DOMPET SEHAT SEBAGAI PERSONAL FINANCE DENGAN ACCOUNT AGGREGATION

TUGAS AKHIR

Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Sarjana
Jurusan Teknik Informatika



Oleh :

Nama : Andwi Prima Valentin
NIM : 08 523 379

JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM INDONESIA
YOGYAKARTA
2013

LEMBAR PENGESAHAN PEMBIMBING

DOMPET SEHAT SEBAGAI PERSONAL FINANCE DENGAN ACCOUNT AGGREGATION

TUGAS AKHIR

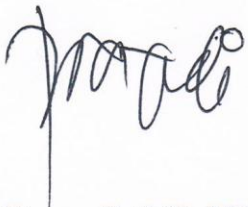
Oleh :

Nama : Andwi Prima Valentin

NIM : 08 523 379

Yogyakarta, Oktober 2013

Pembimbing



Yudi Prayudi, S.Si, M.Kom.

HALAMAN PENGESAHAN PENGUJI

DOMPET SEHAT SEBAGAI PERSONAL FINANCE DENGAN ACCOUNT AGGREGATION

TUGAS AKHIR

Oleh:

Nama : Andwi Prima Valentin

NIM : 08 523 379

Telah Dipertahankan di Depan Sidang Penguji sebagai Salah Satu Syarat untuk Memperoleh Gelar Sarjana Jurusan Teknik Informatika Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, Oktober 2013

Tim Penguji :

Ketua :

Yudi Prayudi, S.Si., M.Kom. : _____

Anggota 1 :

Arwan Ahmad Khoiruddin, S.Kom. : _____

Anggota 2 :

Lizda Iswari, S.T., M.Sc. : _____

Mengetahui,
Ketua Jurusan Teknik Informatika
Universitas Islam Indonesia

Yudi Prayudi, S.Si., M.Kom.

LEMBAR PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan dibawah ini

Nama : Andwi Prima Valentin
NIM : 08 523 379

Menyatakan bahwa seluruh komponen dan isi dalam Laporan Tugas Akhir dengan judul **"DOMPET SEHAT SEBAGAI PERSONAL FINANCE DENGAN ACCOUNT AGGREGATION"** yang diajukan untuk diuji pada tanggal 1 September 2013 adalah hasil karya saya sendiri.

Apabila di kemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya saya sendiri, maka saya siap menanggung resiko dan konsekuensi apapun.

Demikian pernyataan ini saya buat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, September 2013

Yang Membuat Pernyataan

(Andwi Prima Valentin)

HALAMAN PERSEMBAHAN

Karya nyata sederhana yang begitu banyak perjuangan ini, penulis persembahkan kepada:

Allah SWT

Hanya kepada Engkaulah kami menyembah dan hanya kepada Engkaulah kami memohon pertolongan. Engkaulah segala pelindung, penuntun jalanku, semoga karya ini menjadi suatu bentuk ibadah dan bermanfaat.

Kedua orang tuaku, Bapak dan Ibu yang selalu mendo'akan, membimbingku, memberikan semangat buatku, jerih payah mu mengajarkan aku arti dari sebuah perjuangan dan keikhlasan, kasih sayang mu mengajarkan aku arti dari sebuah ketulusan, terima kasih Bapak dan Ibu atas segala pengorbanan, kesabaran, dan ketulusanmu serta untuk Istriku tersayang terimakasih atas dukungan yang engkau berikan, banyak hal yang kita lalui dalam menjalani ini, terimkashi atas kesabaran dan pengorbananmu.

Semoga karya ini bermanfaat bagi banyak orang.

HALAMAN MOTTO

"It is well to be up before daybreak, forsuch habits contribute to health, wealt, and wisdom"

"Manfaatkanlah lima(keadaan), sebelum (datangnya) lima (yang lain): hidup sebelum matimu, sehat sebelum sakitmu, waktu luangmu sebelum waktu sempitmu, masa mudamu sebelum masa tuamu, dan kayamu sebelum miskinmu" (HR Hakim, Shahih, no. 7959)"

"Hidup adalah Sesuatu yang harus dipertanggungjawabkan"

"Ketika melangkah pelajarilah hal yang terjadi dari detik detik yang telah kau lewatkan sebelumnya."

KATA PENGANTAR



Assalamualaikum Wr. Wb.

Alhamdulillahi rabbil 'alamin. Segala puji bagi Allah SWT yang telah memberikan kesempatan bagi penulis untuk menyelesaikan Laporan Tugas Akhir ini. Sesungguhnya hanya atas izin dan kehendak-Nya penulis dapat menyelesaikan tugas akhir ini.

Tugas akhir ini merupakan syarat wajib di jurusan Teknik Informatika Universitas Islam Indonesia untuk memperoleh gelar sarjana. Untuk itu pada kesempatan baik ini, penulis ingin mengucapkan terima kasih kepada :

1. Allah SWT atas semua berkah dan rahmat-Nya sehingga Tugas Akhir ini dapat diselesaikan.
2. Bapak, Ibu, saudara, dan keluarga atas kasih sayang, segala limpahan doa dan dukungan.
3. Istriku tercinta Tri Endah Purnamasari yang telah memberi dukungan yang tak terhingga harganya. I love you.
4. Yang saya hormati Bapak Ir. Gumbolo HS., M.Sc selaku Dekan Fakultas Teknologi Industri Universitas Islam Indonesia.
5. Ketua Jurusan Teknik Informatika FTI UII dan juga sebagai Dosen Pembimbing tunggal dalam penyusunan Tugas Akhir ini Bapak Yudi Prayudi S.Si.,M.Kom.
6. Seluruh Dosen di Jurusan Teknik Informatika Universitas Islam Indonesia. Semoga ilmu yang telah diajarkan dapat menjadi amal, Amin.
7. Sahabat-sahabatku atas segala dukungan dan semangatnya.
8. Teman-teman angkatan 2008 (SNIPER) di Jurusan Teknik Informatika Universitas Islam Indonesia dan rekan-rekan seperjuangan.

9. Teman-teman kost yang selalu dan terus memberikan semangatnya, sukses untuk kalian semua.
10. Tidak lupa untuk teman-teman sepermainan, teman mulai dari TK, SD, SMP, SMA terima kasih banyak atas dukungan dan waktunya selama dalam pengerjaan skripsi ini.

Semua pihak yang tidak dapat saya sebutkan satu persatu. Penulis menyadari bahwa tugas akhir ini masih jauh dari sempurna. Untuk itu, penulis mengharapkan kritik dan saran yang bersifat membangun agar dapat berguna dikemudian hari. Penulis berharap semoga Tugas Akhir ini bermanfaat bagi semua orang dan bagi penulis sendiri. Amin.
Wassalamu'alaikum Wr.Wb.

Yogyakarta, September 2013

Penulis

ABSTRAK

Pada umumnya orang-orang senantiasa untuk selalu mencoba hal hal baru yang ada di sekitar mereka, sehingga pada akhirnya menimbulkan sifat boros, ditambah lagi dengan adanya kemajuan teknologi untuk mempermudah segala hal, temasuk untuk mengeluarkan uang dan berbelanja tanpa batas.

Banyak media yang bisa digunakan untuk melakukan hal tersebut, mulai dari toko online (*e-Commerse*), melalui media sosial (*Social Networking*) , *Smartphone* dan alat *Electronic Data Capture* (EDC) atau disebut *Automated teller machine* (ATM) mini sebuah bank yang digunakan untuk mengesek kartu debit atau kartu kredit.

Pada umumnya EDC digunakan di toko-toko, supermarket dan pusat perbelanjaan, sehingga kita tidak perlu membawa uang tunai untuk berbelanja, cukup menggunakan kartu debit atau kartu kredit. Dengan kemudahan tersebut akan menyebabkan tidak terkontrolnya pengeluaran secara personal. Penelitian ini menyajikan sebuah Platform Manajemen keuangan personal yang teragregasi dengan salah satu Bank di Indonesia.

Kata Kunci : *Technological Progress, Electronic Data Capture* (EDC), *Automated teller machine* (ATM) , Belanja, Applikasi, Aggregasi, BANK, Debit, Kredit, Saldo, *Trends, Budgets, Personal Finance.*

TAKARIR

Browsing internet	Mencari data atau suatu objek yang diinginkan dengan menggunakan teknologi internet
CodeIgniter	Sebuah web application framework yang bersifat open source yang digunakan untuk membangun aplikasi php dinamis
Account Aggregation	Sebuah aplikasi yang dapat teridentifikasi oleh URI dan memiliki interface yang didefiniskan, dideskripsikan, dan dimengerti oleh XML dan juga mendukung interaksi langsung dengan aplikasi yang lain dengan menggunakan pesan berbasis XML atau JSON melalui protokol internet
Personal Finance	Laporan keuangan secara personal
Transaction	Pertemuan antara dua belah pihak (Penjual dan Pembeli) yang saling menguntungkan dengan adanya data / bukti / dokumen pendukung yang dimasukkan kedalam jurnal setelah melalui pencatatan

DAFTAR ISI

HALAMAN JUDUL ……...…….. i

LEMBAR PENGESAHAN PEMBIMBING ……...…… ii

LEMBAR PENGESAHAN PENGUJI …… iii

LEMBAR PERNYATAAN KEASLIAN HASIL TUGAS AKHIR ……..…… iv

HALAMAN PERSEMBAHAN …… v

HALAMAN MOTTO …… vi

KATA PENGANTAR ……..... vii

ABSTRAK …….……...……..... ix

TAKARIR …….……..... x

DAFTAR ISI ……..… xi

DAFTAR GAMBAR ……...……..…… xiv

DAFTAR TABEL ……..…….…… xvi

BAB I PENDAHULUAN……..…….…… 1

1.1. Latar Belakang ……..... 1

1.2. Rumusan Masalah …….. 2

1.3. Batasan Masalah …… 3

1.4. Tujuan Penelitian …...…….…… 3

1.5. Manfaat Penelitian ……..…… 3

1.6. Metodologi Penelitian ……..……..... 3

1.6.1. Pengumpulan data ……..... 4

1.6.2. Pembuatan Perangkat Lunak …… 4

1.7. Sistematikan Penulisan …… 5

BAB II LANDASAN TEORI …… 7

2.1. Konsep Dasar Sistem Informasi ……..……... 7

2.2. Aplikasi ……...……..……. 8

2.3. Guidelines for Rapid Application Engineering (Grapple) ……. 9
2.4. Basisdata …… 10
2.5. Unified Modelling Langguange (UML) …… 12
2.5.1. *Use Case Diagram* ……..……..…… 12
2.5.2. *Class Diagram* ……...…… 14
2.6. PHP …….……..…… 18
2.6.1. Definisi…..…… 18
2.6.2. Konsep Kerja…..…….....…..... 18
2.7. MySQL …… 19
2.7.1. Definisi …..…….. 19
2.7.2. Konsep Kerja ……..….....… 19
2.8. *Framework CodeIgniter* ……... 21
2.8.1. Definisi ……..…… 21
2.8.2. Konsep Kerja ……..……..……..……..…….. 21
2.9. *HTML* ……..……..……..…… 23
2.10. *Rational Rose*…….....……... 23
2.11. *Personal Finance* ….....…… 24
2.12. *Transaction* ….....……..... 25
2.13. *Budgets* ….....…….…… 25
2.14. *Trends*….....…… 25
2.15. *Account Aggregation*….....…… 26

BAB III METODOLOGI PENELITIAN...…...……

3.1. *Requirement Gathering* ……..…… 26
3.1.1. Diagram *Use Case Level* Bisnis ……..…. 26
3.1.2. Diagram *High Level Class* ……...……..….. 27
3.2. Analisis ……..…… 31
3.1.1. Diagram *Use Case* ……..……..……..…… 31
3.1.2. Diagram *Class* ……...……..……..……. 34

3.3. Perancangan ……. 35
3.2.1. Basisdata ……..……..……..……. 35
3.2.2. Struktur Menu …… 44
3.2.3. Perancangan Antarmuka ……. 46

