# Blueprint Kebutuhan Aplikasi — Draft v0.4 (sakuwuz)

Dokumen kerja hidup (living document). Kita kunci (sign‑off) setelah semua bagian terisi dan disetujui.

## Step 0 — Persiapan Lingkungan & Tooling (Web First)

**Stack yang dipilih:** Python **Django** + **Django Template** + **Tailwind/Bootstrap**, DB **PostgreSQL** (prod), **SQLite** (dev). Deploy: **Railway/Render/Heroku** (staging).

**Prasyarat di mesin dev:** - Python 3.11/3.12, pip, venv - Git & GitHub (repo privat) - SQLite (sudah bundle Python) & akses ke PostgreSQL (untuk staging/prod) - Node.js + npm (untuk build Tailwind; opsional kalau Bootstrap CDN)

**Python packages awal:** - Django - psycopg2-binary - python-dotenv atau django-environ - argon2-cffi (password hasher) - whitenoise (static di prod) - gunicorn (deploy WSGI) - django-axes (rate limit login, opsional)

**Front-end:** - Pilihan A: **Tailwind** via npm - Pilihan B: **Bootstrap** via CDN - Opsional: **HTMX + Alpine.js** untuk interaktivitas ringan

**Keamanan & konfigurasi:** - .env: SECRET\_KEY, DEBUG, DATABASE\_URL, ALLOWED\_HOSTS, TIME\_ZONE=Asia/Jakarta - Password hasher default: **Argon2** - Session & CSRF aktif; cookie Secure, HttpOnly di prod - Email reset password: **Mailtrap**/**Gmail SMTP** (staging)

**Struktur repositori (awal):**

sakuwuz/  
 manage.py  
 requirements.txt  
 .env.example  
 README.md  
 apps/  
 core/ # settings split, utils, middleware  
 accounts/ # auth, register/login/reset  
 budgets/ # model Budget  
 expenses/ # model Expense + kalender/detail  
 templates/  
 static/

## Step 1 — Latar Belakang & Tujuan

**Nama proyek (sementara):** Sakuwuz (Aplikasi Pencatatan Pengeluaran Bulanan/Harian)

**Masalah yang ingin diselesaikan (pain points):** - Sulit mengontrol pengeluaran harian agar tetap sesuai anggaran bulanan. - Ada alert otomatis ketika pengeluaran melebihi batas bulanan. - Ringkasan bulanan/tahunan biasanya manual (Excel, catatan kertas). - User ingin tracking pengeluaran + catatan deskripsi dalam satu aplikasi.

**Tujuan bisnis (SMART):** 1. Memudahkan pengguna mencatat pengeluaran harian ≤ 10 detik per input. 2. Membantu pengguna menjaga disiplin anggaran bulanan dengan alert otomatis (≥ 95% akurasi deteksi over‑budget). 3. Menyediakan ringkasan bulanan & tahunan dengan visual sederhana (≥ 90% pengguna aktif bulanan mengakses laporan).

**Jenis & cakupan aplikasi:** - Platform: Web (versi awal) → bisa dikembangkan ke Mobile. - Audiens: Individual / personal finance (B2C). - Multi-tenant: Tidak (per akun pribadi). - Wilayah operasi & bahasa: Indonesia (default) → bisa multi‑bahasa ke depannya.

**KPI 90 hari pertama:** - Adopsi: ≥ 50 user aktif mingguan. - Kinerja: input pengeluaran < 300 ms, laporan bulanan < 1 detik. - Kualitas: error rate < 1%, CSAT ≥ 4,5/5.

