# CATAT KERJA - MANAJEMEN KEUANGAN MIKRO

# FREELANCER


**Disusun Oleh**

**ABDULLAH AZZAM RABBANI**

**10240038**

**AMMAR ICHSAN ANTHONY**

**10240005**

**KELAS 10.3A.01**



**PROGRAM STUDI REKAYASA PERANGKAT LUNAK**

**UNIVERSITAS BINA SARANA INFORMATIKA**

**MARGONDA**

**2025/2026**

# KATA PENGANTAR

Jakarta, 09 September 2025

penyusun

# DAFTAR ISI

# BAB I
# PENDAHULUAN

## 1.1. Latar Belakang

ketidakstabilan pendapatan freelancer yang menyebabkan kesulitan dalam perencanaan keuangan pribadi, alokasi dana untuk pajak, tabungan, dan kebutuhan operasional. Meskipun freelancer mungkin pandai melacak jam kerja, mereka sering kali gagal mengubah pendapatan yang masih "di atas kertas" (dari jam yang baru dicatat) menjadi alokasi dana yang jelas dan terstruktur.

Dikutip dari penelitian Purwasih, R., & Firdaus bahwasanya *"Di era digital saat ini, banyak freelancer menghadapi tantangan dalam mengelola waktu mereka dan memantau kinerja mereka... [Sistem yang dibuat harus bertujuan] untuk memberikan solusi yang efektif dengan memanfaatkan teknologi untuk meningkatkan efisiensi kerja... [dan] mengurangi beban administratif..."*[1]. Permasalahan yang dihadapi oleh freelancer tidak hanya bersifat teknis, tetapi juga fundamental terkait manajemen waktu dan administrasi keuangan. Hal ini didukung oleh kajian akademis yang menunjukkan *bahwa meskipun bekerja dari rumah (Work from Home) memiliki dampak positif pada keseimbangan kerja-hidup, namun jika tidak dilakukan dengan benar, ada kemungkinan WFH akan memberikan dampak negatif pada keseimbangan kerja-hidup pekerja* (Belaman et al., 2022). Selain itu, *pekerja harus mengelola waktu mereka dengan sangat baik karena mereka cenderung menyeimbangkan pekerjaan dan waktu keluarga* (Belaman et al., 2022). Oleh karena itu, kebutuhan freelancer terhadap dukungan teknologi untuk mengatasi beban administrasi dan manajemen waktu sangat mendesak.

Diperlukan perangkat lunak yang tidak hanya melacak waktu tetapi juga berfungsi sebagai asisten keuangan mikro otomatis. perangkat lunak ini dirancang untuk mengatasi masalah inti pekerja lepas dengan mengintegrasikan fitur Alokasi Dana Instan (Mikro-budgeting). Dengan sistem yang memaksa pekerja lepas untuk menginvestasikan uang virtual ke pos-pos penting (seperti Pajak dan Tabungan)

tepat setelah mereka mencatat pekerjaan mereka, mereka diharapkan dapat mengatur keuangan, mengurangi beban pajak, dan memiliki arus kas yang lebih teratur.

## 1.2. Identifikasi Masalah

Identifikasi masalah ini berfokus pada empat area kritis yang sering menjadi hambatan bagi *freelancer* yang telah berhasil mencatat jam kerjanya tetapi gagal dalam mengelola hasil finansial dari waktu tersebut.

1) Alokasi Dana yang Reaktif

   Freelancer cenderung menunda alokasi dana penting (seperti untuk Pajak dan Tabungan) hingga uang dari klien benar-benar diterima. Hal ini menyebabkan dana sering terpakai lebih dulu, sehingga alokasi menjadi tidak disiplin dan berantakan.

2) Sulitnya Proyeksi Kewajiban Pajak

   Tidak ada sistem yang otomatis menghitung perkiraan dana pajak yang harus disisihkan secara real-time berdasarkan jam kerja yang baru dicatat. Ini membuat freelancer sering terkejut saat harus membayar pajak besar.

3) Waktu Penagihan yang Tidak Optimal

   Freelancer sering menunda penagihan atau lupa bahwa milestone proyek sudah tercapai, yang mengakibatkan arus kas (cash flow) tertunda.

4) Kelemahan Dana Penyangga (Buffer)

   Kurangnya sistem untuk memvisualisasikan dan menargetkan akumulasi dana penyangga (financial buffer) untuk menutupi ketidakpastian antara tanggal penagihan dan tanggal pembayaran.

## 1.3. Ruang Lingkup Perangkat Lunak

### 1.3.1. Fungsi Utama

1. Modul Pelacakan Waktu (*Time Tracking*)
   a) Fitur timer untuk mencatat jam kerja per proyek dan klien secara real-time.

b) Menghitung total Nilai Rupiah yang dihasilkan secara instan saat timer dihentikan.

2. Modul Alokasi Dana Instan (Micro-Budgeting)

    a) Sistem meminta pengguna untuk membagi (mengalokasikan) persentase dari Nilai Rupiah yang baru dihasilkan ke dalam pos-pos virtual (misalnya: Pajak dan Tabungan) sebelum dana tersebut ditagih.
    b) Menampilkan total dana yang sudah terakumulasi secara virtual untuk setiap pos alokasi.

3. Modul Penagihan dan Faktur (*Invoicing*)

    a) Memberikan saran kapan waktu terbaik untuk menagih klien.
    b) Pembuatan Faktur Otomatis: Membuat dokumen Faktur/Invoice (PDF) yang rapi berdasarkan data jam kerja dan tarif yang tercatat.

### 1.3.2. Batasan

Batasan ini menjelaskan bahwa meskipun Catat-Kerja mengelola alokasi dana secara virtual, ia bukanlah platform perbankan atau akuntansi riil. PL Catat-Kerja tidak akan mencakup fungsi-fungsi berikut:

1) Perangkat Lunak Catat-Kerja hanya dikembangkan untuk penggunaan freelancer individual (single user).
2) Aplikasi tidak terhubung dengan sistem perbankan atau payment gateway untuk transfer dana riil; alokasi dana hanya bersifat pencatatan virtual (pembukuan internal).
3) Aplikasi bukanlah sistem akuntansi penuh, sehingga tidak mencakup pencatatan biaya operasional yang mendalam, rekonsiliasi bank, atau pelaporan pajak resmi.
4) Pengembangan berfokus pada aplikasi berbasis web.

## 1.4. Rumusan Masalah

1) Bagaimana cara membuat fitur pelacak waktu yang dapat dengan cepat mengubah waktu kerja yang dicatat menjadi uang (nilai yang ditagih)?
2) Bagaimana menerapkan fitur Micro-Budgeting (alokasi dana instan) yang memaksa freelancer untuk menyisihkan persentase pendapatan virtual untuk Pajak dan Tabungan secara proaktif?
3) Bagaimana sistem dapat memberikan notifikasi pintar untuk menentukan waktu penagihan yang optimal kepada klien berdasarkan milestone atau akumulasi jam?
4) Bagaimana menerapkan fitur Micro-Budgeting yang memaksa freelancer untuk menyisihkan persentase pendapatan virtual untuk Pajak dan Tabungan secara proaktif?
5) Seberapa signifikan kebutuhan freelancer terhadap dukungan teknologi untuk mengatasi beban administrasi dan manajemen waktu yang berdampak pada keberlanjutan karir?

### 1.5. Tujuan Penelitian

Tujuan yang ingin dicapai dari penelitian dan pengembangan Perangkat Lunak Catat-Kerja ini adalah:

1) Membangun fungsionalitas Time Tracker yang mampu mengonversi jam kerja menjadi nilai uang secara otomatis.
2) Menciptakan fitur Micro-Budgeting yang efektif untuk mempromosikan kedisiplinan alokasi dana proaktif (Pajak dan Tabungan) bagi freelancer.
3) Mengimplementasikan notifikasi cerdas untuk optimalisasi waktu penagihan.
4) Menghasilkan Perangkat Lunak Catat-Kerja yang terintegrasi dan siap diuji, yang dapat mempermudah proses administrasi dan perencanaan keuangan mikro freelancer.

