computing Issue Period*, 22–39. doi:<https://doi.org/10.52362/jisicom.v6i1.760>
- Subaeki, B. (2022). PERANCANGAN ARSITEKTUR SISTEM INFORMASI MENGGUNAKAN METODE ENTERPRISE ARCHITECTURE PLANNING (Studi Kasus : Universitas Purwakarta - Purwakarta). *e-Journal Bina Sarana Informatika*, 1-13.
- Tawfik, D. (2023, Juni 18). *Why tracking tasks and using a task tracker is essential to productivity*. Diambil kembali dari [monday.com](https://monday.com/blog/project-management/task-tracker/): <https://monday.com/blog/project-management/task-tracker/>
- Wahid, A. A. (2020). Analisis Metode Waterfall Untuk Pengembangan Sistem. *Jurnal Ilmu-ilmu Informatika dan Manajemen STMIK*. Retrieved from

https://www.researchgate.net/profile/Aceng-Wahid/publication/346397070_Analisis_Metode_Waterfall_Untuk_Pengembangan_Sistem_Informasi/links/5fbfa91092851c933f5d76b6/Analisis-Metode-Waterfall-Untuk-Pengembangan-Sistem-Informasi.pdf

LAMPIRAN

Lampiran 1 *Log Activity*

Minggu Ke-	Kegiatan	Hasil
1	Saya mengikuti serangkaian kegiatan <i>Onboarding</i>	Saya memperoleh berbagai macam wawasan dan informasi bermanfaat untuk kegiatan kedepannya.
2	Saya mempelajari materi <i>General Competence</i> .	Saya memahami modul dalam sub-materi mengenai <i>Thinking Skills</i> , <i>Project Management Tech</i> , dan <i>Communication & Presentation</i> . Sehingga saya dapat mengerjakan <i>Post Test</i> dengan baik.
3	Saya mempelajari materi <i>Software Engineering week 1</i>	Saya memahami modul <i>Introduction to Software Engineering</i> , <i>Terminal</i> , <i>Version Control</i> , dan <i>IDE/Code Editor</i> . Sehingga saya dapat mengerjakan <i>Assignment 1</i> dengan baik.
4	Saya mempelajari materi <i>Software Engineering week 2</i>	Saya memahami modul <i>Network</i> , dan <i>Algorithm</i> . Sehingga saya dapat mengerjakan <i>Assignment 2</i> dan <i>Post Test</i> dengan baik.
5	Saya mengerjakan <i>Final Course Project – Software Engineering</i>	Saya menyelesaikan <i>Final Course Project – Software Engineering</i> dengan baik.
6	Saya mempelajari materi <i>Basic FE With HTML and CSS week 1</i>	Saya memahami beberapa hal berikut dalam modul HTML: • Membuat sebuah halaman HTML sederhana dalam bentuk <i>Online CV</i>. • Menggunakan <i>IDE/Code</i>

		<i>editor</i> dan <i>Web Browser</i> dalam pembuatan halaman HTML. • Menerapkan <i>HTML Element / Tag</i> untuk membungkus konten dengan tepat dan membuat <i>attribute</i> dengan benar di halaman HTML. • Membuat sebuah form di halaman HTML. • Menerapkan <i>semantic HTML</i> di halaman HTML. Sehingga saya dapat menyelesaikan <i>Assignment</i> dengan baik.
7	Saya mempelajari materi <i>Basic FE With HTML and CSS week 2</i>, serta mengerjakan <i>Final Course Project – Basic FE With HTML and CSS</i>.	Saya memahami beberapa hal berikut dalam modul CSS: • Menggunakan CSS untuk mempercantik halaman HTML melalui sebuah <i>file CSS</i> dan/atau penerapan <i>syntax CSS</i> di dalam dokumen HTML. • Menggunakan <i>IDE/Code editor</i> dan <i>Web Browser</i> dalam penerapan CSS pada halaman HTML. • Menerapkan CSS <i>Rule</i> ke HTML <i>Element</i> dengan menggunakan CSS <i>Selector</i>. • Menerapkan <i>layouting</i> menggunakan <i>Grid</i> dan <i>Flexbox</i> pada halaman HTML. • Menerapkan <i>Responsive Web Design</i> pada halaman HTML. • Menerapkan CSS <i>Framework Tailwind CSS</i> pada halaman HTML. Sehingga saya dapat menyelesaikan <i>Assignment, Post Test</i>, serta <i>Final Course Project – Basic FE With HTML and CSS</i> dengan baik.