## Step 2 — Stakeholder & Persona

| Persona | Peran | Tujuan Utama | Hambatan Saat Ini | Frekuensi Pakai |
| --- | --- | --- | --- | --- |
| User Individu | Pemilik aplikasi | Catat pengeluaran harian, pantau sisa anggaran bulanan | Lupa input, repot hitung manual | Harian |
| Admin Sistem | Developer/Owner | Konfigurasi & support | Maintenance server, bug | Mingguan |

**Stakeholder RACI:** - Responsible: Developer - Accountable: Product Owner (kamu) - Consulted: User tester (teman/keluarga) - Informed: Komunitas/early adopter

## Step 3 — Ruang Lingkup (Scope)

**In-scope (V1/MVP):** - Set **budget bulanan** (mis. Rp 5.000.000). - **Input pengeluaran harian** (nominal + deskripsi teks). - **Kalender rekap & detail harian**: pilih tanggal → tampil **log/histori pengeluaran** hari itu. - **Perhitungan otomatis sisa anggaran** bulanan. - **Alert over-budget** (pop-up: “Tolong jangan terlalu boros…”). - **Laporan bulanan** (rekap harian + total) & **tahunan** (rekap bulanan + total).

**Out-of-scope (nanti):** - Integrasi dengan bank/e-wallet. - Fitur multi-user (sharing keluarga). - AI rekomendasi tabungan.

**Asumsi:** User input data manual.

**Batasan:** - Anggaran awal sederhana (server kecil, free tier DB). - Data sensitif (harus aman).

## Step 4 — Proses Bisnis

**AS-IS:** 1. User catat pengeluaran manual di kertas/Excel. 2. Akumulasi dihitung manual → sering salah. 3. Ada alert hanya jika hitung manual — tidak real-time.

**TO-BE (di aplikasi):** 1. User set budget bulanan (misal Rp 5.000.000). 2. User input pengeluaran harian (nominal + deskripsi). 3. Sistem mengurangi otomatis dari total budget bulanan & menghitung **total harian**. 4. User membuka **kalender** → klik **tanggal** → layar **Detail Harian** menampilkan **histori item** (waktu input, nominal, deskripsi) dan **subtotal hari itu**. 5. Jika sisa bulanan < 0 → muncul alert “Tolong jangan terlalu boros…”. 6. Sistem simpan riwayat & tampilkan rekap bulanan/tahunan.

**Aturan bisnis:** - Budget bulanan wajib diisi tiap bulan baru. - Nominal pengeluaran harus angka positif. - Deskripsi wajib minimal 1 kata. - **Satu tanggal bisa memiliki banyak item pengeluaran**; subtotal harian = sum(item pada tanggal tsb). - **Kalender default ke bulan & tahun saat ini**; user boleh navigasi ke bulan/tahun lain.

## Step 5 — Fitur & Prioritas (MoSCoW)

| Prioritas | Fitur | Deskripsi Singkat | Acceptance Criteria |
| --- | --- | --- | --- |
| Must | Set Budget Bulanan | Input nominal batas | Budget tersimpan, digunakan kalkulasi |
| Must | Input Pengeluaran Harian | Nominal + deskripsi | Data tersimpan, sisa budget berkurang |
| Must | Alert Over Budget | Notifikasi jika sisa < 0 | Pop‑up muncul, user tidak bisa abaikan |
| Should | Laporan Bulanan | Rekap harian → total bulanan | Ringkasan otomatis tersaji |
| Should | Laporan Tahunan | Rekap bulanan → total tahunan | Ringkasan otomatis tersaji |
| Could | Visual Chart | Pie/Bar chart pengeluaran | Tampil sesuai data |
| Won’t (V1) | Integrasi e-wallet | Import otomatis | Dijadwalkan V2 |

## Step 6 — Data & Integrasi

**Entitas utama:** | Entitas | Atribut Kunci | Relasi | |—|—|—| | User | id, nama, email, password | one-to-many → Budget, Expense | | Budget | id, bulan (1–12), tahun (YYYY), nominal\_max, sisa | many-to-one ke User; one-to-many → Expense | | Expense | id, tanggal (YYYY-MM-DD), waktu\_input, nominal, deskripsi, kategori (opsional) | many-to-one ke Budget |

**Indeks & validasi data:** - Index pada (user\_id, tahun, bulan, tanggal) untuk cepat filter per hari. - Constraint: nominal > 0; tanggal harus berada di bulan-tahun budget terkait.