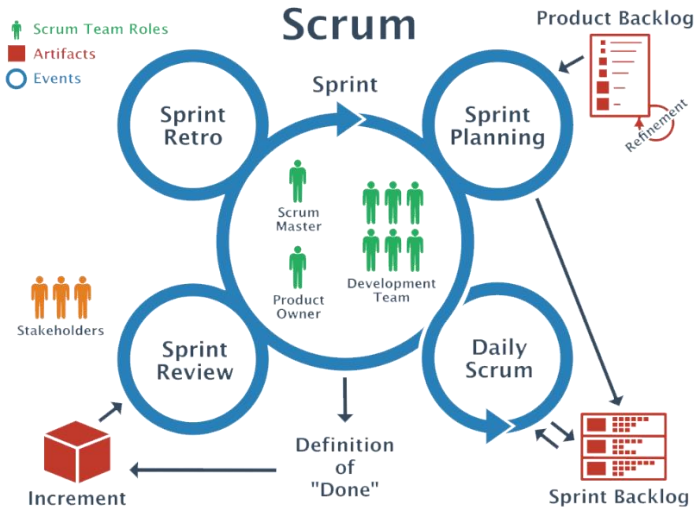
### 1.6. Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan manfaat bagi berbagai pihak, yaitu:

1) Untuk Freelancer: ini menawarkan solusi komprehensif yang mengurangi beban administrasi penagihan, menstabilkan arus kas, dan meningkatkan disiplin keuangan (alokasi dana proaktif).
2) Bagi Akademisi: Menambah khazanah ilmu pengetahuan di bidang Rekayasa Perangkat Lunak, khususnya dalam implementasi model Agile (Scrum) untuk mengembangkan FinTech mikro yang berfokus pada user behavior.
3) Bagi Pengembangan Industri Software: Menjadi studi kasus dalam perancangan aplikasi software as a service (SaaS) yang secara khusus menargetkan kebutuhan manajemen keuangan unik freelancer.

### 1.7. Model Proses pengembangan

Model proses pengembangan yang dipilih adalah Model Scrum. Model ini merupakan kerangka kerja Agile yang sangat sesuai untuk pengembangan perangkat lunak yang membutuhkan interaksi pengguna yang cepat, penyesuaian desain UI/UX, dan rilis fitur bertahap (incremental) guna memvalidasi fitur unik (Alokasi Dana Mikro) dengan cepat.




### 1.8. Tahapan Pengembangan (Siklus Scrum)

Pengembangan PL akan melalui siklus berulang sebagai berikut:

a) Product Backlog yaitu langkah awal dalam menyusun daftar fitur prioritas untuk Software Backlog sesuai dengan kebutuhan Hipster (pengalaman pengguna) dan Hustler (nilai bisnis).
b) Sprint Planning yaitu memilih fitur dari Product Backlog yang akan dikerjakan dalam Sprint tertentu, dan pembagian tugas tim (Hacker, Hipster, Hustler).
c) Mengadakan rapat singkat untuk mengevaluasi dan mengidentifikasi masalah
d) Evaluasi tim internal terhadap proses kerja untuk perbaikan di Sprint berikutnya.

# BAB II

# BAB III
# ANALISIS DAN PERANCANGAN APLIKASI

## 3.1 Penjelasan Secara Umum tentang Objek yang Diambil

Objek yang diambil dalam perancangan ini adalah Catat-Kerja, sebuah Perangkat Lunak (PL) berbasis web yang berfungsi sebagai Sistem Manajemen Waktu dan Keuangan Mikro yang ditujukan khusus untuk pekerja lepas (freelancer) individu.

Perangkat Lunak ini dikembangkan untuk mengatasi ketidakstabilan arus kas dan kedisiplinan finansial freelancer dengan mengintegrasikan dua fungsi utama: Pelacakan Waktu (Time Tracking) dan Alokasi Dana Proaktif (Micro-Budgeting). Tujuan utamanya adalah memastikan bahwa setiap jam kerja yang dicatat secara langsung dikonversi menjadi nilai uang, dan nilai uang tersebut segera dialokasikan secara virtual ke pos-pos kewajiban (seperti Pajak) dan investasi (seperti Tabungan), sehingga mengubah perilaku keuangan pengguna dari reaktif menjadi proaktif.

## 3.2 Analisis Aplikasi

### 3.2.1 Analisis Masalah

Analisis masalah dilakukan untuk menegaskan bahwa PL Catat-Kerja dikembangkan berdasarkan solusi untuk masalah yang nyata dan telah divalidasi, yaitu:

1. Kegagalan Alokasi Dana Proaktif: Freelancer saat ini cenderung menunda alokasi dana kewajiban (Pajak) hingga uang dari klien diterima, yang rentan menyebabkan defisit dana tak terduga.
2. Sulitnya Proyeksi Kewajiban: Tidak adanya alat otomatis yang menghitung proyeksi dana pajak dan dana penyangga yang harus disisihkan berdasarkan jam kerja yang sedang berlangsung.
3. Inefisiensi Penagihan: Proses penagihan yang tidak terstruktur atau tertunda (karena lupa) mengakibatkan terhambatnya arus kas.

### 3.2.2 Analisis Kebutuhan

Berdasarkan masalah di atas, PL Catat-Kerja dirancang untuk memenuhi tiga kebutuhan utama bagi freelancer:

a) Kebutuhan Manajemen Waktu & Nilai: Pencatatan waktu dan konversi otomatis menjadi Nilai Rupiah.
b) Kebutuhan Alokasi Dana Proaktif: Fitur pop-up wajib untuk alokasi dana virtual dan dasbor akumulasi dana.
c) Kebutuhan Administrasi Penagihan Efisien: Notifikasi cerdas dan pembuatan faktur otomatis.

## 3.3 Pemodelan Proses Bisnis dan Fungsional

A. Use Case Diagram

Aktor utama adalah Freelancer (Pengguna Sistem) yang melakukan use case utama: Melacak Waktu & Nilai, Melakukan Alokasi Dana Instan, Mengelola Faktur, dsb.