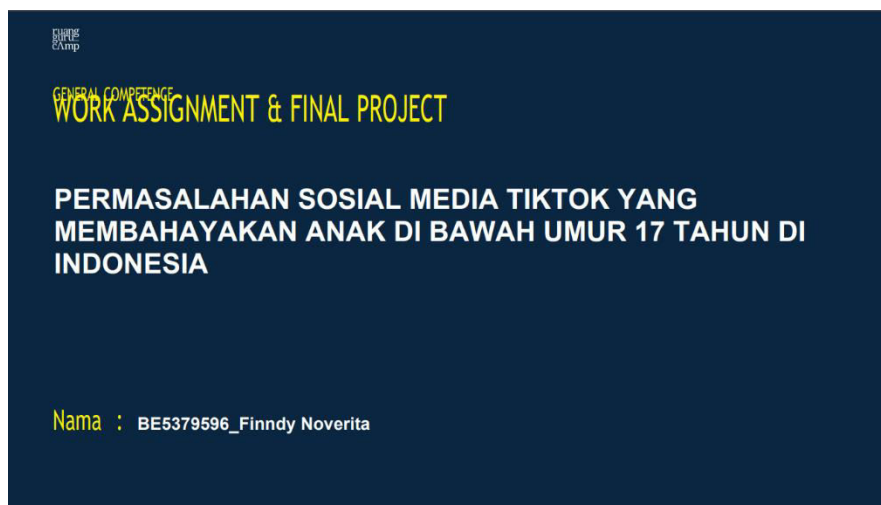
8	Saya mempelajari materi Dasar Pemrograman <i>Backend</i> week 1 tentang <i>Basic Golang</i> .	Saya memahami tentang <i>Basic Command and Console, Working with Data, Penggunaan Grdaer, Condition, Looping, dan Function</i> . Sehingga saya dapat menyelesaikan <i>Exercise</i> dan <i>Assignment</i> 1 dengan baik.
9	Saya mempelajari materi Dasar Pemrograman <i>Backend</i> week 2	Saya memahami tentang <i>Pointer, Array, Slic, Map, dan Slice Multidimensional</i> . Sehingga saya dapat menyelesaikan <i>Exercise</i> dan <i>Assignment</i> 2 dengan baik.
10	Saya mempelajari materi Dasar Pemrograman <i>Backend</i> week 3, serta mengerjakan <i>Final Course Project – Dasar Pemrograman Backend</i> .	Saya memahami tentang <i>Struct, Interface, Import, Package, dan Handling Error</i> . Sehingga saya dapat menyelesaikan <i>Exercise, Post Test, dan Final Course Project – Dasar Pemrograman Backend</i> dengan baik.
11	Saya tidak mempelajari materi karena cuti bersama Hari Raya Idul Fitri	-