BAB IV HASIL DAN PEMBAHASAN …… 63
4.1. Perangkat Keras ..……. ……. 63
4.2. Perangkat Lunak …… 63
4.3. Tampilan Antarmuka ……..……..…… 64
4.3.1. Halaman Antarmuka *Visitor* ……..…….. 64
4.3.2. Halaman Antarmuka *User*…….…… 68
4.3.3. Halaman Antarmuka *Administrator* …….……. 81

BAB V PENUTUP ……..……..……. 87
5.1. Kesimpulan ……..…... 87
5.2. Saran ……..…. 87

DAFTAR PUSTAKA ……..... 88

DAFTAR GAMBAR

Gambar 2.1 Blok Diagram Sistem informasi ……..……. 15
Gambar 2.2 Diagram hubungan UML *view* dengan Diagramnya …… 16
Gambar 3.1 Diagram *Use Case Level* bisnis ……..…….. 17
Gambar 3.2 Diagram *High Level Class* ……..…..…..…..….. 24
Gambar 3.3 Pemodelan *Use Case* ……..…...…...…...…...…..…..... 25
Gambar 3.4 *Diagram Class Administrator* ……... 34
Gambar 3.5 *Diagram Class Visitor* ……..……..…….. 35
Gambar 3.6 *Diagram Class User* ……..……..... 35
Gambar 3.7 Relasi Antar Tabel ……...……...……...…… 37
Gambar 3.8 Perancangan Struktur Menu untuk *Visitor* …… 45
Gambar 3.9 Perancangan Struktur Menu untuk *User* …….. 45
Gambar 3.10 Perancangan Struktur Menu untuk *Administrator* ..……... 46
Gambar 3.11 Rancangan Halaman Utama *Visitor* ……...……... 46
Gambar 3.12 Rancangan Halaman *What's DompetSehat*.…… …… …… . 47
Gambar 3.13 Rancangan Halaman *How it works* ...…… ...…… ...…..... 47
Gambar 3.14 Rancangan Halaman *Take a tour* ...…… ...…… ...…..... 48
Gambar 3.15 Rancangan Halaman *Updates* ...…… ...…… ...…..... 48
Gambar 3.16 Rancangan Halaman *Sign Up* ...…… ...…… ...…..... 49
Gambar 3.17 Rancangan Halaman *Login Admin* ...…… ...…… ...…..... 49
Gambar 3.18 Rancangan Halaman *Dashboard Admin* ...…… ...…… .. 50
Gambar 3.19 Rancangan Halaman *Slide Home* ...…… ...…….....…..... 50
Gambar 3.20 Rancangan Halaman Page Statis ...…… ...…… ... … 51
Gambar 3.21 Rancangan Halaman *Article*...…… ...…… ...…..... ...…... 51
Gambar 3.22 Rancangan Halaman *Edit Article*...…… ...…… ...…..... .. 52
Gambar 3.23 Rancangan Halaman *Template Email* ...…… ...…… ... 52
Gambar 3.24 Rancangan Halaman *Feedback* ...…… ...…… ...…..... 53

Gambar 3.25 Rancangan Halaman Kategori 53
Gambar 3.26 Rancangan Halaman *Add* Kategori 54
Gambar 3.27 Rancangan Halaman Member 54
Gambar 3.28 Rancangan Halaman Ubah *Password* 55
Gambar 3.29 Rancangan Halaman *Dashboard User* 55
Gambar 3.30 Rancangan Halaman *Transactions* 56
Gambar 3.31 Rancangan Halaman *Budgets* 57
Gambar 3.32 Rancangan Halaman *Trends* 57
Gambar 3.33 Rancangan Halaman *Trends by overtime* 58
Gambar 3.34 Rancangan Halaman *Trends by category* 58
Gambar 3.35 Rancangan Halaman Notifikasi 59
Gambar 3.36 Rancangan Halaman Auto Kategori 59
Gambar 3.37 Rancangan Halaman *Auto Translate* 60
Gambar 3.38 Rancangan Halaman Rekening 60
Gambar 3.39 Rancangan Halaman *Transfer* Rekening 61
Gambar 3.40 Rancangan Halaman *Connect your Bank* 61
Gambar 3.41 Rancangan Halaman *Profile* 62
Gambar 3.41 Rancangan Halaman *Update Password* 62
Gambar 4.1 Halaman *Homepage* 64
Gambar 4.2 Halaman *What's Dompetsehat* 65
Gambar 4.3 Halaman *How it works* 66
Gambar 4.4 Halaman *Take a tour* 67
Gambar 4.5 Halaman *Updates dan Blog*..... 67
Gambar 4.6 Halaman *Sign Up*..... 68
Gambar 4.7 Halaman *Overview* 69
Gambar 4.8 Halaman *Transaction* 70
Gambar 4. 9 Halaman *Manage Category* 70
Gambar 4.10 Halaman *Budgets Overview* 71
Gambar 4.11 Halaman *Income Budgets* 71

Gambar 4.12 Halaman *Spending Budgets* 72
Gambar 4.13 Halaman *Manage Budgets* 72
Gambar 4.14 Halaman *Trends* 73
Gambar 4.15 Halaman *Spending by Overtime* 74
Gambar 4.16 Halaman *Spending by Category* 74
Gambar 4.17 Halaman *Net Income* 75
Gambar 4.18 Halaman Notifikasi 75
Gambar 4.19 Halaman *All Notification* 76
Gambar 4.20 Halaman *Transfer History* 76
Gambar 4.21 Halaman *Add Transfer* 77
Gambar 4.22 Halaman Rekening 77
Gambar 4.23 Halaman Tambah Rekening 78
Gambar 4.24 Halaman *Connect Bank account* 78
Gambar 4.25 Halaman *Setting Auto Category* 79
Gambar 4.26 Halaman *Setting Auto Category* 79
Gambar 4.27 Halaman *Setting Auto Translate* 80
Gambar 4.28 Halaman *Profile User* 80
Gambar 4.29 Halaman *Update Password* 81
Gambar 4.30 Halaman *Login Admin* 81
Gambar 4.31 Halaman *Dashboard Administrator* 82
Gambar 4.32 Halaman *Slidehome* 82
Gambar 4.33 Halaman *Page Statis* 83
Gambar 4.32 Halaman *Article* 83
Gambar 4.35 Halaman *Template* 84
Gambar 4.36 Halaman *Feedback* 84
Gambar 4.37 Halaman Kategori 85
Gambar 4.38 Halaman *member* ....

DAFTAR TABEL

Tabel 2.1 Notasi pada Diagram *Use Case* ……..... 33
Tabel 3.1 Tabel admin ……... 36
Tabel 3.3 Tabel bank ……..……..……... 36
Tabel 3.4 Tabel feedback ……...…… 36
Tabel 3.5 Tabel jenis_tagihan ……...……..…… 36
Tabel 3.6 Tabel kategorinew ……..……... 36
Tabel 3.7 Tabel kota ……..…… 37
Tabel 3.8 Tabel master_setting ……..……... 37
Tabel 3.9 Tabel news ……..……..……..... 37
Tabel 3.10 Tabel otomatis_translate ……..……..... 37
Tabel 3.11 Tabel pagestatis ……..……..... 38
Tabel 3.12 Tabel slidehome ……..……..... 38
Tabel 3.13 Tabel template_email ……..……..... 38
Tabel 3.12 Tabel template_notifikasi……..……... 39
Tabel 3.13 Tabel user ……..……..... 39
Tabel 3.19 Tabel user_bank ……..……..... 40
Tabel 3.20 Tabel user_budget ……..……..... 40
Tabel 3.21 Tabel user_cashflow ……..……..... 40
Tabel 3.17 Tabel user_kategori ……..……..... 41
Tabel 3.18 Tabel user_notifikasi ……..……..... 41
Tabel 3.14 Tabel user_otomatis_kategori ……..……... 42
Tabel 3.15 Tabel user_otomatis_sensor ……..……... 42
Tabel 3.22 Tabel user_rekening ……..……..... 43
Tabel 3.23 Tabel user_setting ……..……..... 43

BAB I

PENDAHULUAN

1.1 Latar Belakang

Indonesia adalah salah satu negera dengan jumlah penduduk terbanyak di dunia. Badan Kependudukan dan Keluarga Berencana Nasional (BKKBN) menyampaikan tahun 2013 penduduk Indonesia diperkirakan akan mencapai 250 juta jiwa dengan laju pertumbuhan penduduk 1,49 persen per tahun. Hasil sensus penduduk tahun 2012, Indonesia berada pada posisi keempat dunia dengan jumlah penduduk terbanyak, setelah Cina, India dan Amerika Serikat. Setiap tahun terjadi penambahan jumlah penduduk Indonesia sekitar 4 juta jiwa (www.bps.go.id). Pertumbuhan laju penduduk berdampak pada meningkatnya pertumbuhan ekonomi nasional Indonesia. Pertumbuhan ekonomi secara kasar bisa dilihat dari transformasi tingkat perekonomian masyarakat, masyarakat tingkat bawah bertransformasi menjadi masyarakat tingkat menengah, begitu juga dengan masyarakat tingkat menengah bertransformasi ke tingkat atas. Transformasi tersebut berpengaruh terhadap tingkat pendapatan dan pengeluaran personal, keluarga, ataupun institusi yang semakin besar.

Selain laju pertumbuhan penduduk, yang mempengaruhi pertumbuhan ekonomi Indonesia adalah Kemajuan teknologi (*technological progress*). Kemajuan Teknologi merupakan sumber pertumbuhan ekonomi yang paling penting. Dalam pengertian yang paling sederhana, kemajuan teknologi terjadi karena ditemukannya cara baru atau perbaikan atas cara-cara lama dalam menangani pekerjaan-pekerjaan tradisional seperti kegiatan menulis digantikan dengan kegiatan mengetik komputer. Kemajuan teknologi semakin berkembang pesat, semua hal dilakukan dengan mudah dan cepat, salah satunya untuk mengeluarkan uang dan berbelanja tanpa batas, banyak media yang bisa digunakan untuk melakukan hal tersebut, mulai dari Toko Online (*e-Commerse*), melalui media social (*Social Networking*), *Smartphone*, dan alat *Electronic Data Capture* (*EDC*) atau disebut juga *Automated Teller Machine (ATM)* mini sebuah bank yang digunakan untuk mengesek kartu debit atau kredit, pada umumnya alat

ini digunakan di toko-toko, supermarket dan pusat perbelanjaan, sehingga tidak perlu membawa uang tunai untuk berbelanja, hanya dengan mengesek kartu debit yang berisi saldo. Dengan kemudahan tersebut akan menyebabkan tidak terkontrolnya pengeluaran secara personal. Oleh karena itu, adanya sebuah Platform Manajemen Keuangan yang teragregasi dengan Bank merupakan solusi pintar untuk masyarakat Indonesia.

Dalam Penelitian ini, akan dibuat platform bernama "Dompet Sehat". Dompet Sehat adalah sebuah platform *Personal Finance* bebasis *website* yang dapat mengagregasi salah satu bank terbaik di Indonesia, hasil agregasi tersebut akan menampilkan data-data transaksi terakhir yang dilakukan oleh akun bank yang besangkutan, kemudian transaksi tersebut diolah oleh sistem Dompet Sehat melalui perantara *user*, dan mencatat aktifitas keuangan (pemasukan dan pengeluaran) harian *user*, kemudian secara otomatis di translasi ke bahasa yang mudah dipahami, serta secara otomatis menempatkannya di kategori – kategori yang diinginkan, selain itu Dompet Sehat juga dilengkapi dengan pencatatan rekening secara manual, dengan *input* data dari *user*.

Pengguna Dompet Sehat adalah pengguna yang menginginkan perencanaan keuangan simpel, nyaman dan mudah digunakan. Dengan menggunakan *platform* Dompet Sehat maka pengguna akan dapat mengatur keuangan pribadinya dengan lebih rapi, mudah dan nyaman.

1.2 Rumusan Masalah

Masalah yang dirumuskan pada penelitian ini adalah sebagai berikut :

a. Bagaimana membuat sebuah *platform* manajemen keuangan personal yang dapat berguna bagi masyarakat Indonesia.
b. Bagaimana menerapkan konsep *account aggregation* pada Dompetsehat.
c. Bagaimana menerapkan translasi bahasa bank ke bahasa natural yang mudah pahami dan mengkategorikan data transaksi secara automatis.