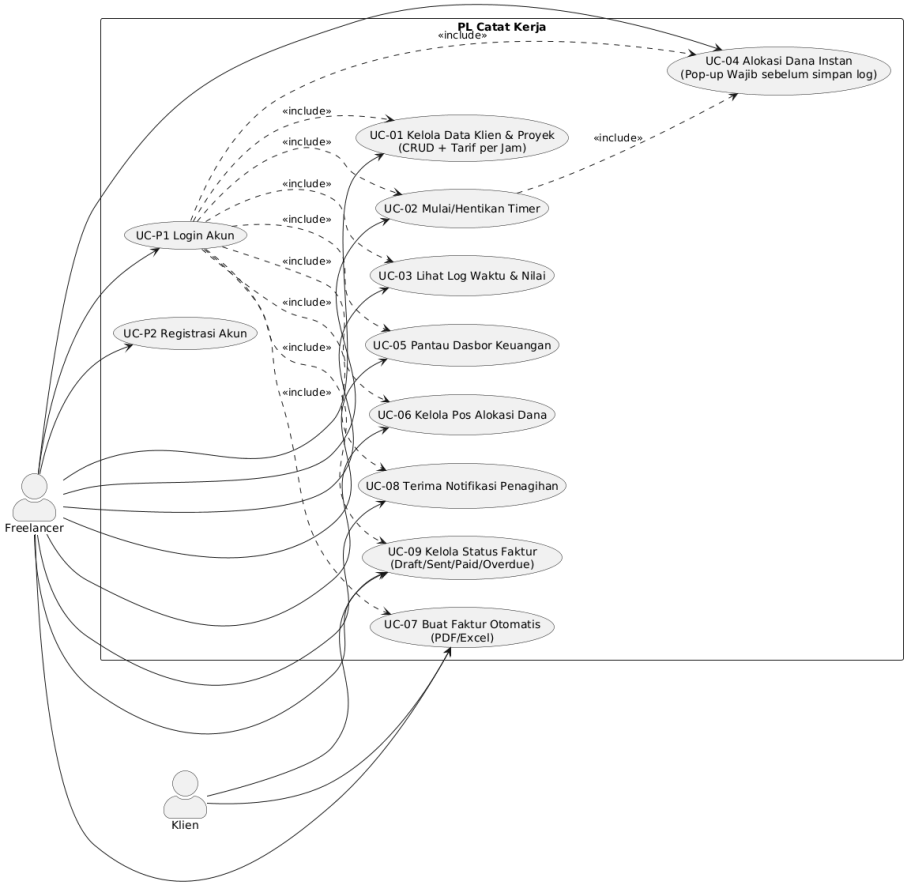


Table Deskripsi

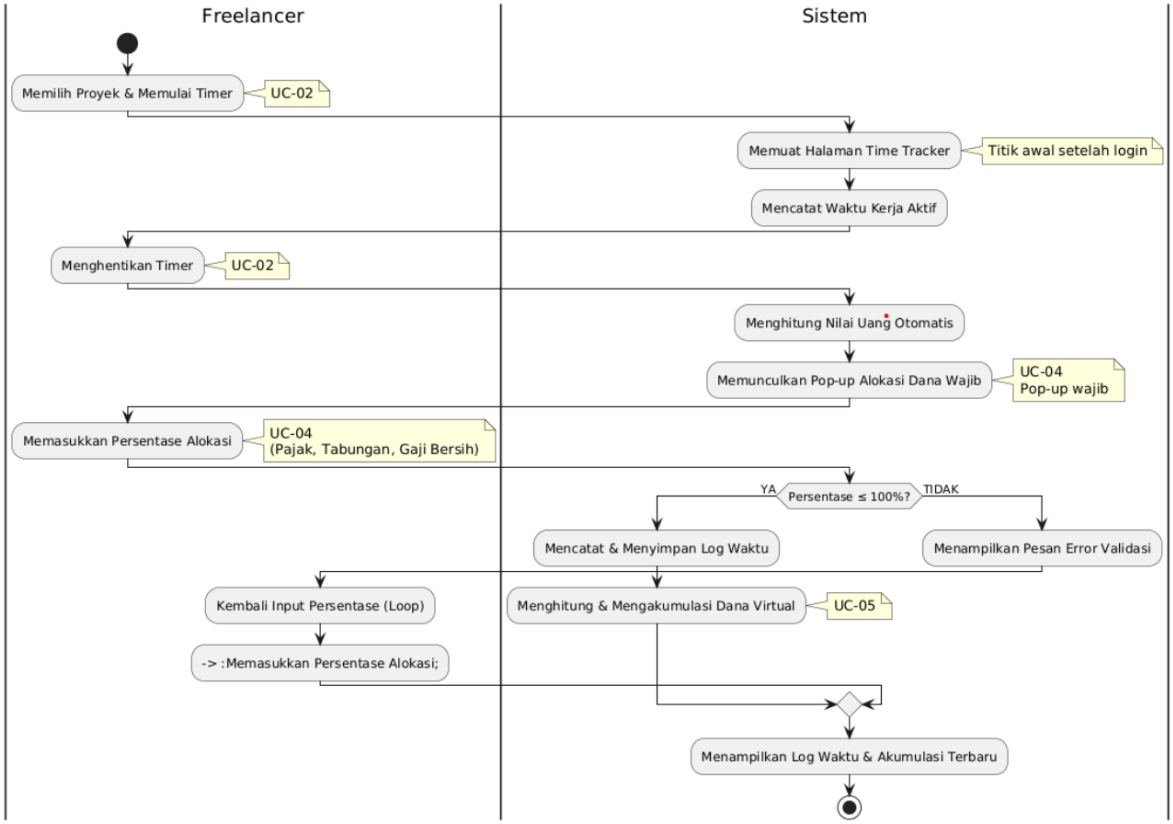
| ID UC | Nama *Use Case* | Aktor | Deskripsi Singkat |
|---|---|---|---|
| UC-01 | Kelola Data Klien & Proyek | Freelancer | Mengatur (menambah, mengubah, menghapus) data Klien dan detail Proyek, termasuk penetapan **Tarif per Jam** (CRUD). |
| UC-02 | Mulai/Hentikan Timer | Freelancer | Mengaktifkan dan menonaktifkan penghitungan waktu kerja. Proses ini memerlukan (<<include>>) UC-04 Melakukan Alokasi Dana Instan saat timer dihentikan. |
| UC-03 | Lihat Log Waktu & Nilai | Freelancer | Menampilkan riwayat sesi kerja yang tercatat, durasi, dan nilai Rupiah yang dihasilkan. |
| UC-04 | Melakukan Alokasi Dana Instan | Freelancer | Fitur Wajib/Pop-up: Menyisihkan persentase pendapatan ke pos Pajak dan Tabungan/Investasi sebelum log waktu disimpan. |
| UC-05 | Pantau Dasbor Keuangan | Freelancer | Menampilkan ringkasan visual dari Akumulasi Dana Virtual (Pajak, Tabungan) dan proyeksi cash flow. |
| UC-06 | Kelola Pos Alokasi | Freelancer | Mengatur atau menyesuaikan persentase *default* alokasi dana yang dibutuhkan *freelancer* untuk Pajak, Tabungan, dll. |
| UC-07 | Buat Faktur Otomatis | Freelancer/Klient | Menggabungkan data log waktu dan alokasi dana untuk menghasilkan dokumen Faktur resmi (PDF/Excel) yang ditujukan kepada Klien. |

| | | | |
|---|---|---|---|
| UC-08 | Terima Notifikasi Penagihan | Freelancer | Sistem memberikan peringatan cerdas (misalnya, milestone jam kerja tercapai) untuk menyarankan waktu penagihan optimal. |
| UC-09 | Kelola Status Faktur | Freelancer/Klient | Mengubah status faktur (Draft, Sent, Paid, Overdue), yang memengaruhi data cash flow di Dasbor. |
| UC-P1 | Login Akun | Freelancer | Proses otentikasi yang wajib dilakukan (<<include>>) sebelum mengakses semua Use Case fungsional (UC-01 s/d UC-09). |
| UC-P2 | Registrasi Akun | Freelancer | Proses pembuatan akun baru yang mencakup input data identitas dasar pengguna. |

B. Activity Diagram

Menggambarkan alur kritis Pencatatan Waktu Hingga Alokasi Dana Proaktif: Mulai Timer -> Stop -> Sistem Menghitung Nilai Uang -> Sistem Memunculkan Pop-up Alokasi Wajib -> Freelancer Memasukkan Persentase -> Sistem Memperbarui Dana Virtual.

## 1. Visualisasi Diagram




## 2. Deskripsi Alur Proses

Berikut adalah urutan langkah-langkah yang terjadi, dimulai dari pencatatan waktu hingga penyimpanan alokasi dana virtual:

1. Sistem [START]: Memuat Halaman Time Tracker setelah Freelancer berhasil login.
2. Freelancer: Memilih Proyek spesifik dan Memulai Timer.
3. Sistem: Mulai Mencatat Waktu Kerja Aktif (Durasi) secara real-time.
4. Freelancer: Menghentikan Timer setelah sesi kerja selesai.
5. Sistem: Menghitung Nilai Uang Otomatis dengan rumus: Durasi $\times$ Tarif/Jam.

6. Sistem: Memunculkan Pop-up Alokasi Dana Wajib untuk memicu langkah Micro-Budgeting proaktif.
7. Freelancer: Memasukkan Persentase Alokasi (Wajib diisi untuk Pajak Penghasilan, Tabungan/Investasi, dan Sisa Gaji Bersih).
8. Sistem [DECISION]: Memeriksa apakah Persentase Total Alokasi (Pajak + Tabungan) $\le$ 100%.
9. Sistem [GUARD: NO]: Jika total alokasi melebihi 100%, sistem menampilkan pesan Error dan kembali ke langkah Memasukkan Persentase Alokasi (Loop).
10. Sistem [GUARD: YES]: Jika alokasi valid, Sistem Mencatat dan Menyimpan Log Waktu serta data alokasi yang sudah diperinci.
11. Sistem: Menghitung dan Mengakumulasi Dana Virtual (memperbarui total Dana Pajak dan Dana Tabungan di basis data).
12. Sistem [END]: Menampilkan Log Waktu & Akumulasi Terbaru di dashboard.