**Integrasi eksternal (future):** Bank API, e-wallet.

## Step 7 — NFR (Non‑Functional)

* Availability ≥ 99% jam kerja.
* Performance: input < 300 ms, laporan < 1 detik.
* Security: hash password, enkripsi data at‑rest & in‑transit.
* Compliance: sesuai UU PDP 27/2022.

## Step 8 — UX Ringkas

* **Sitemap:** Dashboard → Input → Kalender → Detail Harian → Laporan → Setting.
* **User flow kunci:**
  + Set budget → Input harian → Lihat sisa budget (dashboard).
  + **Buka kalender → Pilih tanggal → Detail Harian** (list histori: jam, nominal, deskripsi → subtotal hari).
  + Buka laporan bulanan → lihat ringkasan per tanggal; buka laporan tahunan → ringkasan per bulan.
* **Wireframe (ringkas, deskripsi teks):**
  + *Dashboard*: kartu “Budget Bulan Ini” (Max, Terpakai, Sisa) + tombol “Input Pengeluaran” + tombol “Kalender”.
  + *Kalender Bulanan*: grid tanggal; badge total harian; klik tanggal → *Detail Harian*.
  + *Detail Harian*: header (Tanggal, Subtotal Hari), daftar item (waktu, nominal, deskripsi), tombol “Tambah Item”.

## Step 9 — UAT & Test Case (contoh awal)

| ID | Action | Expected Result | Test Result |
| --- | --- | --- | --- |
| TC-01 | Set budget Rp 5.000.000 | Budget tersimpan, tampil di dashboard | [Pass/Fail] |
| TC-02 | Input pengeluaran Rp 800.000 + “kebutuhan service motor” pada 2025-10-02 | Sisa budget berkurang Rp 800.000; item tampil di Detail Harian 02-10-2025 | [Pass/Fail] |
| TC-03 | Buka Kalender → klik 02-10-2025 | Muncul layar Detail Harian: daftar histori dengan subtotal hari | [Pass/Fail] |
| TC-04 | Input hingga melebihi budget | Pop-up alert “Tolong jangan terlalu boros…” muncul | [Pass/Fail] |
| TC-05 | Buka Laporan Bulanan Oktober 2025 | Rekap per tanggal + total bulanan tampil akurat | [Pass/Fail] |
| TC-06 | Buka Laporan Tahunan 2025 | Rekap per bulan + total tahunan tampil akurat | [Pass/Fail] |

**Acceptance Criteria tambahan (kalender & detail harian):** - Klik tanggal selalu memuat histori hari itu ≤ 300 ms untuk ≤ 200 item. - Subtotal hari = jumlah semua item pada tanggal tsb; konsisten dengan rekap bulanan. - Navigasi kalender antar bulan/tahun berjalan tanpa reload penuh.

## Step 10 — Risiko & Mitigasi

| Risiko | Dampak | Prob. | Mitigasi |
| --- | --- | --- | --- |
| User lupa input | Medium | Tinggi | Tambah reminder notifikasi |
| Data hilang (server down) | Tinggi | Rendah | Backup harian |
| Over‑budget alert tidak muncul | Tinggi | Rendah | Tambah unit test & QA |

## Step 11 — Ide Pengembangan Lanjutan

* Reminder via WhatsApp/Email untuk input harian.
* Fitur tabungan (sisa uang otomatis disarankan untuk ditabung).
* Kategori pengeluaran (makan, transport, hiburan).
* Integrasi ke Power BI untuk visualisasi lebih lanjut.

## Step 12 — Timeline Awal

| Milestone | Deliverable | Target |
| --- | --- | --- |
| M1 | Requirement sign‑off | Minggu 1 |
| M2 | Desain UX & DB | Minggu 2 |
| M3 | Prototype input & budget | Minggu 3 |
| M4 | Laporan bulanan & alert | Minggu 4 |
| M5 | UAT | Minggu 5 |
| M6 | Go‑Live V1 | Minggu 6 |