1.3 Batasan Masalah

Penelitian ini dibatasi pada beberapa masalah:

a. Sistem yang dibuat fokus pada manajemen keuangan personal yang teragregasi dengan salah satu bank terbaik di Indonesia yaitu Bank Central Asia (BCA).
b. DompetSehat dibangun dengan bahasa pemograman *PHP*, dan meggunakan teknik *cUrl* untuk agregasi kepada Bank serta memakai *code igniter* sebagai *framework* dan *MySQL* sebagai basisdata.

1.4 Tujuan Penelitian

Tujuan penelitian dalam tugas akhir ini adalah :

a. Merancang dan mengimplementasikan suatu program aplikasi manajemen keuangan personal yang teragregasi dengan Bank BCA.
b. Membangun sebuah *platform* manajemen keuangan personal yang akan berguna untuk seluruh rakyat Indonesia.

1.5 Manfaat Penelitian

Manfaat dari penelitian ini adalah :

a. Memberikan pengetahuan pada banyak orang bahwa manajemen keuangan itu sangat penting untuk kehidupan yang lebih baik.
b. Membantu dan mempermudah *user* mencatat aktifitas keuangan (pemasukan dan pengeluaran) harian *user*, secara otomatis atau manual.

1.6 Metodologi Penelitian

Metodologi yang digunakan dalam penelitian ini yaitu

1.6.1 Pengumpulan Data

Metodologi pengumpulan data adalah metodologi yang digunakan untuk pengumpulan data yang diperlukan dalam penelitian. Metodologi tersebut meliputi:

a. Studi pustaka

   Pada metode ini pencarian data atau melakukan pengumpulan data yang dibutuhkan dilakukan dengan cara mencari atau mengumpulkan data-data melalui buku-buku atau *literature* yang berhubungan dengan masalah yang dibahas yaitu tentang applikasi keuangan personal berbasis *website* dengan *account aggregation* menggunakan *framework codeigniter*.

b. Wawancara

   Pada metode ini pencarian data dilakukan dengan cara melakukan wawancara ke beberapa User dan perencana keuangan yang dijadikan sebagai *sample*.

c. Kuisioner

   Pada metode ini pencarian data dilakukan dengan cara melakukan menyebarkan kuisioner secara online untuk mendapatkan data-data masukan dari *user*.

1.6.2 Pembuatan Perangkat Lunak

Setelah data-data yang dibutuhkan terkumpul, maka dilanjutkan dengan tahap-tahap sebagai berikut:

a. Analisis Kebutuhan Sistem: UML (*Unified Modeling Language*).
b. Perancangan Sistem: sistem *interface*, relasi *database*, rancangan *website dan account aggregation*.
c. Pengujian Sistem: cek *error*, pengujian skenario sistem.

1.7 Sistematika Penulisan

Sistematika penulisan laporan tugas akhir ini adalah sebagai berikut:

BAB I PENDAHULUAN

Pada bab ini berisi tentang latar belakang permasalahan, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metodologi penelitian, dan sistematika penulisan laporan tugas akhir.

BAB II LANDASAN TEORI

Bab ini membahas dasar teori yang digunakan untuk merancang dan membangun sistem. Teori yang dibahas antara lain teori tentang aplikasi keuangan berbasis *website*, PHP, MySql, *framework codeigniter*, UML (*Unified Modeling Langguage*) dan *account aggregation*.

BAB III METODOLOGI

Bab ini membahas langkah-langkah penyelesaian masalah dalam penelitian ini.

BAB IV HASIL DAN PEMBAHASAN

Bab ini menjelaskan tentang hasil pengujian dan analisis kinerja sistem. Pengujian perangkat lunak yang dijelaskan dalam bentuk antar muka. Analisis hasil pengujian yang memuat dokumentasi pengujian perangkat lunak terhadap kebenaran dan kesesuaiannya dengan kebutuhan perangkat lunak yang dituliskan dalam bagian sebelumnya yang kemudian dianalisis. Hasil dari pengujian terhadap perangkat lunak berisi tentang kemungkinan kesalahan yang muncul ketika digunakan oleh *user* dan penampilan pesan kesalahan.

BAB V KESIMPULAN DAN SARAN

Bab ini merupakan bab terakhir yang menguraikan kesimpulan selama proses dilakukan sehingga terwujudnya sebuah *Platform Personal Finance* dengan *Account Aggregation* yang dapat digunakan oleh seluruh masyarakat Indonesia dan saran-saran yang dapat digunakan untuk pengembangan sistem berikutnya.

BAB II
LANDASAN TEORI

2.1 Sistem Informasi

Sistem Informasi dapat di definisikan sebagai suatu sistem di dalam suatu organisasi yang merupakan kombinasi dari orang-orang, fasilitas, teknologi, media, prosedur-prosedur dan pengendalian yang di tujukan untuk mendapatkan jalur komunikasi penting, memproses tipe-tipe transaksi tertentu, memberi sinyal kepada manajemen dan yang lainnya terhadap kejadian-kejadian internal dan eksternal yang penting dan menyediakan suatu dasar informasi untuk pengambilan keputusan (Hartono, 1999).

Sistem Informasi mempunyai enam buah komponen atau disebut juga dengan blok bangunan (*building block*), dan keenam komponen ini harus ada bersama-sama dan membentuk satu kesatuan. Jika satu atau lebih komponen tersebut tidak ada, maka sistem informasi tidak akan dapat melakukan fungsinya, yaitu pengolahan data dan tidak dapat mencapai tujuannya, yaitu menghasilkan informasi yang relevan, tepat waktu dan akurat. Komponen-komponen tersebut adalah sebagai berikut :

a. Blok Masukan (*Input Block*)

   Input merupakan data yang masuk ke dalam sistem informasi.

b. Blok Model (*Model Block*)

   Kombinasi prosedur, logika, dan model matematik yang akan memanipulasi data input dan data yang tersimpan di basis data dengan cara yang sudah ditentukan untuk meghasilkan keluaran yang diinginkan.

c. Blok Keluaran (*Output Block*)

   Keluaran yang merupakan informasi yang berkualitas dan dokumentasi yang berguna untuk semua tingkatan manajemen serta semua pemakai sistem.

d. Blok Teknologi (*Technology Block*)

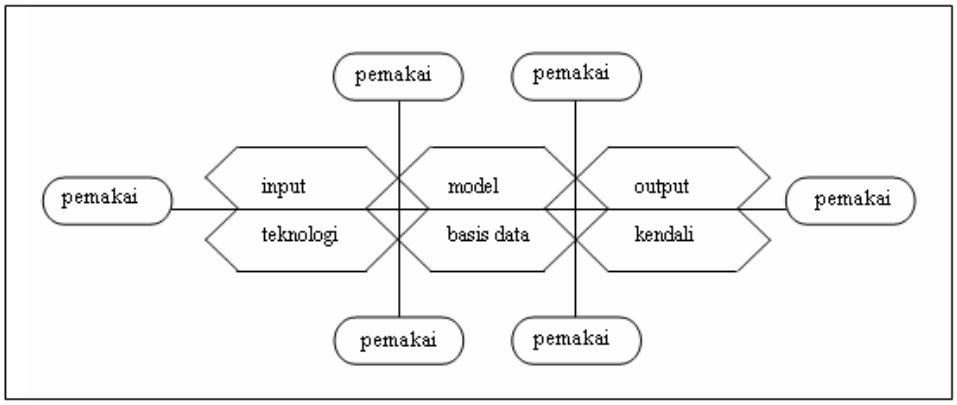
Teknologi merupakan kotak alat (*toolbox*) dalam sistem informasi. Teknologi digunakan untuk menerima input, menjalankan model, menyimpan dan mengakses data, menghasilkan dan mengirimkan keluaran dan membantupengendalian dari sistem secara menyeluruh.

e. Blok Basis Data (*Database Block*)

Merupakan kumpulan dari data yang saling berhubungan satu dengan yang lainnya, tersimpan di perangkat keras komputer dan digunakan perangkat lunak untuk memanipulasinya.

f. Blok Kendali (*Control Block*)

Beberapa pengendalian yang dirancang secara khusus untuk menanggulangi gangguan-gangguan terhadap sistem.



Gambar 2.1 Blok Diagram Sistem Informasi

2.2 Aplikasi

Aplikasi adalah penggunaan atau penerapan suatu konsep yang menjadi pokok pembahasan, salah satu implementasinya adalah pada suatu program yang dibuat untuk ditujukan melakukan tugas khusus. Tugas khusus tersebut tidak terlepas dari jalannya suatu sistem yang telah dirancang untuk menyelesaikan tugas yang diinginkan oleh user untuk memperoleh keluaran berupa informasi (www.wikipedia.org).

2.3 *Guidelines for Rapid Application Engineering* (*GRAPPLE*)

GRAPPLE merupakan salah satu dari metodologi pengembangan sistem RAD (Rapid Application Development) dengan kerangka pengembangan GRAPPLE (Guidelines for Rapid Application Engineering) yang merupakan aksi-aksi yang dilakukan dalam tiap segmen, metode ini digunakan untuk membangun sistem berorientasi objek dalam waktu yang singkat tanpa mengurangi kualitas sistem yang dibangun. *GRAPPLE* merupakan sebuah pemodelan proses dalam pengembangan *software* yang menekankan pada aksi-aksi yang dilakukan pada sejumlah tahapan (Schmuller, 1999).

Tahapan yang digunakan dalam *GRAPPLE* mencakup analisis kebutuhan sistem, pengembangan model dan diagram, pembuatan kode hingga tahap instalasi dan evaluasi. Metode ini memiliki 5 segmen/ langkah-langkah yang terdiri dari :

a. *Requirement Gathering*

   Tahap ini adalah tahap dimana dilakukan analisis terhadap masalah, fungsi dan komponen produk yang akan dibuat (*system requirements*). Tahap ini merupakan tahap yang sangat penting, karena tahap lain tidak dapat dibuat sesuai dengan yang diinginkan jika tidak memahami produk yang akan dibuat.

b. *Analysis*

   Tahap *analysis* adalah tahap pengembangan model dari data dan informasi yang diperoleh dari *requirement gathering*. Model merupakan bentuk transisi dari informasi dasar ke dalam bentuk model dan diagram.

c. *Design*

   Merupakan tahap implementasi dan perancangan dari model serta diagram yang telah dianalisis. Dalam tahap ini akan dikembangkan sejumlah objek diagram dengan fungsi, interaksi dan operasi tertentu. Diagram-diagram tersebut antara lain akan menunjukkan proses dan

aktifitas pada sistem, rancangan data dan penyimpanan data, serta rancangan antarmuka.

d. *Development*

Merupakan tahap penerapan model dan diagram yang telah terbentuk, antara lain dengan melakukan pengembangan *source code*, pengecekan dan *test code*, serta pembuatan *user interface*.

e. *Deployment*

Deployment merupakan tahap terakhir yang dilakukan setelah *development*. Sistem yang terbentuk akan diintregrasikan dengan *hardware* maupun dengan sistem operasi yang digunakan

2.4 Basis Data

Secara konseptual, data adalah deskripsi tentang benda, kejadian, aktifitas, dan transaksi, yang tidak mempunyai makna atau tidak berpengaruh secara langsung kepada pemakai (Kadir, 2002).

Basis data adalah kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa pengulangan (redudansi) yang tidak perlu, untuk memenuhi berbagai kebutuhan (Fathansyah, 2002).

Untuk mengolah basis data diperlukan perangkat lunak pengolahan basis data. Perangkat lunak yang dapat digunakan untuk mengelola basis data disebut *Database Management System* (DBMS). DBMS adalah perangkat lunak sistem yang memungkinkan para pemakai membuat, memelihara, mengontrol dan mengakses basis data dengan cara yang praktis dan efisien. DBMS dapat digunakan untuk mengakomodasikan berbagai macam pemakai yang memiliki kebutuhan akses yang berbeda-beda. Operasi-operasi dasar untuk manajemen data yang dapat dilakukan dengan *database* meliputi :

a. *Create database* (pembuatan basis data baru).
b. *Drop database* (penghapusan basis data).
c. *Create table* (pembuatan file/tabel baru).
d. *Drop Table* (penghapusan file).
e. *Insert* (penambahan/pengisian data baru pada file).

f. *Retrieve/search* (pengambilan data dari sebuah file).
g. *Update* (pengubahan data dari sebuah file).
h. *Delete* (penghapusan data dari sebuah file).
i. *Select* (perintah pengambilan data/akses data).
j. *Alter* (perintah untuk pengubahan data/file).