### 3.4 Perancangan Antarmuka Pengguna (UI/UX)

Perancangan Antarmuka Pengguna (UI) dan Pengalaman Pengguna (UX) untuk aplikasi Catat-Kerja mengadopsi pendekatan Single Page Interface (SPI) atau Halaman Tunggal. Pendekatan ini dipilih untuk memaksimalkan efisiensi kerja *freelancer* dengan meminimalkan waktu yang dihabiskan untuk navigasi, sehingga pengguna dapat langsung fokus pada tugas inti dan administrasi penting.

Desain ini secara khusus memenuhi Kebutuhan Non-Fungsional Kemudahan Penggunaan (*Usability*) dan secara strategis memprioritaskan aktivitas Kuadran II (administrasi dan perencanaan) di halaman utama (*Dashboard*).

#### A. Struktur Layout Halaman Utama (Dashboard SPI)

Halaman utama dirancang dengan membagi layar menjadi tiga zona fungsional utama yang terintegrasi secara visual, di mana setiap zona mewakili satu pilar fungsionalitas aplikasi:

| Zona Desain | Fungsi Utama | Kebutuhan Fungsional Terkait |
|---|---|---|
| **Zona 1: Time Tracking & Kontrol Proyek** | Pusat *input* dan kontrol waktu kerja *real-time*. | Kebutuhan 1 & 2 |
| **Zona 2: Administrasi Keuangan & Arus Kas** | Tinjauan cepat status pembayaran dan *invoice*. | Kebutuhan 3 & 5 |
| **Zona 3: Perencanaan Dana Proaktif** | Modul untuk mengatur dan menyajikan simulasi alokasi dana otomatis. | Kebutuhan 4 & 5 |

**B. Detail Komponen Antarmuka Pengguna**

**1. Zona 1: Time Tracking & Kontrol Proyek**

Zona ini diposisikan di area paling mudah dijangkau di halaman (biasanya sisi atas/kiri) untuk memastikan fungsi pelacakan waktu dapat segera diakses.

a) Modul *Timer* Aktif: Berisi tombol START/STOP besar yang menonjol dan menampilkan durasi *real-time* kerja yang sedang berlangsung.
b) Pemilihan Tugas: *Dropdown* atau *field* yang memungkinkan pengguna memilih Klien dan Proyek/Tugas dari daftar sebelum *timer* dimulai.
c) Log Waktu Hari Ini: Ringkasan entri waktu yang baru dibuat, dilengkapi dengan opsi Edit dan Delete langsung.

**2. Zona 2: Administrasi Keuangan & Arus Kas**

Zona ini menyajikan data keuangan yang dihasilkan dari *Time Tracking*. Fungsinya adalah mengubah data mentah (jam kerja) menjadi data administrasi (penagihan).

a) Kartu Statistik Cepat: Menampilkan metrik utama seperti "Total Jam Kerja Bulan Ini" dan "Total Pending Invoice Value".
b) Daftar *Invoice* Ringkas: *Widget* yang menunjukkan daftar *invoice* dengan status kritis (misalnya, *Overdue* atau *Due Soon*), dengan tombol aksi cepat "Buat Invoice Baru" yang memicu *modal* untuk proses otomatisasi penagihan.

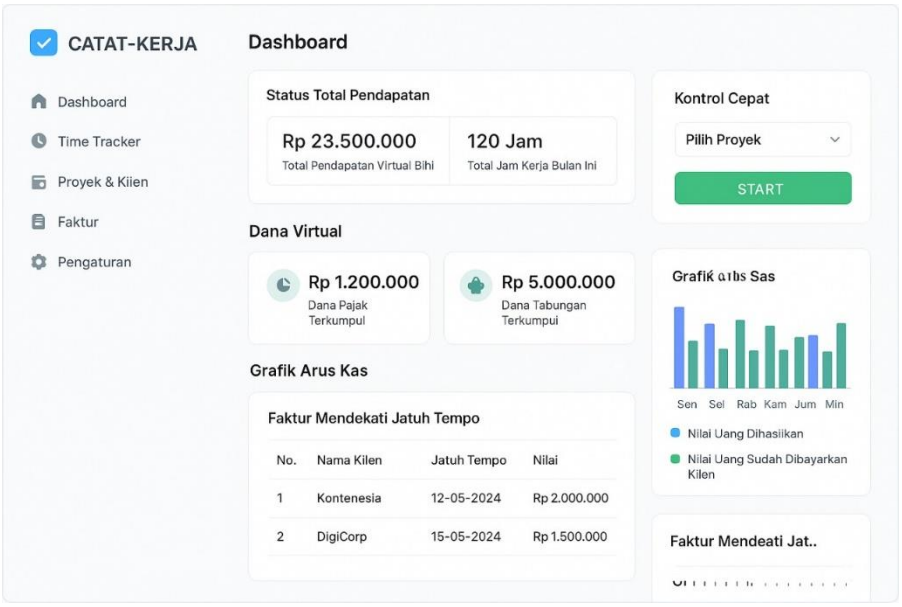
**3. Zona 3: Perencanaan Dana Proaktif**

Zona ini merupakan fitur unik aplikasi yang mengatasi Kesenjangan Perencanaan Keuangan Proaktif. Zona ini menyajikan simulasi alokasi dana dan pengaturan persentase.

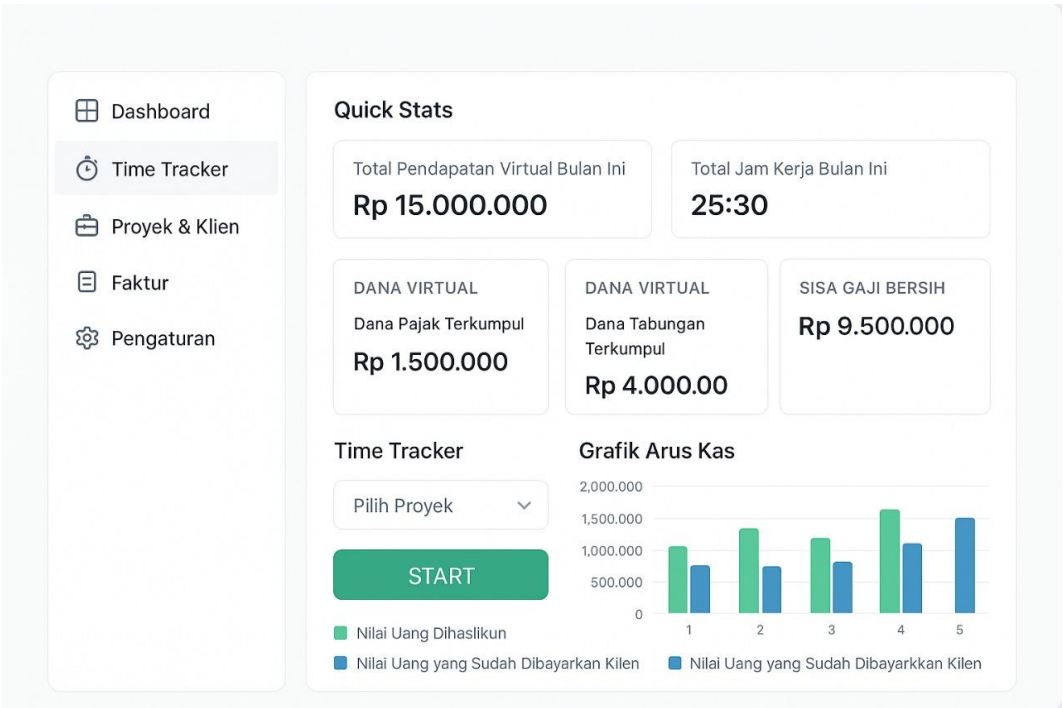
a) Pengaturan Persentase Alokasi: Tampilan yang memungkinkan pengguna mengatur dan memvisualisasikan pembagian persentase dana untuk Pajak, Tabungan/Investasi, dan Dana Bersih.
b) Simulasi Dana (Proaktif): Tampilan kunci yang menunjukkan bagaimana pendapatan yang baru diterima atau *invoice* yang akan dibayar akan didistribusikan sesuai persentase yang sudah diatur (misalnya: "Dari $1,000 yang akan dibayar, $150 dialokasikan untuk Pajak").

a) Grafik Alokasi: Visualisasi grafik (*pie chart* atau *bar chart*) yang menunjukkan distribusi dana secara keseluruhan, membantu *freelancer* secara visual mengukur stabilitas finansial mereka.