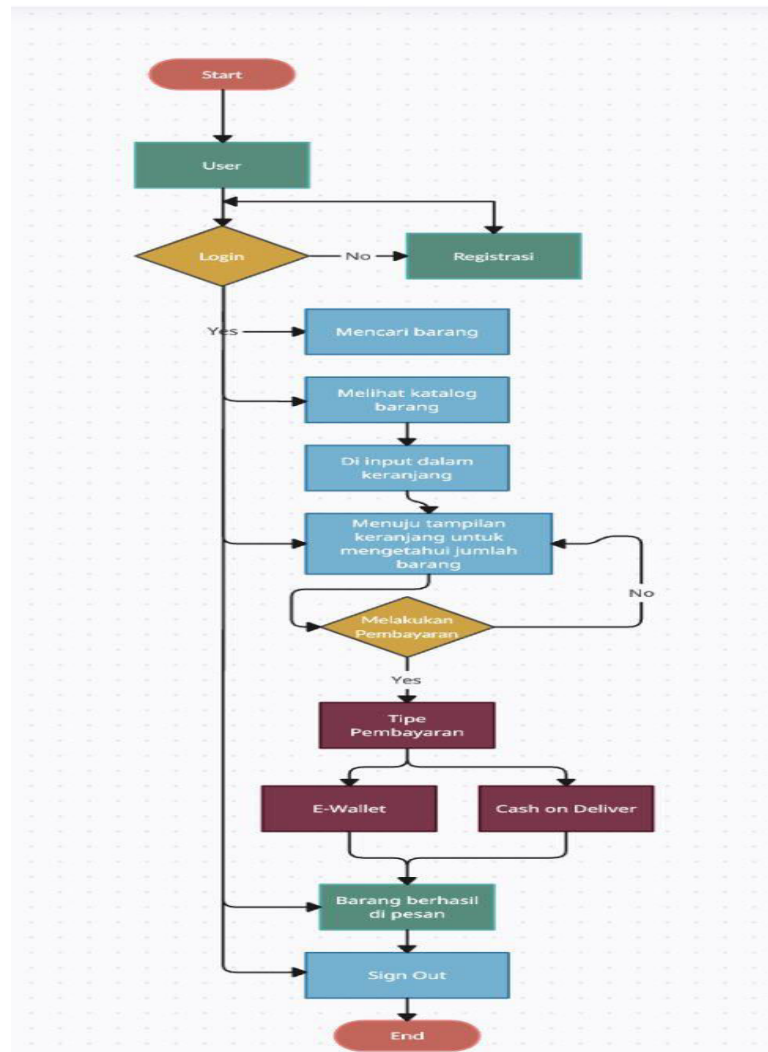
12	Saya mempelajari materi Pemrograman <i>Backend</i> Lanjutan <i>week 1</i> tentang <i>Golang i/O</i>.	Saya memahami tentang File, JSON, HTTP Client, dan Server HTTP. Serta beberapa hal berikut: • Membuat sebuah web server (aplikasi berbasis web) menggunakan protokol HTTP dalam bahasa Golang. • Menggunakan Golang I/O untuk mengambil, mengirim maupun menyimpan data untuk aplikasi berbasis web yang dibuat dengan bahasa pemrograman Golang. Sehingga saya dapat menyelesaikan <i>Exercise</i>, dan <i>Assignment 1</i> dengan baik.
13	Saya mempelajari materi Pemrograman <i>Backend</i> Lanjutan <i>week 2</i> tentang <i>Concurrency, Data Structure & Design Pattern</i>.	Setelah mempelajari materi tentang <i>concurrency</i>, saya dapat: • Menerapkan konsep <i>concurrency</i> pada aplikasi Golang yang dibuat. • Membuat <i>Goroutine</i> untuk kode yang bersifat <i>concurrent</i> pada aplikasi Golang yang dibuat. • Menerapkan <i>Channel</i> dan <i>Buffer Channel</i> untuk bertukar data antar <i>Goroutine</i> pada bahasa Golang. Setelah saya mempelajari materi tentang <i>Data Structure & Design Pattern</i>, saya dapat: • Menerapkan <i>complexity analysis</i> untuk menganalisa aplikasi yang dibuat. • Menggunakan struktur data dengan tepat sesuai dengan kebutuhan. • Menerapkan <i>Factory Pattern</i> dengan tepat pada aplikasi

		yang dibuat. Sehingga saya dapat menyelesaikan <i>Exercise</i> , dan <i>Post Test</i> dengan baik.
14	Saya mengerjakan <i>Final Course Project - Pemrograman Backend Lanjutan</i> .	Saya menyelesaikan <i>Final Course Project - Pemrograman Backend Lanjutan</i> dengan baik.
15	Saya mempelajari materi <i>Database Management & Technology Session 1</i>	Saya memahami tentang <i>Database System, Basic Principle, Data Model, PostgreSQL, Data Definition Language, Data Manipulation Language, Data Query Language</i> , dan <i>Joining Database Table</i> . Sehingga saya dapat menyelesaikan <i>Exercise</i> dengan baik.
16	Saya mempelajari materi <i>Database Management & Technology Session 2</i> , serta mengerjakan <i>Final Course Project - Database Management & Technology</i> .	Saya memahami tentang <i>ORM (GROM) & NoSQL</i> . Sehingga saya dapat menyelesaikan <i>Exercise, Assignment, Post Test</i> , dan <i>Final Course Project - Database Management & Technology</i> dengan baik.
17	Saya mempelajari materi <i>Web Application</i> materi 1	Saya memahami tentang <i>REST API & Clean Architecture</i> . Sehingga saya dapat menyelesaikan <i>Exercise</i> dan <i>Assignment 1</i> dengan baik.
18	Saya mempelajari materi <i>Web Application</i> materi 2, serta mengerjakan <i>Final Course Project - Web Application</i> .	Saya memahami tentang <i>Postman & PaaS</i> . Sehingga saya dapat menyelesaikan <i>Exercise, Assignment 2</i> , serta <i>Final Course</i>