DBMS merupakan perantara bagi pemakai dengan basis data dalam *disk*. Cara berinteraksi antara pemakai dengan basis data tersebut di atur dalam suatu bahasa khusus yang ditetapkan oleh perusahaan pembuat DBMS. Bahasa tersebut dapat kita sebut sebagai Bahasa Basis Data yang terdiri atas sejumlah s*tatement* yang diformulasikan dan dapat diberikan *user* dan dikenali/diproses oleh DBMS untuk melakukan aksi tertentu. Sebuah bahasa basis data biasanya dapat dipilih ke dalam tiga bentuk, yaitu :

a. *Data Definition Language* (DDL)

   Struktur/skema basis data yang menggambarkan/mewakili desain basis data secara keseluruhan dispesifikasikan dengan bahasa khusus yang disebut *Data Definition Language* (DDL). Hasil dari kompilasi perintah DDL adalah kumpulan tabel yang disimpan dalam file khusus yang disebut kamus data.

b. *Data Manipulation Language* (DML)

   Merupakan bentuk Bahasa Basis Data yang berguna untuk melakukan manipulasi dan pengambilan data pada suatu basis data. DML merupakan bahas aynag bertujuan memudahkan pemakai untuk mengakses data sebagaimana dipresentasikan oleh model data.

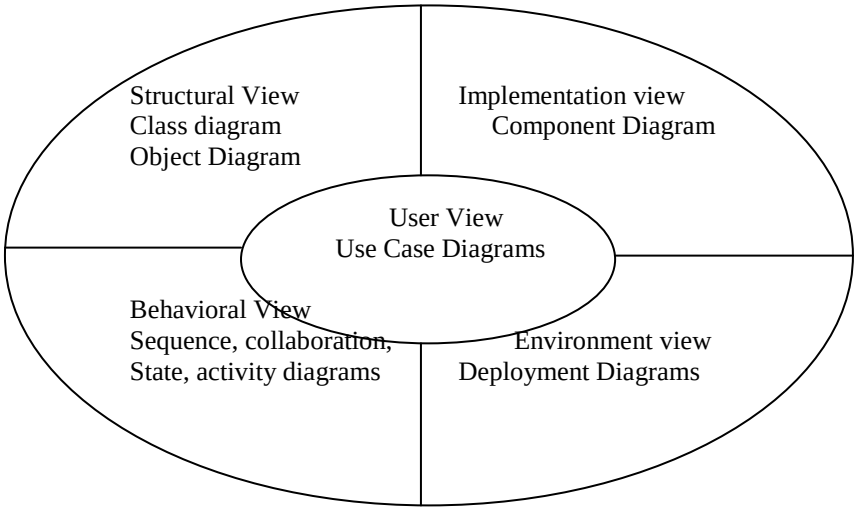
c. *Data Control Language* (DCL)

   Merupakan perintah yang akan membantu mengontrol keamanan setiap *database* atau sebagian isi dari *database* dengan membuat hak akses tertentu bagi setiap user.

2.5 *Unified Modelling Language* (UML)

UML adalah bahasa pemodelan visual yang memungkinkan pengembang sistem untuk membuat *blueprint* berisi visi standar yang mudah untuk dimengerti dan mekanisme efektif untuk berbagi dan mengkomunikasikan visi mereka kepada orang lain (Schmuller, 2004).

UML terdiri atas sejumlah elemen grafis yang membentuk berbagai diagram. Tujuan dibuatnya berbagai diagram tersebut adalah untuk mempresentasikan beragam sudut pandang (*view*) terhadap suatu sistem. Berikut adalah gambar hubungan UML *view* dengan diagramnya.



Gambar 2.2 Diagram hubungan UML *view* dengan diagramnya.

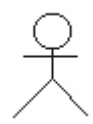
Model UML menggambarkan bagaimana sistem akan dibangun, bukan bagaimana sistem diimplementasikan.

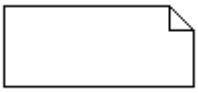
2.5.1 *Use-Case Diagram*

Diagram *use case* adalah diagram yang memperlihatkan hubungan-hubungan yang terjadi antara aktor-aktor dengan *use case-use case* dalam sistem (Nugroho, 2005). Salah satu manfaat dari *use* case adalah untuk komunikasi. Calon pengguna sistem/ perangkat lunak dapat mengamati diagram *use* case untuk mendapatkan pemahaman yang utuh tentang sistem yang akan dikembangkan.

Dengan melihat *use* case maka dapat diketahui siapa saja (atau apa saja) yang akan berinteraksi dengan sistem. Dengan melihat kombinasi-kombinasi sejumlah aktor dan *use case*, maka akan diketahui secara jelas ruang lingkup dari proyek pengembangan sistem / perangkat lunak itu. Dibawah ini adalah tabel notasi dalam *Use Case diagram*.

Tabel 2.1 Notasi pada Diagram *Use Case*

Notasi	Nama Notasi	Fungsi
 aktor	*Actor*	Menambah actor baru pada diagram. Aktor yang dimaksud disini dapat berupa orang, perangkat keras atau system lain yang berinteraksi langsung dengan aplikasi perangkat lunak. Aktor memberikan gambaran tentang apa yang harus diperbuat oleh aplikasi perangkat lunak. Aktor berinteraksi dengan *use case* dalam sistem, tetapi tidak memiliki control terhadapnya.
 UseCase	*Use Case*	Menambah *Use Case* pada diagram. *Use Case* merupakan notasi dalam UML untuk menjelaskan urutan kegiatan yang dilakukan aktor dan sistem untuk mencapai suatu tujuan tertentu. *Use Case* hanya menjelaskan apa yang dilakukan oleh aktor dan system, bukan bagaimana aktor dan sistem melakukan kegiatan tersebut.

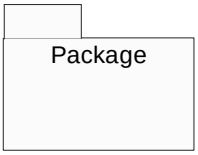
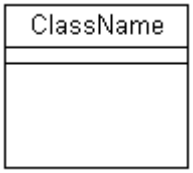
	*Dependency*	Merupakan relasi yang memberikan penjelasan bahwa perubahan yang terjadi pada salah satu elemen memberi pengaruh pada elemen lain.
	*Association*	Menggambarkan relasi antara aktor dan *Use Case*.
	*Generalization*	Menggambarkan relasi lanjut antar *Use Case* atau menggambarkan struktur pewarisan antar aktor.
	*Note*	*Note* digunakan untuk memberikan keterangan dan komentar tambahan dari suatu elemen sehingga dapat langsung terlampir dalam model. *Note* dapat ditempelkan ke semua elemen notasi yang lain.
	Anchor to note	Melekatkan catatan pada *Use Case* atau aktor tertentu dalam diagram.

2.5.2 *Class Diagram*

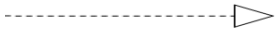
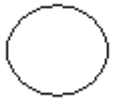
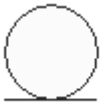
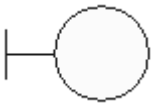
Class diagram adalah diagram yang digunakan untuk menampilkan beberapa kelas serta paket-paket yang ada dalam sistem / perangkat lunak yang sedang dikembangkan (Nugroho, 2005). Diagram kelas memberi gambaran statis tentang sistem/perangkat lunak dan relasi-relasi yang ada di dalamnya. Diagram kelas adalah kakas perancangan utama bagi tim pengembang. Diagram-diagram kelas akan membantu para pengembang untuk melihat dan merencanakan struktur dari sistem/ perangkat lunak sebelum kode pemrograman dituliskan. Hal ini akan membantu pengembang untuk memastikan bahwa sistem/ perangkat lunak

dianalisis dan dirancang dengan baik dari awal. Dibawah ini adalah tabel notasi dalam *class diagram*.

Tabel 2.2 Notasi pada *Class Diagram*

Notasi	Nama Notasi	Fungsi
	*Note*	*Note* digunakan untuk memberikan keterangan dan komentar tambahan dari suatu elemen sehingga dapat langsung terlampir dalam model. *Note* dapat ditempelkan ke semua elemen notasi yang lain.
	*Package*	Merupakan kontainer atau wadah konseptual yang digunakan untuk mengelompokkan elemen-elemen dari sistem yang sedang dibangun. Dari hal ini dapat dibuat model yang lebih sederhana. Tujuan dari *package* ini untuk mempermudah penglihatan (*visibility*) dari model yang sedang dibangun.
	*Class*	Kelas berupa kumpulan obyek yang memiliki atribut dan operasi. Kelas digunakan untuk mengimplementasikan *interface* da mengabstraksikan elemen-elemen dari sistem yang sedang dibangun. Kelas

		dapat pula digunakan untuk mempresentasikan perangkat lunak maupun perangkat keras, baik konsep maupun benda nyata.
	*Dependency*	Merupakan relasi yang memberikan penjelasan bahwa perubahan yang terjadi pada salah satu elemen memberi pengaruh pada elemen lain.
	*Association*	*Association* menggambarkan hubungan antar kelas. Dalam suatu komunikasi *unidirectoral association*, akhir dari arah panah mengindikasikan siapa atau apa yang menerima komunikasi.
	*Generalization*	Generalization menunjukkan hubungan antar elemen yang lebih umum ke elemen yang lebih khusus atau spesifik. Dengan generalization, kelas yang lebih spesifik (subclass) akan menurunkan atribut dan operasi dari kelas yang lebih umum (superclass).
	Anchor to note	Melekatkan catatan pada Use Case atau aktor tertentu dalam diagram.

	*Realization*	*Realization* menunjukkan bahwa hubungan elemen yang ada di bagian tanpa panah akan merealisasikan apa yang dinyatakan oleh elemen yang ada di bagian dengan panah.
 Interface	*Interface*	Interface merupakan kumpulan operasi tanpa implementasi dari suatu kelas. Implementasi operasi dalam interface dijabarkan oleh operasi dalam kelas. Oleh karena itu keberadaan interface selalu disertai oleh kelas yang mengimplementasikan operasinya.
	*Entity*	*Entity* biasanya berupa kata benda yang nantinya digunakan untuk menjalankan tanggungjawab sistem.
	*Boundary*	*Boundary* menggambarkan kelas untuk menangani komunikasi antara lingkungan luar sistem dengan lingkungan dalam sistem.
	*Control*	*Control* menggambarkan kelas yang bertanggung-jawab menjalankan aliran kejadian dalam use case secara dinamis.

2.6 PHP

PHP atau PHP *Hypertext Preprocessor* adalah sebuah bahasa pemrograman web berbasis server (*server-side*) yang mampu mem-*parsing* kode php dari kode web dengan ekstensi .php, sehingga menghasilkan tampilan website yang dinamis d sisi *client* (*browser*). Dengan PHP halaman HTML (*hypertext markup language*) menjadi lebih *powerful* dan dapat dipakai sebagai aplikasi lengkap, misalnya untuk beragam aplikasi *cloud computing.*(Edy Winarno , Ali Zaki, SmiDev Community, 2010)

2.6.1 Pengenalan PHP

PHP awalnya dikembangkan oleh seorang programmer bernama Rasmus Lerdoft pada tahun 1995, namun semenjak itu dikembangkan oleh kelompok independen yang disebut Group PHP dan kelompok ini juga yang mendefinisikan standar *de facto* untuk PHP karena tidak ada spesifikasi formal. Saat ini pengembangannya dipimpin oleh Andi Gutmans dan Zeev Suraski.