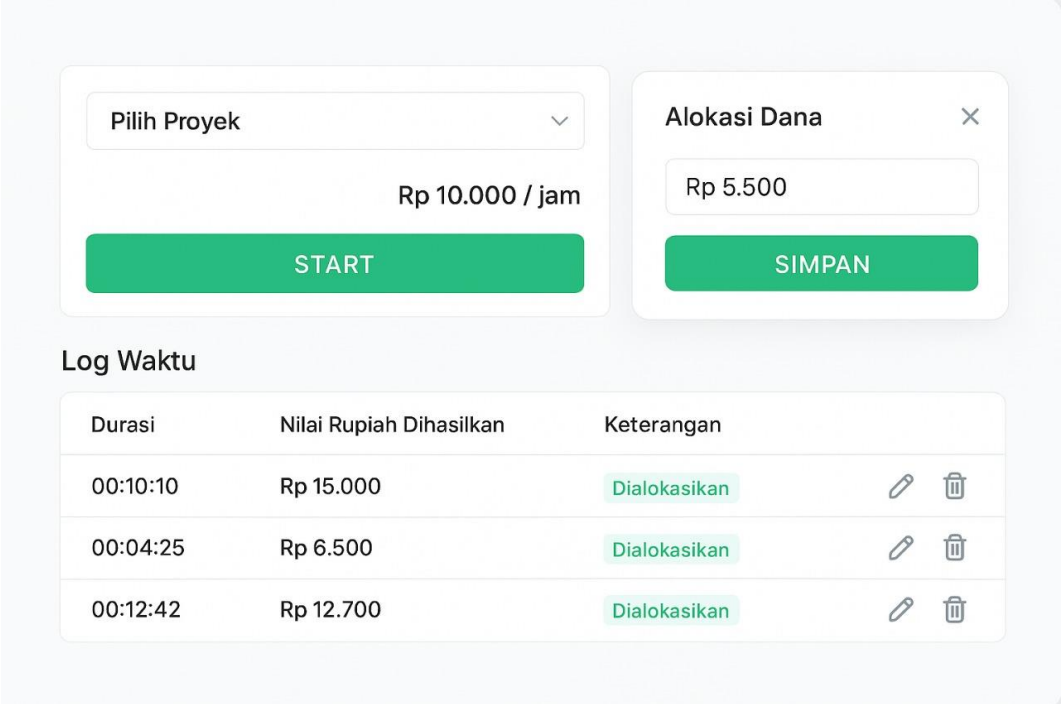
**Dashboard**




**Time Tracker**



**Alokasi Dana**




# BAB IV
# PEMBAHASAN

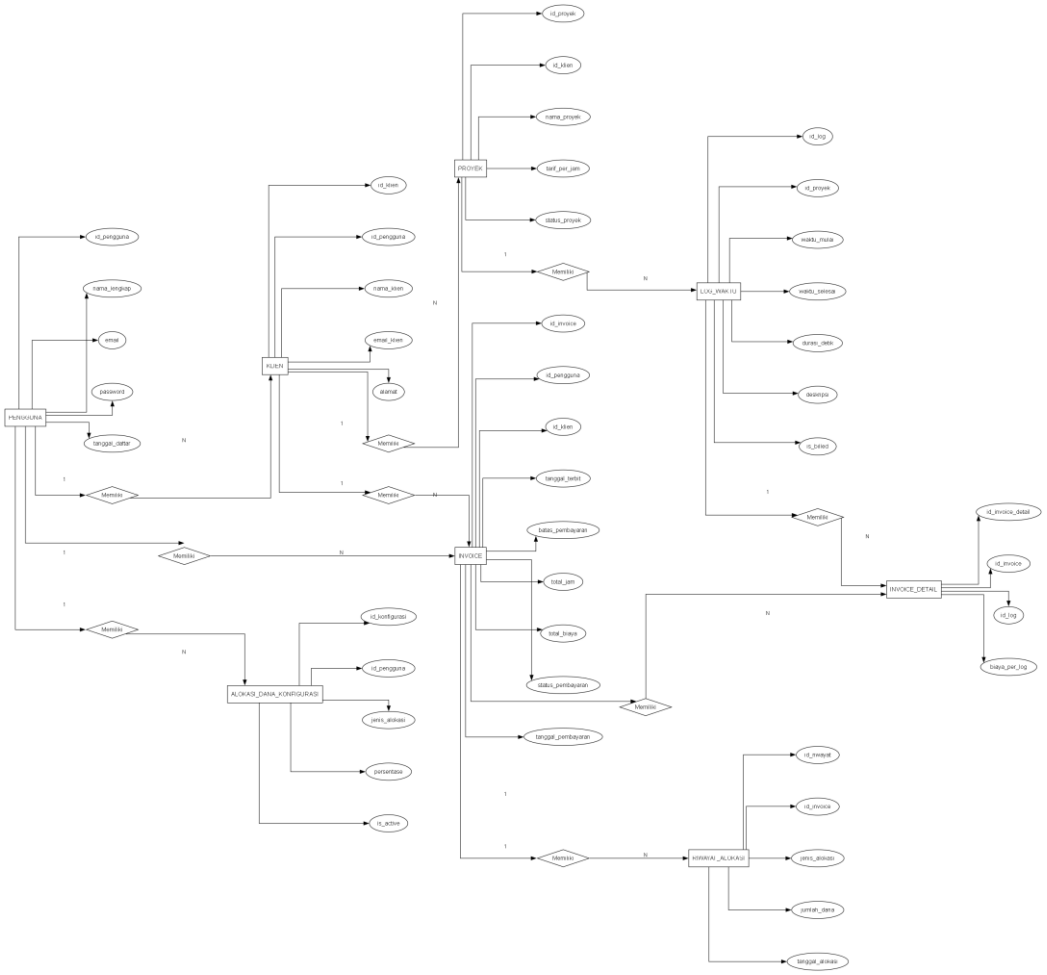
## 4.1 Perancangan Sistem

Sub-bab ini berfokus pada perancangan logika dan data sistem yang akan digunakan untuk mengimplementasikan fungsionalitas **Catat-Kerja**, yang telah didefinisikan dalam Bab 3.

### 4.1.1 Perancangan Basis Data

Perancangan basis data adalah fondasi logis untuk menyimpan semua informasi kunci mengenai manajemen waktu, klien, penagihan, dan alokasi dana.

#### 4.1.1.1 Entity Relationship Diagram (ERD)



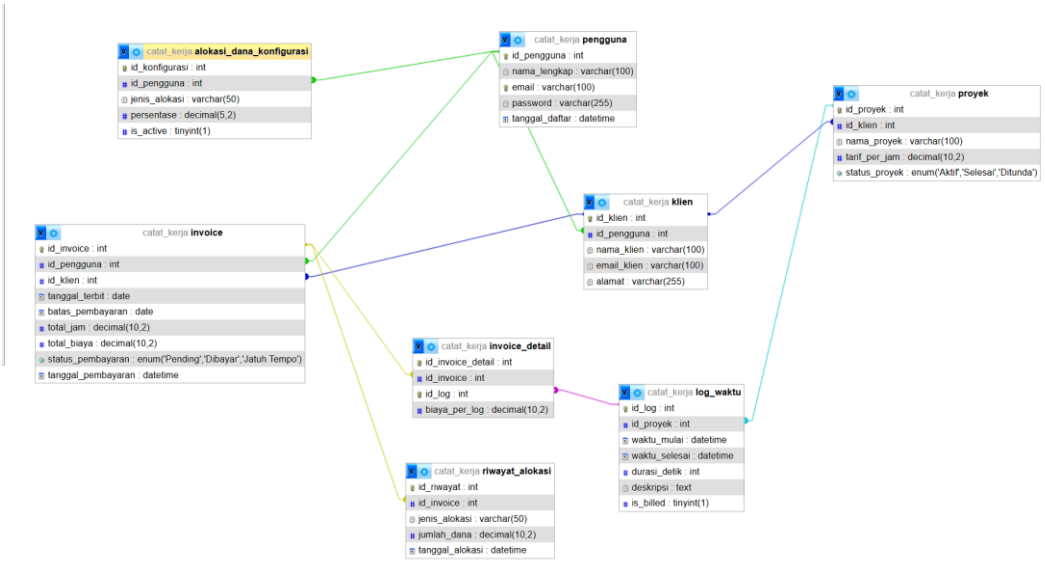
| No. | Entitas Sumber | Kunci Relasi (FK) | Entitas Tujuan | Kardinalitas | Penjelasan |
|---|---|---|---|---|---|
| 1 | PENGGUNA (Freelancer) | → id_pengguna | KLIEN | One-to-Many (1:M) | Satu PENGGUNA memiliki banyak KLIEN. |
| 2 | KLIEN | → id_klien | PROYEK | One-to-Many (1:M) | Satu KLIEN menawarkan banyak PROYEK. |
| 3 | PROYEK | → id_proyek | LOG_WAKTU | One-to-Many (1:M) | Satu PROYEK menghasilkan banyak *record* |