		<i>Project - Web Application</i> dengan baik.
--	--	---

Lampiran 2 Dokumen Teknik MSIB PT. Ruang Raya Indonesia





```

Function shoppingcart()
  Start
    keranjang_belanja={}
    A=True
    print "Opsi Keranjang Belanja"
    print "1.Tambah item"
    print "2.Hapus item"
    print "3.Cek Keranjang"
    print "4.Pembayaran"
    print "5.Main Menu"
  While A == True
    Get option
    B=True
    While B == True

      if option == 1 then
        item=customize()
        jmlh=Jumlah()
        keranjang_belanja[item]=jmlh
        A=True
        B=False
      else if option == 2 then
        Get item
        Delete(keranjang_belanja[item])
        A=True
        B=False
      else if option == 3 then
        print keranjang_belanja[item]=jmlh
        A=True
        B=False
        else if option == 4 then
          pembayaran()
          confirm=confirmB()
          A=True
          B=False
        else if option == 5 then

```

```

category.go - fcp-web-application-v1 - Visual Studio Code
handler > web > category.go
22 embed embed.FS
23 }
24
25 func NewCategoryWeb(categoryClient client.CategoryClient, sessionService service.SessionService, embed embed.FS) *categoryWeb {
26     return &categoryWeb{categoryClient, sessionService, embed}
27 }
28
29 func (c *categoryWeb) category(ctx *gin.Context) {
30     var email string
31     if temp, ok := ctx.Get("email"); ok {
32         if contextData, ok := temp.(string); ok {
33             email = contextData
34         }
35     }
36
37     session, err := c.sessionService.GetSessionByEmail(email)
38     if err != nil {
39         ctx.JSON(http.StatusInternalServerError, model.ErrorResponse{Error: err.Error()})
40         return
41     }
42 }
43
44 // History restored
45
46 [24 Jun 23 10:38:40] GET /client/category "
47 [24 Jun 23 10:38:40] GET /client/jquery-3.6.0.min.js "
48 [24 Jun 23 10:39:40] POST /client/category/add/process "
49 [24 Jun 23 10:39:40] GET /api/v1/task/list "
50 [24 Jun 23 10:39:40] GET /client/task "
51 [24 Jun 23 10:39:40] GET /client/jquery-3.6.0.min.js "
52 [24 Jun 23 10:39:40] GET /api/v1/category/list "
53 [24 Jun 23 10:39:40] GET /client/category "
54 [24 Jun 23 10:39:40] GET /client/jquery-3.6.0.min.js "
55 [24 Jun 23 10:39:40] POST /client/category/add/process "
56 [24 Jun 23 14:59:40] GET /client/login "
57 [24 Jun 23 15:00:40] GET / "
58 [24 Jun 23 15:00:40] GET /jquery-3.6.0.min.js "

```

```

category.go - fcp-web-application-v1 - Visual Studio Code
handler > web > category.go
22 embed embed.FS
23 }
24
25 func NewCategoryWeb(categoryClient client.CategoryClient, sessionService service.SessionService, embed embed.FS) *categoryWeb {
26     return &categoryWeb{categoryClient, sessionService, embed}
27 }
28
29 func (c *categoryWeb) category(ctx *gin.Context) {
30     var email string
31     if temp, ok := ctx.Get("email"); ok {
32         if contextData, ok := temp.(string); ok {
33             email = contextData
34         }
35     }
36
37     session, err := c.sessionService.GetSessionByEmail(email)
38     if err != nil {
39         ctx.JSON(http.StatusInternalServerError, model.ErrorResponse{Error: err.Error()})
40         return
41     }
42 }
43
44 // History restored
45
46 [24 Jun 23 10:38:40] GET /client/category "
47 [24 Jun 23 10:38:40] GET /client/jquery-3.6.0.min.js "
48 [24 Jun 23 10:39:40] POST /client/category/add/process "
49 [24 Jun 23 10:39:40] GET /api/v1/task/list "
50 [24 Jun 23 10:39:40] GET /client/task "
51 [24 Jun 23 10:39:40] GET /client/jquery-3.6.0.min.js "
52 [24 Jun 23 10:39:40] GET /api/v1/category/list "
53 [24 Jun 23 10:39:40] GET /client/category "
54 [24 Jun 23 10:39:40] GET /client/jquery-3.6.0.min.js "
55 [24 Jun 23 10:39:40] POST /client/category/add/process "
56 [24 Jun 23 14:59:40] GET /client/login "
57 [24 Jun 23 15:00:40] GET / "
58 [24 Jun 23 15:00:40] GET /jquery-3.6.0.min.js "