Yang menyebabkan PHP banyak dipakai orang banyak adalah karena PHP adalah perangkat lunak bebas (*open source*) yang dirilis dibawah lisensi PHP. Artinya, untuk menggunakan bahasa pemrograman ini gratis, bebas, dan terbuka. Setiap kode PHP akan dieksekusi oleh runtime PHP, hasilnya adalah kode PHP yang dinamis bergantung kepada *script* PHP yang dituliskan. PHP dapat digunakan di banyak server web, system operasi, dan *platform.*

Selain itu, PHP juga digunakan di system manajemen *database* relasional (RDBMS). Semua ini dapat diperoleh gratis, dan Group PHP menyediakan *source code* lengkap bagi pengguna untuk membangun, menyesuaikan, dan mengutak-atik sesuai dengan fungsi yang diinginkan. (Winarno , Ali Zaki, SmiDev Community, 2010).

2.6.2 Konsep Kerja PHP

PHP secara mendasar dapat mengerjakan semua yang dapat dikerjakan oleh program CGI, seperti mendapatkan data dari form, menghasilkan isi halaman web yang dinamik, dan menerima *cookies*. Kemampuan (*Feature*)

PHP yang paling diandalkan dan signifikan adalah dukungan kepada banyak *database*. Membuat halaman web yang menggunakan data dari database dengan sangat mudah dapat dilakukan. Berikut adalah *database* yang didukung oleh PHP, diantaranya: Adabas D, dBase, Empress, FilePro (read only), FrontBase, Hyperwave, IBM DB2, Informix, Ingres, Interbase, MSQL, Direct MS SQL, MySQL, ODBC, Oracle (OCI7 dan OCI8), Ovrimos, PostgreSQL, Solid, SQLite, Sybase, Velocis, dan Unix DBM. PHP juga mendukung untuk berkomunikasi dengan layanan lain menggunakan protokol IMAP, SNMP, NNTP, POP3, HTTP, dan lainnya yang tidak terhitung. Pemrograman juga dapat membuka soket jaringan secara mentah dan berinteraksi dengan menggunakan protokol lainnya (Ir. Betha Sidik, 2004)

2.7 MySQL

2.7.1 Definisi

MySQL adalah sebuah *software database. Database* merupakan sebuah tempat untuk menyimpan data yang jenisnya beraneka ragam. MySQL merupakan tipe data relasional yang artinya MySQL menyimpan datanya dalam bentuk tabel-tabel yang saling berhubungan. Keuntungan menyimpan data di database adalah kemudahannya dalam penyimpanan dan menampilkan data karena dalam bentuk tabel. (Winarno , Ali Zaki, SmiDev Community, 2010)

2.7.2 Konsep Kerja MySQL

MySQL termasuk *database* yang memiliki beberapa distro yang mampu dijalankan pada beberapa *platform*, meski secara resmi tentu saja memiliki sebuah *platform* tersendiri. Pada mulanya MySQL memang hanya berjalan di bawah *platform* Unix dan Linux, akan tetapi dalam perkembangannya sekarang bermunculan beberapa distro yang mampu berjalan pada beberapa *platform* yang sifatnya *shareware* dan *corporate*, seperti Windows.

MySQL sendiri adalah sebuah *database server* yang mampu menangani beberapa *user* di dalamnya. Dengan demikian, MySQL juga mampu menangani beberapa instruksi sekaligus dalam setiap waktu akses. Di dalam sistemnya, MySQL merekam semua data *user* dalam sebuah *table user* yang berada pada *database* yang bernama MySql. Dalam *table user* tersebut semua akses dan hak akses user mampu ditangani dengan baik.

Sebagai sebuah *database server* tentunya MySQL mampu menangani beberapa aplikasi lain yang akan mengakses data yang disimpannya. Aplikasi-aplikasi tersebut dapat berupa program kompiler maupun bahasa *Scripting Server Site* seperti PHP, Perl, CGI, Java dan lain sebagainya. Karena kebutuhan tersebut, server MySQL memerlukan sebuah program lain untuk menghubungkan aplikasi dengan data server. Program tersebut biasanya berupa MyODBC untuk menghubungkan program yang bersifat kompiler ataupun *Personal Web Server* (PWS) untuk menangani aplikasi-aplikasi yang berbasis web.

Karena kondisi MySQL sebagai *Server Database*, ia dapat berjalan secara *online* di dalam jaringan, sehingga para pengguna dan administrator dapat mengelola *database* dan servernya sendiri melalui jaringan. Untuk pengguna Linux dapat mengaksesnya melalui program *client* seperti SSH maupun Telnet.

Berikut adalah beberapa kelebihan MySQL dalam teknik mengakses data maupun mengelola servernya:

a. *Local Client* : MySQL sebagai *Server Database*, juga dapat berfungsi sebagai *client* yang dijalankan pada komputer lokal di mana MySQL itu berjalan.
b. *Remote Client* : Sebagai sebuah *database server* MySQL dapat diakses melalui komputer jaringan dengan cara *Remote*. Pada proses *remote* ini dapat digunakan program *client* yang berjalan pada aplikasi Unix seperti SSH ataupun Telnet yang sebelumnya harus di *Login* terlebih dahulu pada sistem komputer server. Apabila pada Windows, dapat digunakan program Client seperti Putty, telnet, psftp, atau lainnya sesuai keinginan.

c. *Remote Login* : MySQL dapat diakses dengan menggunakan program MySQL yang berada pada komputer lain, sehingga komputer tersebut berfungsi sebagai *client* MySQL.
d. *Web Browser* : Dengan menggunakan fungsi-fungsi API yang dimiliki MySQL, maka database dapat diakses menggunakan program yang berbasis *Server Site* yang berjalan di bawah Web Browser.
e. *Scripting Language* : MySQL juga dapat diakses melakui program aplikasi *client* yang kita ciptakan sendiri dengan menggunakan pemrograman visual maupun non-visual yang berjalan di dalam jaringan. Untuk program-program client yang berbasis GUI, anda dapat menggunakan MySQL Manager (Bunafit, 2004).

2.8 Framework Codeigniter

2.8.1 Definisi

Framework adalah sebuah struktur konseptual dasar yang digunakan untuk memecahkan sebuah permasalahan, bahkan isu-isu kompleks yang ada. Sebuah *framework* telah berisi sekumpulan arsitektur atau konsep-konsep yang dapat mempermudah dalam pemecahan sebuah permasalahan. *Framework* bukanlah peralatan atau *tools* untuk memecahkan sebuah masalah, tetapi sebagai alat bantu. *Framework* hanya menjadi sebuah konstruksi dasar yang menopang sebuah konsep atau sistem yang bersifat "*essential support*" (penting tapi bukan komponen utama). Di dalam sebuah *framework* biasanya sudah tersedia struktur aplikasi yang baik, diantaranya adalah *standard coding*, *best practice*, *desaign pattern*, dan *common function*. Dengan menggunakan *framework* kita dapat langsung fokus kepada *business* process yang dihadapi tanpa harus berfikir banyak masalah struktur aplikasi, standar *coding* dan lain-lain

Codeigniter (CI) adalah sebuah *web application framework* yang bersifat *open source* digunakan untuk membangun aplikasi PHP dinamis. Tujuan utama pengembangan *codeigniter* adalah untuk membantu *developer* untuk mengerjakan aplikasi lebih cepat dari pada menulis semua kode dari awal.

Codeigniter menyediakan berbagai macam *library* yang dapat mempermudah dalam pengembangan. *Codeigniter* dibangun menggunakan konsep *Model-View-Controller development pattern*. *Codeigniter* sendiri merupakan framework tercepat dibandingkan dengan framework lainnya.

2.8.2 Konsep Kerja Framework Codeigniter

Framework codeigniter bekerja dengan konsep MVC (*Model-View-Controller*). MVC adalah sebuah konsep dasar yang harus diketahui sebelum mengenal *codeigniter*. MVC sebenarnya adalah sebuah *pattern*/teknik pemrograman yang memisahkan bisnis *logic* (alur pikir), data *logic* (penyimpanan data) dan presentasi *logic* (antarmuka aplikasi) atau secara sederhana adalah memisahkan antara desain, data dan proses. Adapaun komponen-komponen MVC antara lain:

a. Model

Model berhubungan dengan data dan interaksi ke *database* atau *web service*. Model juga merepresentasikan struktur data dari aplikasi yang bisa berupa basis data maupun data lain, misalnya dalam bentuk teks, *file* XML maupun *web service*. Biasanya di dalam *model* akan berisi *class* dan fungsi untuk mengambil, melakukan *update* dan menghapus data *website*. Sebuah aplikasi web biasanya menggunakan basis data dalam menyimpan data, maka pada bagian *model* biasanya akan berhubungan dengan perintah-perintah *query* SQL.

b. View

View berhubungan dengan segala sesuatu yang akan ditampilkan ke *end-user*. Bisa berupa halaman *web*, *rss*, *javascript*, dan lain-lain. Kita harus menghindari adanya logika atau pemrosesan data di *view*. Di dalam *view* hanya berisi variabel-variabel yang berisi data yang siap ditampilkan. *View* dapat dikatakan sebagai halaman *website* yang dibuat dengan menggunakan HTML dan bantuan CSS atau JavaScript. Di dalam *view* jangan pernah ada kode untuk melakukan koneksi ke

database. *View* hanya dikhususkan untuk menampilkan data-data hasil dari *model* dan *controller*.

c. Controller

Controller bertindak sebagai penghubung data dan *view*. Di dalam *controller* inilah terdapat *class-class* dan fungsi-fungsi yang memproses permintaan dari *view* ke dalam struktur data di dalam *model*. *Controller* juga tidak boleh berisi kode untuk mengakses basis data karena tugas mengakses basis data telah diserahkan kepada *model*. Tugas *controller* adalah menyediakan berbagai variabel yang akan ditampilkan di *view*, memanggil model untuk melakukan akses ke basis data, menyediakan penanganan kesalahan/*error*, mengerjakan proses logika dari aplikasi serta melakukan validasi atau cek terhadap *input* (Ibnu, 2011).

2.9 *HTML*

HTML (*HyperText Markup Language*) merupakan salah satu format yang digunakan untuk membuat dokumen *hypertext* yang dibaca dari suatu *platform* komputer ke *platform* komputer lainnya tanpa harus melakukan perubahan apapun (Kadir, 2002). Meskipun telah banyak terdapat *tool* yang dapat digunakan untuk membangun halaman secara *WYSYWYG* (*what you see is what you get*) seperti *front page*, *dreamweaver* dan *NetscapeEditor*, namun kita tetap harus menguasai *HTML*.

2.10 *Rational Rose*

Rational Rose adalah *tools* pemodelan visual untuk pengembangan sistem berbasis objek yang sangat handal untuk digunakan sebagai bantuan bagi para pengembang dalam melakukan analisis dan perancangan sistem (Nugroho, 2005). *Rational Rose* digunakan untuk melakukan pemodelan sistem sebelum pengembang menulis kode-kode dalam bahasa pemrograman. *Rasional Rose* mendukung pemodelan bisnis, yang komprehensif. *Tools* ini juga membantu analisis sistem dengan cara pengembang membuat diagram *use case* untuk

melihat fungsionalitas sistem secara keseluruhan sesuai dengan harapan dan keinginan pengguna.

Dalam *Rational Rose*, pemodelan adalah cara melihat sistem dari berbagai sudut pandang dan ia juga mencakup semua diagram yang di kenal dalam UML. Alat bantu ini membantu keakuratan proses analisis dan desain, selain karena sudah memiliki notasi sehingga dapat digunakan dengan metode *drag* dan *drop*, alat ini juga sudah menotomatisasi beberapa elemen sehingga saling terhubung dan dapat mengidentifikasi dengan mudah bila terjadi perubahan pada satu elemen karena perubahan pada elemen lainnya.

2.10 *Personal Finance*

Dalam Bahasa indonesia diartikan sebagai laporan keuangan secara personal. Pada umumnya laporan keuangan digunakan untuk perusahaan, tetapi semakin berkembangnya zaman, laporan keuangan personal terasa sangat penting. Laporan keuangan merupakan ringkasan dari suatu proses pencatatan atau suatu ringkasan dari transaksi-transaksi keuangan yang terjadi selama tahun buku yang bersangkutan.