| | | | | | |
|---|---|---|---|---|---|
| | | | | | LOG_WAKTU . |
| 4 | KLIEN | → id_klien | INVOICE | One-to-Many (1:M) | Satu KLIEN menerima banyak INVOICE. |
| 5 | INVOICE | → id_invoice | INVOICE_ DETAIL | One-to-Many (1:M) | Satu INVOICE memiliki banyak DETAIL item yang ditagihkan. |
| 6 | LOG_WA KTU | → id_log | INVOICE_ DETAIL | One-to-One (1:1) | INVOICE_DE TAIL adalah tabel *junction* yang memecah relasi M:M antara INVOICE dan LOG_WAKTU (di mana satu log waktu hanya masuk ke satu invoice). |
| 7 | PENGGU NA | → id_penggu na | ALOKASI_ DANA_KO NFIGURAS I | One-to-Many (1:M) | PENGGUNA mengatur banyak konfigurasi alokasi (Pajak, Tabungan, dll.). |
| 8 | INVOICE | → id_invoice | RIWAYAT _ALOKASI | One-to-Many (1:M) | Pembayaran satu INVOICE menghasilkan banyak *record* |

| | | | | | |
|---|---|---|---|---|---|
| | | | | | RIWAYAT_ALOKASI. |

### 4.1.1.2 Logical Record Structure (LRS)

LRS adalah representasi logis dari ERD, yang menggambarkan tabel-tabel basis data, *field* (*attributes*), dan kunci (*primary key/foreign key*) yang akan digunakan untuk mengimplementasikan sistem. LRS memastikan bahwa relasi Many-to-Many (M:M) telah dipecah menjadi relasi One-to-Many (1:M) dan semua tabel telah dinormalisasi hingga **3NF**.




### 4.1.1.3 Specification File (Spesifikasi Tabel)

Database ini dirancang untuk mendukung proses pencatatan waktu kerja freelancer, pembuatan invoice, serta pengelolaan dan alokasi dana dari pembayaran. Struktur database menggunakan model Relational Database Management System (RDBMS) dengan aturan integritas referensial melalui Foreign Key untuk menjaga konsistensi data. Adapun spesifikasi tabel – tabel seperti berikut:

a) TABEL PENGGUNA

Nama File : pengguna.ibd

Akronim : pengguna

Fungsi : menyimpan data pengguna/freelancer yang menggunakan aplikasi

Tipe File : File Master

Organisasi File : Index Sequential

Akses File : Random

Media : Harddisk

Perkiraan Panjang Record : ± 300 byte

Kunci Record (Primary Key) : id_pengguna

Software DBMS : MySQL (InnoDB, UTF8MB4)

| **No** | **Akronim** | **Tipe** | **Panjang** | **Keterangan** |
|---|---|---|---|---|
| 1 | id_pengguna | INT | 11 | Primary key, auto_increment |
| 2 | nama_lengkap | VARCHAR | 100 | - |
| 3 | email | VARCHAR | 100 | Unique |
| 4 | password | VARCHAR | 255 | Password terenkripsi |
| 5 | tanggal_daftar | DATETIME | - | Tanggal registrasi |

b) TABEL KLIEN

Nama File : klien

Akronim : klien.ibd

Fungsi : Menyimpan data klien pemilik proyek

Tipe File : file master

Organisasi File : Index Sequential

Akses File : random

Media : Harddisk

Panjang Record : (estimasi) 350 byte

Kunci Field : id_klien

Software : MySQL

| **No** | **Field** | **Tipe** | **Panjang** | **Keterangan** |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| 1 | id_klien | Int | 8 | Primary key, auto increment |
| 2 | nama_klien | Varchar | 150 | - |
| 3 | telepon | Varchar | 20 | - |
| 4 | email | Varchar | 150 | - |
| 5 | alamat | Text | - | - |

c) Tabel proyek

Nama File : proyek

Akronim : proyek.ibd

Fungsi : Menyimpan proyek yang dikerjakan untuk klien

Tipe File : file transaksi

Organisasi File : Index Sequential

Akses File : random

Media : Harddisk

Panjang Record : (estimasi) 400 byte

Kunci Field : id_proyek

Software : MySQL

| **No** | **Field** | **Tipe** | **Panjang** | **Keterangan** |
|---|---|---|---|---|
| 1 | id_proyek | Int | 8 | Primary key |
| 2 | id_klien | Int | 8 | Foreign key -> klien(id_klien) |
| 3 | nama_proyek | Varchar | 150 | - |
| 4 | tanggal_mulai | Date | - | - |
| 5 | tanggal_selesai | Date | - | - |
| 6 | status | Enum | - | Enum(‘baru’, ‘berjalan’, ‘selesai’) |

d) Tabel invoice

Nama File : invoice

Akronim : invoice.ibd

Fungsi : Menyimpan data tagihan terkait proyek

Tipe File : file transaksi

Organisasi File : Index Sequential

Akses File : random

Media : Harddisk

Panjang Record : (estimasi) 350 byte

Kunci Field : id_invoice

Software : MySQL

| No | Field | Tipe | Panjang | Keterangan |
|---|---|---|---|---|
| 1 | id_invoice | Int | 8 | Primary key |
| 2 | id_proyek | Int | 8 | Foreign key -> proyek |
| 3 | tanggal | Date | - | Tanggal penerbitan |
| 4 | total | Decimal | 10,2 | Total biaya |
| 5 | status | Enum | - | Enum('belum','lunas') |

e) Tabel invoice_detail

Nama File : invoice_detail

Akronim : invoice_detail.ibd

Fungsi : Menyimpan detail item dalam invoice

Tipe File : file transaksi

Organisasi File : Index Sequential

Akses File : random

Media : Harddisk

Panjang Record : (estimasi) 350 byte

Kunci Field : id_detail

Software : MySQL

| No | Field | Tipe | Panjang | Keterangan |
|---|---|---|---|---|
| 1 | id_detail | Int | 8 | Primary key |
| 2 | id_invoice | Int | 8 | Foreign key -> invoice |

| | | | | |
|---|---|---|---|---|
| 3 | deskripsi | Varchar | 200 | - |
| 4 | jumlah | Decimal | 10,2 | - |

f) Tabel log_waktu

Nama File : log_waktu

Akronim : log_waktu.ibd

Fungsi : Mencatat waktu kerja staf per proyek

Tipe File : file transaksi

Organisasi File : Index Sequential

Akses File : random

Media : Harddisk

Panjang Record : (estimasi) 300 byte

Kunci Field : id_log

Software : MySQL

| **No** | **Field** | **Tipe** | **Panjang** | **Keterangan** |
|---|---|---|---|---|
| 1 | id_log | Int | 8 | Primary key |
| 2 | id_proyek | Int | 8 | FK |
| 3 | id_pengguna | Int | 8 | FK |
| 4 | tanggal | Date | - | - |
| 5 | durasi | Int | 4 | menit/jam |

g) Tabel alokasi_dana_konfigurasi

Nama File : alokasi_dana_konfigurasi

Akronim : alokasi_dana.konfig

Fungsi : Menyimpan konfigurasi persentase pembagian dana

Tipe File : file master

Organisasi File : Index Sequential

Akses File : random

Media : Harddisk

Panjang Record : (estimasi) 200 byte

Kunci Field : id_alokasi

Software : MySQL

| No | Field | Tipe | Panjang | Keterangan |
|---|---|---|---|---|
| 1 | id_alokasi | Int | 8 | Primary key |
| 2 | persentase | Int | 4 | - |
| 3 | keterangan | Varchar | 150 | - |