```

```

session.go - db-assignment-1-v1 - Visual Studio Code
repository > session.go
37 if err != nil {
38     return err
39 }
40
41 return nil
42 }
43
44 func (u *sessionsRepoImpl) UpdateSessions(session model.Session) error {
45     err := u.db.Exec("UPDATE sessions SET token = $1, expiry = $2 WHERE username = $3", session.Token, session.Expiry, session.Username)
46     if err != nil {
47         return err
48     }
49     return nil
50 }
51
52 func (u *sessionsRepoImpl) SessionWaitToken(name string) error {
53     row := u.db.QueryRow("SELECT id, token, username, expiry FROM sessions WHERE username = $1", name)
54
55     var session model.Session
56     err := row.Scan(&session.ID, &session.Token, &session.Username, &session.Expiry)
57     if err != nil {
58         if err == sql.ErrNoRows {
59             return err
60         }
61         return err
62     }
63
64     return err
65 }
66
67 func (u *sessionsRepoImpl) SessionWaitToken(token string) (model.Session, error) {
68     row := u.db.QueryRow("SELECT id, token, username, expiry FROM sessions WHERE token = $1", token)
69
70     var session model.Session
71     err := row.Scan(&session.ID, &session.Token, &session.Username, &session.Expiry)
72     if err != nil {
73         if err == sql.ErrNoRows {
74             return model.Session{}, err
75         }
76         return model.Session{}, err
77     }
78
79     return session, nil
80 }

```

```
141 err = model.ErrorResponse{
142     Error: "ID, name, or study code is empty",
143 }
144 var response, _ = json.Marshal(err)
145 w.WriteHeader(400)
146 w.Write([]byte(response))
147 } else {
148     errB = json.Unmarshal(body, &newUser)
149     if errB != nil {
150         err = model.ErrorResponse{
151             Error: "ID, name, or study code is empty",
152         }
153         var response, _ = json.Marshal(err)
154         w.WriteHeader(400)
155         w.Write([]byte(response))
156     } else {
157         if newUser.ID == "" || newUser.Name == "" || newUser.Studycode == "" {
158             err = model.ErrorResponse{
159                 Error: "ID, name, or study code is empty",
160             }
161             var response, _ = json.Marshal(err)
162             w.WriteHeader(400)
163             w.Write([]byte(response))
164         } else {
165             var _, exist = FindStudyProgram(newUser.Studycode)
166             if exist {
167                 var listuser, _ = Readdata("./data/users.txt")
168                 if len(listuser) > 0 {
169                     var _, exist = FindUserId(newUser.ID)
170                     if exist {
171                         err = model.ErrorResponse{
172                             Error: "user id already exist",
173                         }
174                     }
175                 }
176             }
177         }
178     }
179 }
```

```
141 err = model.ErrorResponse{
142     Error: "ID, name, or study code is empty",
143 }
144 var response, _ = json.Marshal(err)
145 w.WriteHeader(400)
146 w.Write([]byte(response))
147 } else {
148     errB = json.Unmarshal(body, &newUser)
149     if errB != nil {
150         err = model.ErrorResponse{
151             Error: "ID, name, or study code is empty",
152         }
153         var response, _ = json.Marshal(err)
154         w.WriteHeader(400)
155         w.Write([]byte(response))
156     } else {
157         if newUser.ID == "" || newUser.Name == "" || newUser.Studycode == "" {
158             err = model.ErrorResponse{
159                 Error: "ID, name, or study code is empty",
160             }
161             var response, _ = json.Marshal(err)
162             w.WriteHeader(400)
163             w.Write([]byte(response))
164         } else {
165             var _, exist = FindStudyProgram(newUser.Studycode)
166             if exist {
167                 var listuser, _ = Readdata("./data/users.txt")
168                 if len(listuser) > 0 {
169                     var _, exist = FindUserId(newUser.ID)
170                     if exist {
171                         err = model.ErrorResponse{
172                             Error: "user id already exist",
173                         }
174                     }
175                 }
176             }
177         }
178     }
179 }
```

Lampiran 3 Dokumen Kegiatan