Sedangkan pengertian laporan keuangan menurut Standar Akuntansi Keuangan dalam Kerangka Dasar Penyusunan dan Penyajian Laporan keuangan paragraf 07 menyatakan bahwa :

"*Laporan Keuangan merupakan bagian dari proses pelaporan keuangan. Laporan yang lengkap biasanya meliputi neraca, laporan laba rugi, laporan perubahan posisi keuangan (yang dapat disajikan dalam berbagai cara misalnya : sebagai laporan arus kas atau laporan arus dana), catatan dan laporan ini serta materi penjelasan yang merupakan bagian integral dan informasi tambahan yang berkaitan dengan laporan tersebut misalnya, informasi keuangan segmen industri dan geografis serta pengungkapan pengaruh perubahan harga."* (Ikatan Akuntansi Indonesia, 2002:2)

Laporan keuangan ini dibuat oleh manajemen dengan tujuan untuk mempertanggungjawabkan tugas-tugas yang dibebankan kepadanya oleh para pemilik perusahaan. Disamping itu, laporan keuangan dapat juga digunakan untuk memenuhi tujuan-tujuan lain yaitu sebagai laporan kepada pihak-pihak luar perusahaan.

2.12 *Transaction*

Pengertian transaksi menurut Kamus Besar Bahasa Indonesia (KBBI) adalah "Persetujuan jual beli dalam perdagangan antara pihak pembeli dan penjual".

Transaksi adalah pertemuan antara dua belah pihak (Penjual dan Pembeli) yang saling menguntungkan dengan adanya data / bukti / dokumen pendukung yang dimasukkan kedalam jurnal setelah melalui pencatatan", (Bastian, 2007). Menurut pendapat lainnya Transaksi adalah kejadian yang dapat mempengaruhi posisi keuangan suatu badan usaha serta sebagai hal yang wajar untuk di catat.

2.13 *Budgets*

Budgets dalam bahasa indonesia disebut anggaran merupakan suatu susunan rencana yang mencakup seluruh rangkaian pemasukandan pengeluaran suatu pihak / badan dalam periode tertentu agar terjadi keseimbangan antara arus masuk dan keluar.

Business Budget atau *Budget* (Anggaran) ialah suatu rencana yang disusun secara sistematis, yang meliputi seluruh kegiatan perusahaan(yang menimbulkan penerimaan/hak dan juga pengeluaran/kewajiban),yang dinyatakan dalam unit (kesatuan) moneter dan berlaku untuk jangka waktu / periode tertentu yang akan datang (Munandar, 1997)

Budget disusun pada awal periode sehingga pihak / badan terebut mempunyai pedoman mengenai target pemasukan dan batas pengeluaran yang jelas.

2.14 *Trends*

Trend atau *tendensi* posisi dan kemajuan keuangan perusahaan yang dinyatakan dalam presentase (*trend presentase analysis*), adalah suatu metode atau teknik analisa untuk mengetahui tendensi daripada keadaan keuangannya, apakah menunjukkan tendensi tetap, naik atau bahkan turun.

2.15 *Account Aggregation*

Account aggregation is one of the conveniences offered by online banking and online financial apps (www.financialsoft.about.com). Secara harfiah *account aggregation* diartikan sebagai aggregasi terhadap akun yang dimiliki, sedangkan secara teori keuangan *account aggregation* merupakan layanan online untuk individu yang memungkinkan menggabungkan berbagai macam rekening, tagihan, kartu kredit dan informasi lainnya menjadi satu untuk mempermudah mengontrol keuangan personal.

Metode ini mencakup menggabungkan informasi dari rekening-rekening yang berbeda, yang mana termasuk rekening bank , kartu kredit, investasi dan lainnya dalam satu tempat.

Hal ini membutuhkan sebuah *database* atau dapat menggunakan *screen scrapping* dimana pengguna menyediakan informasi mengenai akunnya sehingga sistem secara otomatis dapat mengumpulkan informasi dan menggabungkan ke dalam satu halaman, biasanya database ini berada didalam aplikasi berbasis *web* atau *client-side sofware.* Dimana layanan seperti ini dirancang terutama untuk mengagregasi informasi keuangan, terkadang mereka juga menampilkan hal-hal lain seperti isi dari *email* dan *news headlines.*

BAB III
METODOLOGI PENELITIAN

Pada bab ini akan dibahas mengenai perencanaan kebutuhan, analisis, dan perancangan sesuai dengan metode *GRAPPLE*. Dalam perencanaan kebutuhan digunakan diagram *use case level* bisnis dan diagram *high-level class*. Pada tahap analisis akan digunakan dua diagram UML yaitu diagram *use case* dan diagram *class*. Sedangkan dalam tahap perancangan menggunakan perancangan database, perancangan struktur menu dan perancangan antarmuka aplikasi. Tahap berikutnya yaitu tahap pengembangan, akan dibahas dalam bab IV.

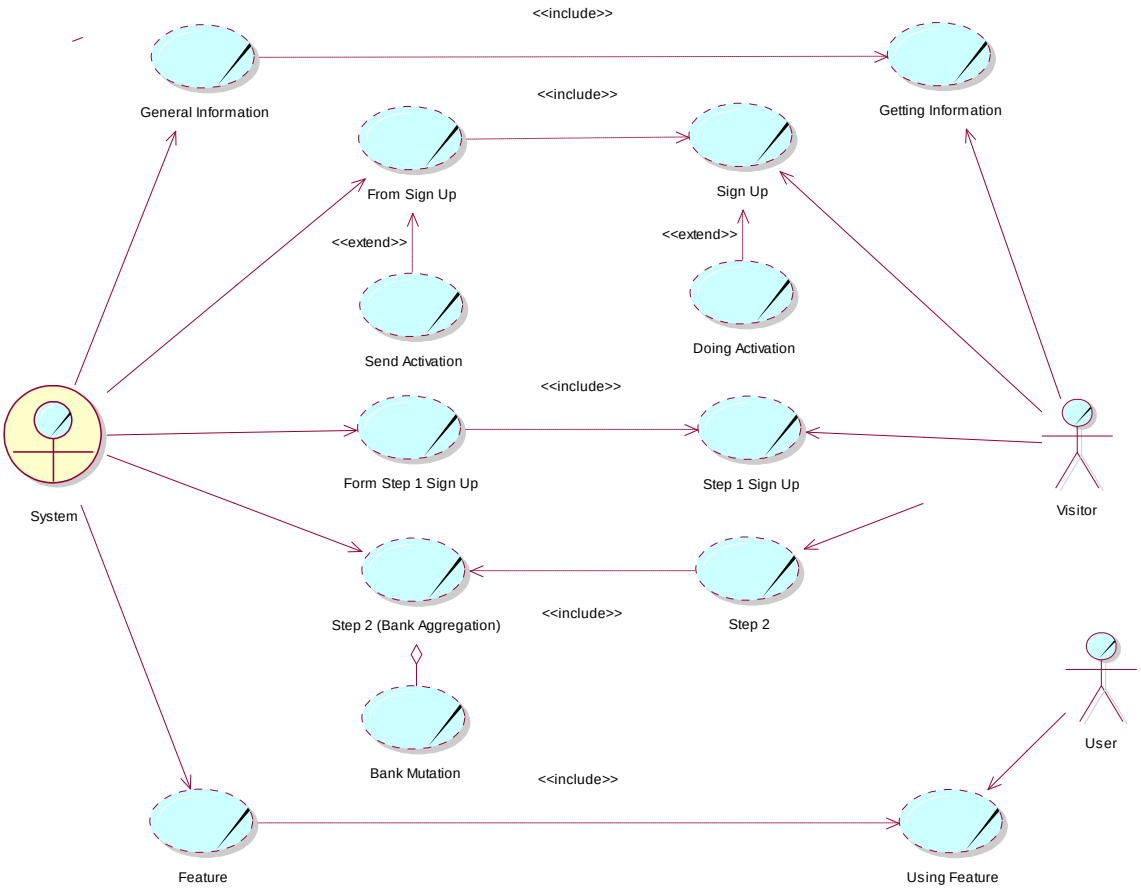
3.1 *Requirement Gathering*

3.1.1 Diagram *Use Case Level* Bisnis

Untuk membuat diagram *use case level* bisnis, yang dilakukan adalah mengambil informasi lengkap dari aplikasi, tentang secara garis besar apa saja yang bisa dilakukan oleh aplikasi yang terbagi dari beberapa aktor dan *requirement* yang dibutuhkan oleh setiap aktor. Dalam aplikasi ini, terdapat tiga aktor, yaitu *System*, *Visitor*, *User*. Berikut adalah daftar *requirement* dalam Dompetsehat.

1. *System* memberikan informasi umum tentang Aplikasi Dompetsehat.
2. *System* menyediakan *form* pendaftaran untuk *User* dan melakukan pengiriman aktivasi melalui email.
3. *System* menyediakan *form* pendaftaran selanjutnya disebut *step 1*.
4. *System* menyediakan form *step 2* yang berisi *form* untuk mengagregasi mutasi *Bank*
5. *System* menyediakan fitur fitur aplikasi Dompetsehat.
6. *Visitor* mendapatkan informasi seputar aplikasi.
7. *Visitor* melakukan pendaftaran untuk menjadi user Dompetsehat.
8. *Visitor* melakukan Aktifasi yang telah di kirimkan system melalui email

9. *Visitor* mengisi Pendaftaran selanjutnya Step 1
10. *Visitor* melakukan Pendaftaran Step 2 dan mengisi akun Internet Banking yang di miliki
11. *User* mengunakan fitur fitur yang disediakan oleh aplikasi, antara lain *Overview* akun, Transaksi, *Budgets, Trends*, Rekening, Kategori dll.



Gambar 3.1 Diagram *Use Case Level* bisnis

3.1.2 Diagram *High Level Class*

Untuk membuat diagram *High Level Class*, dibutuhkan *requirement* kebutuhan pengguna. Dalam tahap ini akan dibahas kebutuhan tiap aktor dalam sistem yang akan dibuat, yaitu :

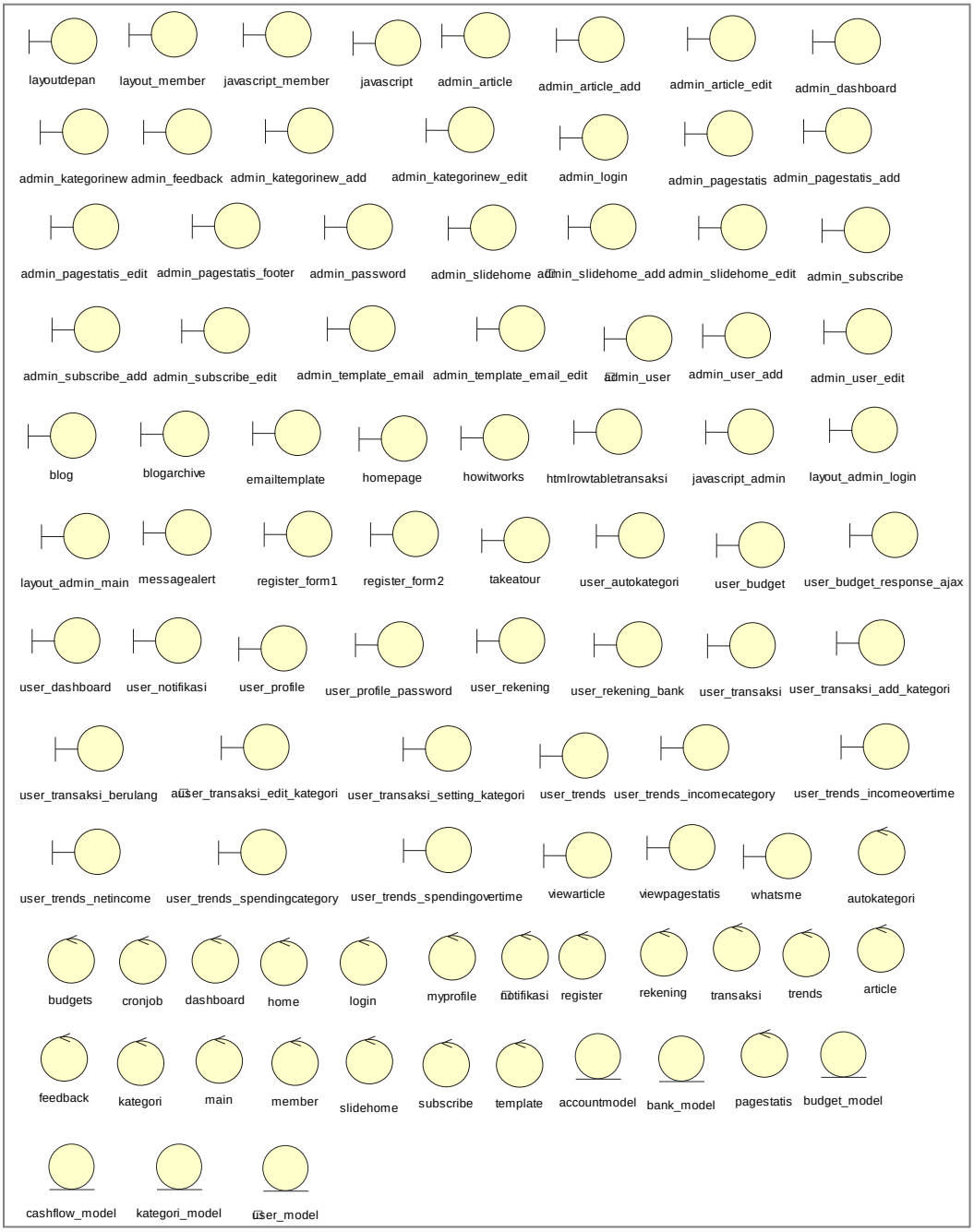
1. *Visitor* megisi *form* registrasi untuk melakukan registrasi.
2. *Visitor* masuk ke *What's Dompetsehat* untuk mengetahui apa itu aplikasi Dompetsehat.