h) Tabel riwayat_alokasi

Nama File : riwayat_alokasi

Akronim : riwayat.alokasi

Fungsi : Menyimpan riwayat pembagian dana berdasarkan invoice

Tipe File : file transaksi

Organisasi File : Index Sequential

Akses File : random

Media : Harddisk

Panjang Record : (estimasi) 320 byte

Kunci Field : id_riwayat

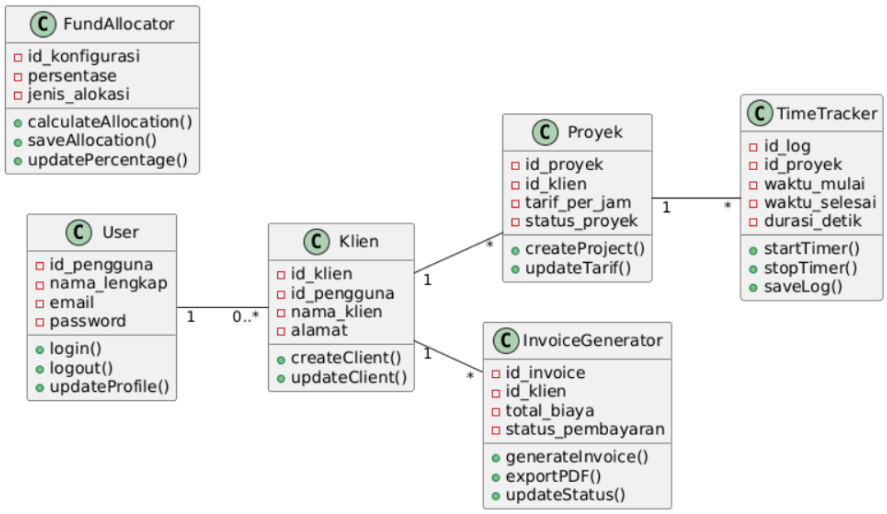
Software : MySQL

| No | Field | Tipe | Panjang | Keterangan |
|---|---|---|---|---|
| 1 | id_riwayat | Int | 8 | Primary key |
| 2 | id_invoice | Int | 8 | FK |
| 3 | jumlah | Decimal | 10,2 | dana dialokasikan |
| 4 | tanggal | Date | - | tanggal proses |

### 4.1.2 Perancangan Pemrograman Objek

#### 4.1.2.1 Class Diagram

*Class Diagram* memodelkan struktur sistem dari sudut pandang pemrograman berorientasi objek (*Object-Oriented Programming* / OOP). Kelas-kelas utama aplikasi **Catat-Kerja** akan merefleksikan entitas basis data tetapi dengan tambahan metode (*methods*) yang merepresentasikan fungsi aplikasi.



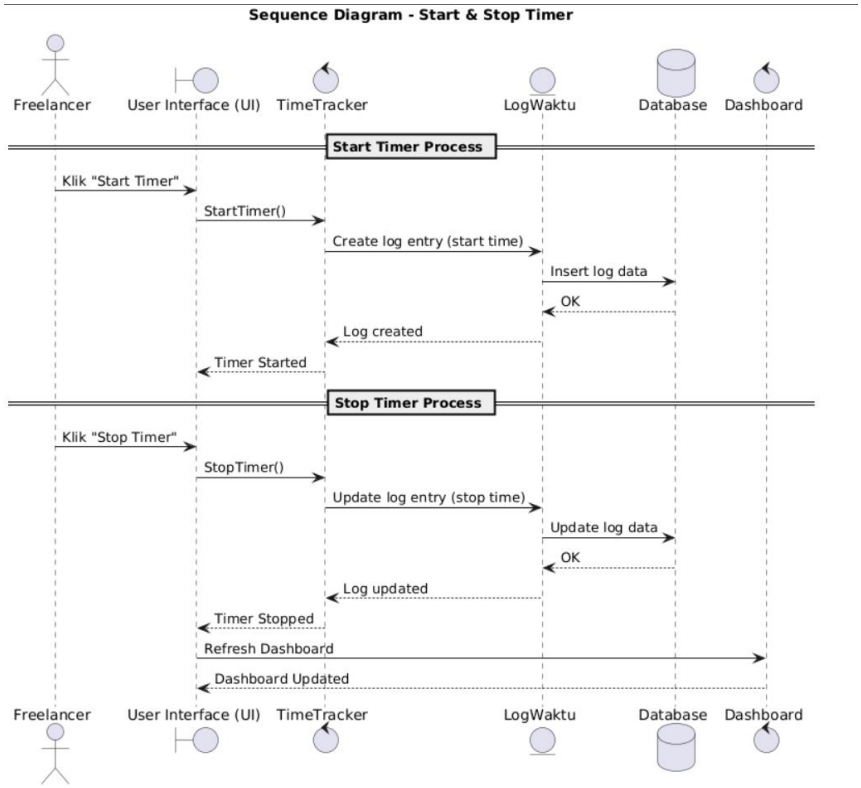

| Kelas | Atribut (Data) | Metode (Fungsi) |
|---|---|---|
| User | id_pengguna, nama_lengkap, email, password | login(), logout(), updateProfile() |
| Klien | id_klien, id_pengguna, nama_klien, alamat | createClient(), updateClient() |
| Proyek | id_proyek, id_klien, tarif_per_jam, status_proyek | createProject(), updateTarif() |
| TimeTracker | id_log, id_proyek, waktu_mulai, waktu_selesai, durasi_detik | startTimer(), stopTimer(), saveLog() |
| InvoiceGenerator | id_invoice, id_klien, total_biaya, status_pembayaran | generateInvoice(), exportPDF(), updateStatus() |
| FundAllocator | id_konfigurasi, persentase, jenis_alokasi | calculateAllocation(), saveAllocation(), updatePercentage() |

#### 4.1.2.2 Sequence Diagram

*Sequence Diagram* memvisualisasikan interaksi antar objek dalam sistem berdasarkan urutan waktu. Diagram ini sangat penting untuk memahami alur eksekusi fungsionalitas utama aplikasi, yaitu Pelacakan Waktu, Otomatisasi Penagihan, dan Alokasi Dana Proaktif.

1) Sequence Diagram: Pelacakan Waktu dan Penyimpanan Log

Diagram ini memodelkan alur kerja kritis ketika *freelancer* memulai (*start*) dan menghentikan (*stop*) pencatatan waktu untuk sebuah proyek, yang kemudian menghasilkan satu *record* di basis data (tabel `LOG_WAKTU`).

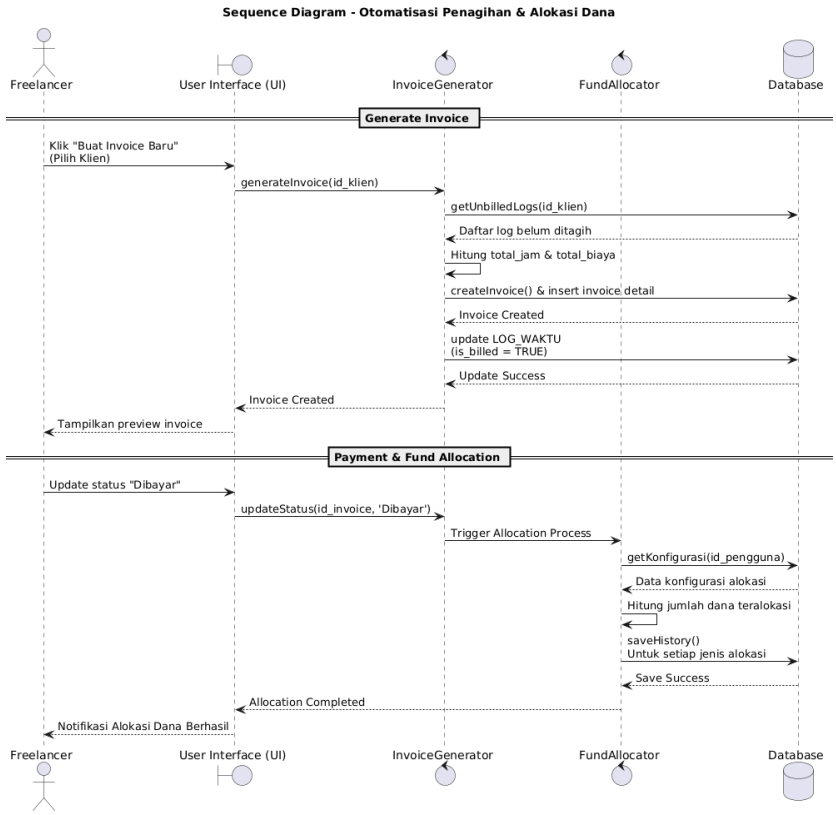



| Objek/Kelas yang Berinteraksi | Keterangan Alur dan Metode |
|---|---|
| Freelancer | Memilih proyek dan menekan tombol "Start Timer". |
| User Interface (UI) | Menerima permintaan dan mengirimkannya ke objek kontrol. |