3. *Visitor* masuk ke halaman *How it Work* untuk mengetahui Bagaimana aplikasi itu bekerja.
4. *Visitor* masuk ke *Take a tour* untuk melihat Demo singkat aplikasi .
5. *Visitor* masuk ke halaman *News* untuk melihat berita *realease* terbaru dari aplikasi.
6. *Visitor* melakukan registrasi melalui *form* registrasi
7. *Visitor* melakukan aktifasi yang dikirim ke email yang telah di inputkan saat registrasi.
8. *Visitor* mengikuti langkah regsitrasi selanjutnya *step 1* .
9. *Visitor* mengikuti langkah regsitrasi *step 2*, dan menginputkan username dan password internet banking.
10. *User login* untuk masuk ke halaman *dashboard overview*.
11. *User* masuk ke halaman *overview* untuk melihat ringkasan dari semua fitur-fitur .
12. *User* masuk ke halaman transaksi untuk mengolah transaksi yang telah di aggregasi, seperti menambahkan dan mengedit transaksi.
13. *User* masuk ke halaman *manage* kategori untuk mengolah kategori, menambahkan, mengedit dan menghapus kategori.
14. *User* masuk ke halaman *budgets* untuk mengolah *budgest*, menambahkan, mengedit menghapus, melihat hasil *budget* dan memfilter *budgets*.
15. *User* masuk ke halaman *trends* untuk melihat dan memfilter *trends*, yang terdiri dari *trends spending*, *trends income*, dan *trends net income* yang masing-masing terbagi menjadi dua bagian, yaitu melihat *trends by category* dan *by overtime.*
16. *User* masuk ke halaman Notifikasi untuk mengatur notifikasi yang diinginkan.
17. *User* masuk ke halaman Auto Kategori untuk menambahkan dan mengedit auto kategori.

18. *User* masuk ke halaman Rekening Manual untuk mengolah rekening secara manual, menambahkan rekening, mengedit rekening dan menghapus rekening.
19. *User* masuk ke halaman *Connect your bank account* untuk menambahkan *account aggregation*.
20. *User* masuk ke halaman *Profile* untuk mengedit *profile user.*
21. *User* masuk ke halaman ganti *password* untuk mengubah *password* user.
22. *Admin* masuk ke halaman *Login admin* untuk melakukan login ke *system administrator.*
23. *Admin* masuk ke halaman *dashboard.*
24. *Admin* masuk ke halaman *Slide Home* untuk mengolah *slide* yang ditampilkan di halaman *home website*, menambahkan, mengedit dan menghapus *slide*.
25. Admin masuk ke halaman *Page Statis* untuk mengolah halaman statis yang akan ditampilkan di menu *What's Dompetsehat, How it Work*, dan *Take a tour*, *admin* dapat menambahkan, mengedit dan menghapus halaman statis.
26. *Admin* masuk ke halaman *Article* untuk mengolah artikel yang akan ditampilkan pada menu *Updates*, admin dapat menambahkan, mengedit dan menghapus artikel.
27. *Admin* masuk ke halaman *Template* untuk mengedit *template email* yang akan dikirimkan kepada *user*.
28. *Admin* masuk ke halaman *Feedback* untuk melihat dan menghapus *feedback* dari user
29. *Admin* masuk ke halaman Kategori untuk mengolah master kategori, *admin* dapat menambahkan, mengedit dan menghapus kategori serta melakukan pencarian kategori.
30. *Admin* masuk ke halaman *Member* untuk mengolah *member*, *admin* dapat menambahkan, menghapus, mengedit dan melakukan aktifasi secara manual serta melakukan pencarian *user*.

31. Admin masuk ke halaman Ganti *Password* untuk mengubah *password* admin.

Dari *requirements* di atas dapat dibuat diagram *high level class* seperti di bawah ini :



Gambar 3.2 Diagram *High Level Class*

3.2 Analisis

Analisis merupakan proses untuk menganalisa dari hasil proses *requirement gathering* yang dapat dijelaskan dalam bentuk pemodelan menggunakan diagram UML, Dalam tahap ini akan digunakan diagram *use case* dan diagram *class*.

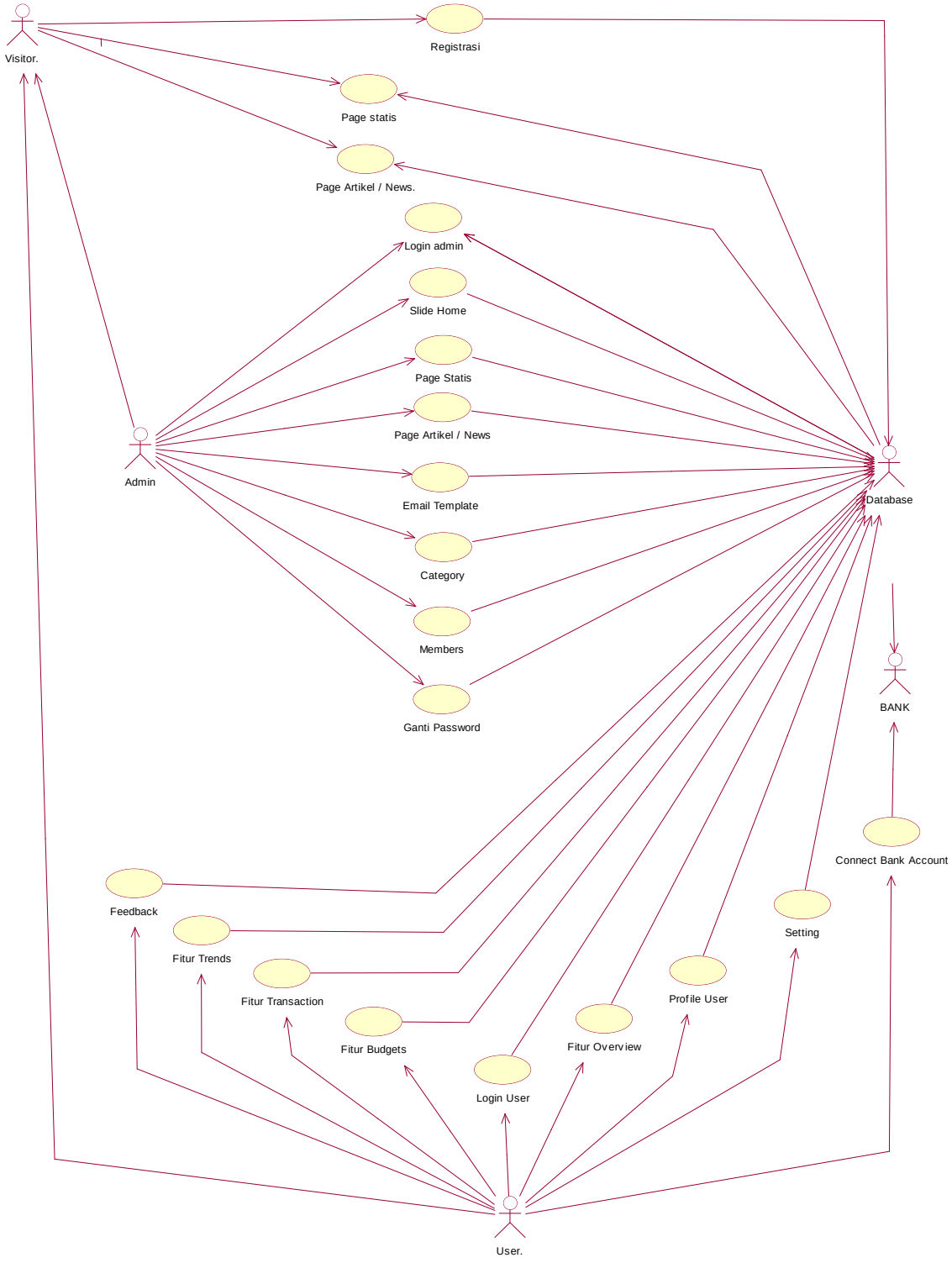
3.2.1 Diagram *Use Case*

Dalam pembuatan diagram *use case* ini, dibutuhkan analisis kebutuhan tiap aktor di dalam sistem yang dapat menjelaskan hubungan antara aktor dengan sistem yang akan dibuat.

1. Registrasi, *Visitor* harus registrasi terlebih dahulu untuk menjadi *User*.
2. *Page Statis*. *Visitor* dapat melihat informasi yang ada pada halaman statis tanpa *login* terlebih dahulu.
3. *Page Artikel / News*. *Visitor* dapat melihat berita terbaru dari aplikasi yang sudah di update oleh *admin* tanpa *login* terlebih dahulu.
4. *Login*. *Visitor* yang telah melakukan registrasi dan sudah menjadi *User*, *login* terlebih dahulu melalui *form login*.
5. Fitur *Setting*. User melakukan *setting* terhadap akunnya dan *user* harus *login* terlebih dahulu. *Setting* terdiri dari sub menu antara lain Notifikasi, Auto kategori, Auto Translate, Manual Akun dan Transfer dana antar akun rekening manual.
6. *Connect Bank Account. User* bisa mengkoneksikan akun internet banking *Bank Central Asia* ke dalam akun *User* tersebut, User harus login terlebih dahulu untuk mengakses menu ini. Setelah di koneksikan sistem akan mengambil mutasi dari akun *Internet Banking* tersebut kemudian dimasukan ke database dan di tampilkan kedalam transaksi *User*.
7. *Profile User*. User dapat mengedit *profile* dan user harus *login* terlebih dahulu.