| | |
|---|---|
| TimeTracker | Menerima pesan startTimer(id_proyek). |
| TimeTracker | Mencatat waktu saat ini (waktu_mulai). |
| TimeTracker | Memicu objek LogWaktu untuk menyimpan *draft* log. |
| Database | Menyimpan *draft* record ke tabel LOG_WAKTU (hanya waktu_mulai). |
| Freelancer | Menekan tombol "Stop Timer" setelah pekerjaan selesai. |
| User Interface (UI) | Mengirimkan pesan stopTimer() bersama ID log yang aktif. |
| TimeTracker | Mencatat waktu_selesai dan menghitung durasi_detik. |
| TimeTracker | Memanggil updateLog() pada objek LogWaktu. |
| Database | Memperbarui record di tabel LOG_WAKTU dengan waktu_selesai dan durasi_detik. |
| User Interface (UI) | Menerima konfirmasi dan menampilkan log waktu yang baru. |

2) Sequence Diagram: Otomatisasi Penagihan dan Alokasi Dana

Diagram ini memodelkan alur kerja terintegrasi yang merupakan fitur unik aplikasi: mengubah log waktu yang belum ditagih menjadi *invoice* dan secara otomatis mencatat alokasi dana ketika pembayaran diterima.




| Objek/Kelas yang Berinteraksi | Keterangan Alur dan Metode |
|---|---|
| Freelancer | Memilih klien dan memicu tombol "Buat Invoice Baru". |
| User Interface (UI) | Mengirimkan permintaan generateInvoice(id_klien) ke *InvoiceGenerator*. |
| InvoiceGenerator | Memanggil getUnbilledLogs(id_klien) dari tabel LOG_WAKTU. |
| InvoiceGenerator | Menghitung total_jam dan total_biaya berdasarkan tarif proyek. |
| InvoiceGenerator | Memanggil createInvoice() dan menyimpan record ke tabel INVOICE dan INVOICE_DETAIL. |
| InvoiceGenerator | Memperbarui status log di tabel LOG_WAKTU (is_billed = TRUE). |

| | |
|---|---|
| User Interface (UI) | Menampilkan pratinjau *invoice* yang berhasil dibuat. |
| Freelancer | (Di luar sistem) Klien membayar *invoice*. Pengguna mengubah status ke 'Dibayar'. |
| User Interface (UI) | Mengirimkan pesan updateStatus(id_invoice, 'Dibayar'). |
| FundAllocator | Otomatis dipicu setelah status pembayaran terverifikasi. |
| FundAllocator | Memanggil getKonfigurasi(id_pengguna) dari tabel ALOKASI_DANA_KONFIGURASI. |
| FundAllocator | Menghitung jumlah dana yang harus dialokasikan (jumlah_dana). |
| FundAllocator | Memanggil saveHistory() untuk setiap jenis alokasi (Pajak, Tabungan). |
| Database | Menyimpan record ke tabel RIWAYAT_ALOKASI. |
| User Interface (UI) | Menampilkan notifikasi Alokasi Dana berhasil dicatat. |
| Objek/Kelas yang Berinteraksi | Keterangan Alur dan Metode |

## 4.2 Deployment Sistem

*Deployment* adalah proses penempatan aplikasi **Catat-Kerja** dari lingkungan pengembangan ke lingkungan produksi agar dapat diakses oleh pengguna akhir (Freelancer). Aplikasi ini dirancang sebagai aplikasi *Web-based* dengan arsitektur *Client-Server*.

### 4.2.1 Arsitektur Deployment

| Komponen | Lingkungan yang Digunakan | Peran |
|---|---|---|
| Server Web | Apache / Nginx | Berfungsi sebagai *web server* yang menangani permintaan HTTP dari *client*. |
| Server Aplikasi | PHP (dengan *Framework* Laravel/CodeIgniter) | Menjalankan logika bisnis, mengelola sesi, dan memproses data. |
| Server Basis Data | MySQL / MariaDB | Menyimpan dan mengelola seluruh data Catat-Kerja (8 tabel yang telah dirancang). |
| Client | Peramban Web (*Web Browser*) | Antarmuka pengguna (UI/UX) diakses melalui peramban pada perangkat *desktop* atau *mobile*. |

### 4.2.2 Tahapan Deployment

1. Persiapan Lingkungan: Pemasangan sistem operasi server (misalnya Linux), Web Server (Apache), dan Database Server (MySQL).

2. Transfer Kode: Seluruh source code aplikasi (termasuk frontend dan backend) dipindahkan ke direktori web server.
3. Konfigurasi Basis Data: Pembuatan basis data kosong di MySQL, diikuti dengan eksekusi Query SQL (DDL) untuk membuat 8 tabel yang sudah dirancang.
4. Konfigurasi Aplikasi: Pengaturan file konfigurasi aplikasi (.env) untuk menghubungkan Server Aplikasi dengan Server Basis Data (database credentials).
5. Pengujian Akses: Memverifikasi bahwa aplikasi dapat diakses publik melalui domain atau IP address.

### 4.3 Pengujian Sistem

Pengujian sistem dilakukan untuk memastikan bahwa semua fungsi aplikasi bekerja sesuai dengan **Analisis Kebutuhan Fungsional (Bab 3)** dan bahwa sistem stabil dan bebas dari bug

#### 4.3.1 Metode Pengujian.

Metode pengujian yang digunakan adalah **Black Box Testing**. Metode ini berfokus pada fungsionalitas sistem dari sudut pandang pengguna (Freelancer) tanpa melihat struktur internal kode.

#### 4.3.2 Skenario Pengujian (Uji Fungsionalitas Kritis)

Pengujian akan berfokus pada fungsionalitas utama sistem (sesuai *Sequence Diagram* dan Kebutuhan Fungsional).

| No. | Fungsionalitas yang Diuji | Input / Skenario | Hasil yang Diharapkan | Status |
|---|---|---|---|---|
| 1. | Pelacakan Waktu (*Time Tracking*) | Pengguna klik START, jeda 10 menit, klik STOP. | Sistem mencatat 1 *record* di LOG_WAKTU dengan durasi_detik = 600. | [LULUS/GAGAL] |
| 2. | Otomatisasi Penagihan | Membuat *invoice* dari 5 log waktu | Sistem berhasil membuat 1 *record* di tabel INVOICE dan 5 | [LULUS/GAGAL] |

| | | | | |
|---|---|---|---|---|
| | | yang belum ditagih. | *record* di INVOICE_DETAIL. | |
| 3. | Alokasi Dana Proaktif | Mengubah status *invoice* berbayar menjadi 'Dibayar'. | Sistem secara otomatis mencatat 2-3 *record* di RIWAYAT_ALOKASI berdasarkan konfigurasi persentase pengguna. | [LULUS/GAGAL] |
| 4. | Login Pengguna | Input *email* yang benar dan *password* yang salah. | Sistem menolak akses dan menampilkan pesan error otentikasi. | [LULUS/GAGAL] |
| 5. | Manajemen Data Master | Membuat data Proyek baru tanpa memilih Klien. | Sistem menampilkan pesan validasi error (Klien wajib diisi). | [LULUS/GAGAL] |

# DAFTAR PUSTAKA