8. Fitur *Overview*. User dapat menggunakan fitur *overview* setelah *login*, *user* bisa melihat singkatan dari semua fitur melalui halaman *overview* ini.
9. Fitur *Budgets*. *User* dapat menggunakan fitur *budgets*, mulai dari melihat detail *budgets* dan mengolah *budgets*.
10. Fitur *Transaction*. *User* bisa menambahkan dan mengedit transaksi, baik transaksi dari aggregasi bank atau transaksi manual.
11. Fitur *Trends*. *User* dapat melihat *trends* dari transaksi yang ada, trends berupa grafik yang terdiri dari grafik batang, grafik garis dan grafik lingkaran.
12. *Feedback*. *User* dapat memberikan *feedback* kepada *admin* dan *user* harus login terlebih dahulu.
13. *Login admin*. *Admin* harus login terlebih dahulu untuk mengakses halaman *admin*.
14. *Slide Home*. Hanya *admin* yang mempunyai wewenang untuk mengolah *Slide* yang ditampilkan pada halaman utama *visitor*.
15. *Page Statis*. Hanya *admin* yang mempunyai wewenang untuk mengolah halaman statis.
16. *Page Artikel / News*. Hanya *admin* yang mempunyai wewenang untuk mengolah halaman berita.
17. *Email Template*. Hanya *admin* yang mempunyai wewenang untuk mengedit *template email.*
18. *Category*. Hanya *admin* yang mempunyai wewenang untuk mengolah kategori.
19. *Members*. Admin dapat melakukan pengolahan data pengguna, yaitu *user* yang terdaftar dalam sistem.
20. Ganti *Password*. Admin dapat mengganti *password* secara berkala melalui *form* ganti *password.*

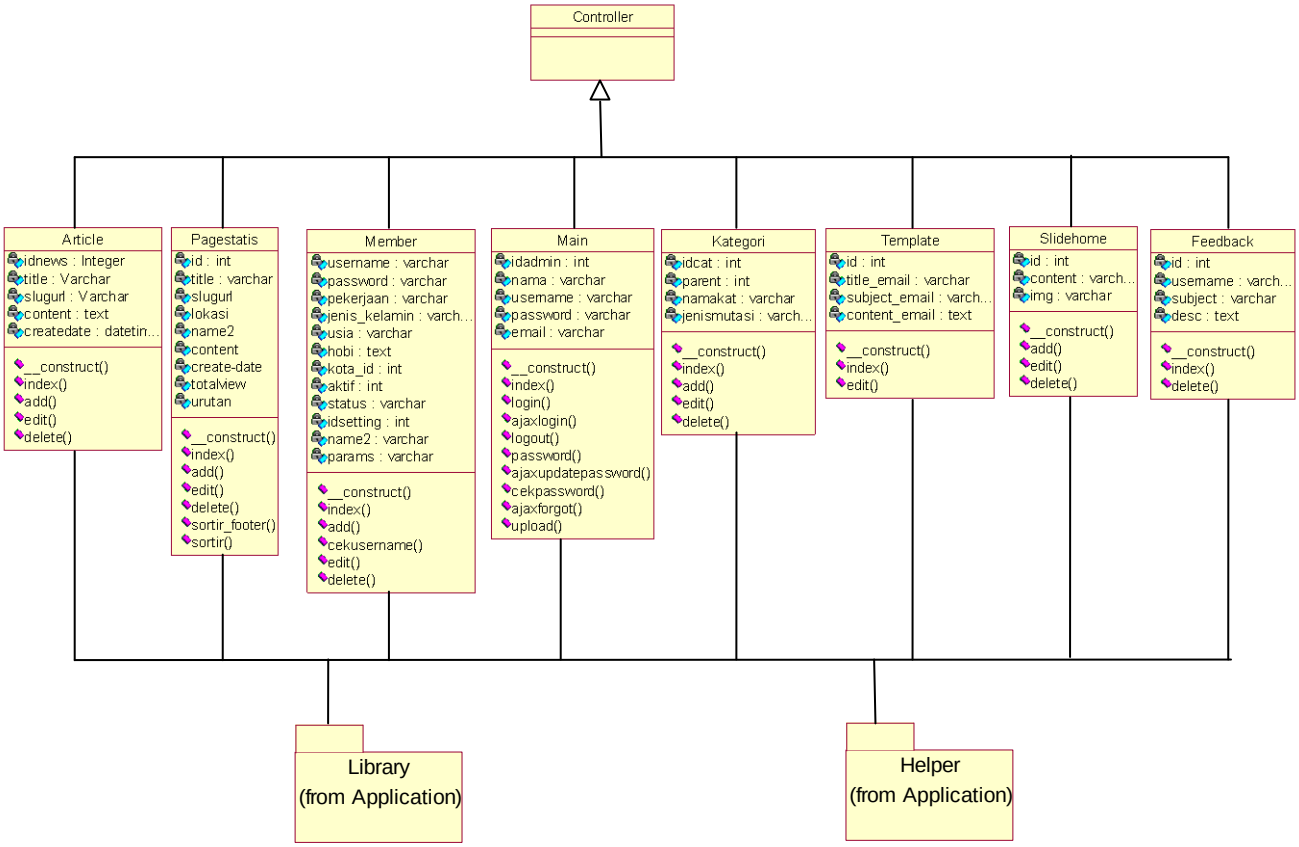
Dari analisis sistem diatas dapat dibuat *use case* seperti dibawah ini



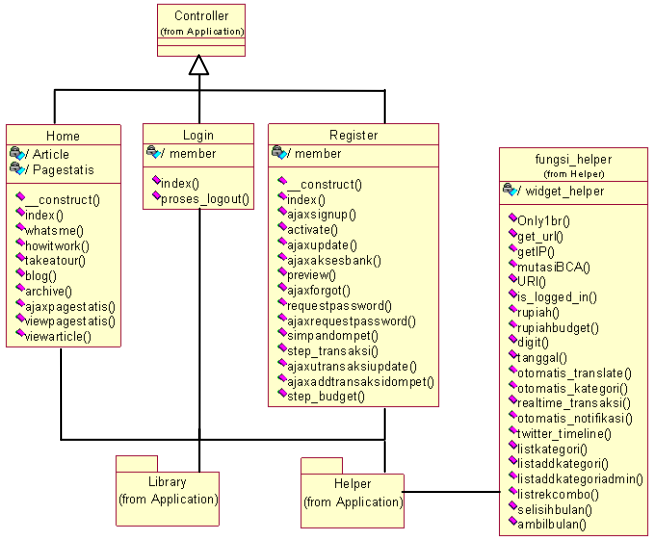
Gambar 3.3 Diagram *Use Case*

3.2.2 Diagram *Class*

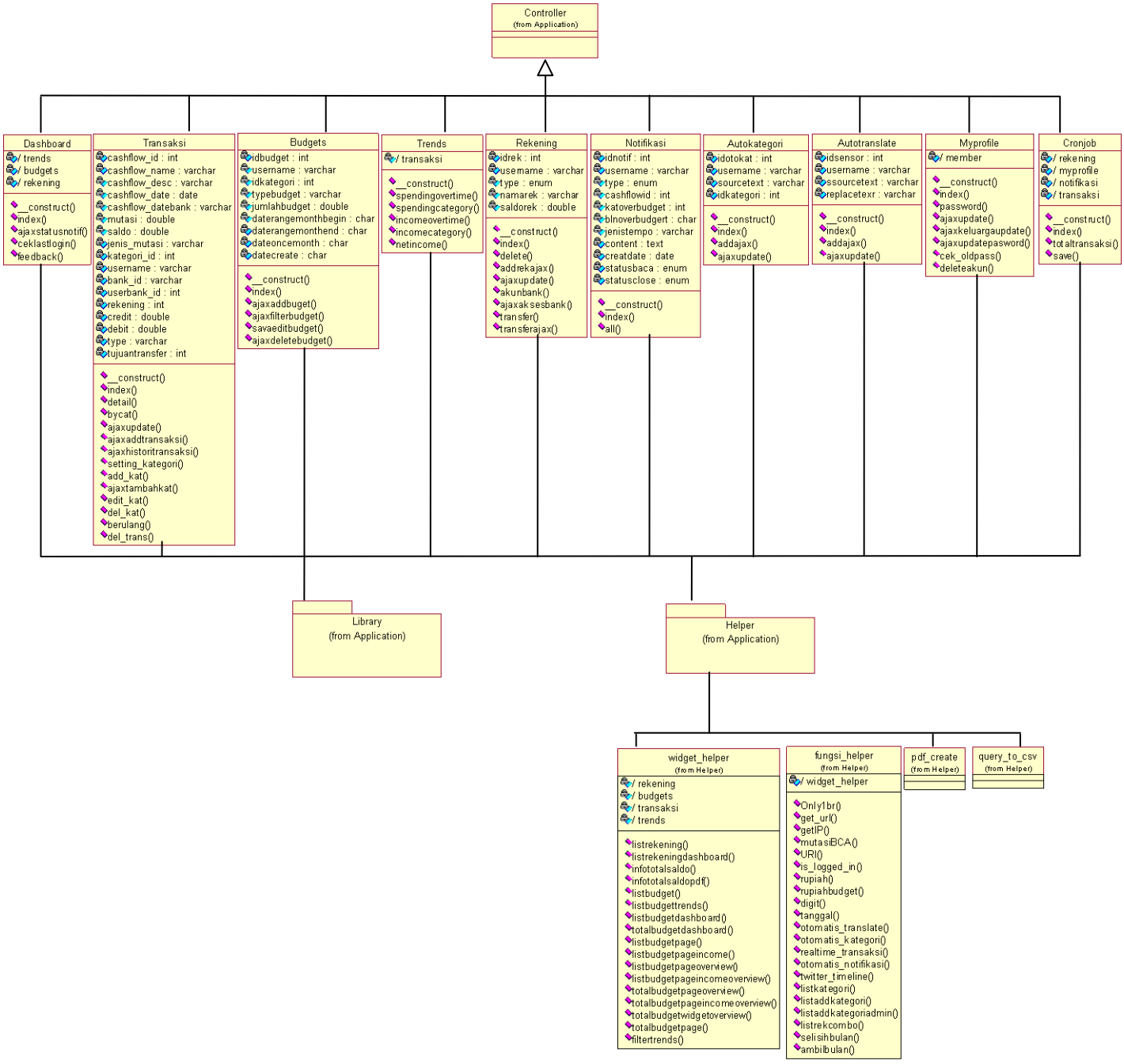
Diagram *class* terdiri dari tiga diagram, antara lain diagram *class administrator*, diagram *class visitor*, dan diagram *class user*.



Gambar 3.4 Diagram *Class Administrator*



Gambar 3.5 Diagram *Class Visitor*



Gambar 3.6 Diagram *Class User*

3.3 Perancangan

Tahap perancangan dilakukan untuk merancang solusi yang dihasilkan pada tahap analisis. Tahap ini terdiri dari perancangan basisdata, struktur menu, dan antarmuka

3.3.2 Basisdata

Dalam aplikasi ini digunakan sebuah basisdata yang terdiri dari 24 tabel. Tabel-tabel tersebut adalah sebagai berikut :

Tabel 3.1 Tabel admin

field	tipe data	constraint	default	keterangan
idadmin	*int(11)*	*not null*		
nama	*varchar(255)*	*null*		
username	*varchar(255)*	*null*		
password	*varchar(255)*	*null*		

Tabel 3.2 Tabel bank

field	tipe data	constraint	default	keterangan
bank_id	*varchar(255)*	*not null*		
bank_name	*varchar(255)*	*not null*		
image	*varchar(255)*	*not null*		

Tabel 3.3 Tabel feedback

field	tipe data	constraint	default	keterangan
id	*int(11)*	*not null*		
username	*varchar(100)*	*null*		
subject	*varchar(200)*	*null*		

Tabel 3.4 Tabel jenis_tagihan

field	tipe data	constraint	default	keterangan
id	*int(11)*	*unsigned not null*		
jenistagihan	*varchar(200)*	*null*		

Tabel 3.5 Tabel kategorinew

field	tipe data	constraint	default	keterangan
idkat	*int(11)*	*not null*		
parent	*int(11)*	*null*		
namakat	*varchar(200)*	*null*		
jenismutasi	*varchar(2)*	*null*		

Tabel 3.6 Tabel kota

field	tipe data	constraint	Default	keterangan
kota_id	*int(11)*	*not null*		
kota_name	*varchar(255)*	*not null*		

Tabel 3.7 Tabel master_setting

field	tipe data	constraint	Default	keterangan
idsetting	*int(11)*	*not null*		
params	*varchar(200)*	*null*		
nilai	*varchar(200)*	*null*		

Tabel 3.8 Tabel news

field	tipe data	constraint	default	keterangan
idnews	*int(11)*	*not null*		
title	*varchar(200)*	*null*		
slugurl	*varchar(200)*	*null*		
metakeyword	*varchar(200)*	*null*		
content text	*text*	*null*		
createdate	*datetime*	*null*		
totalview	*int(11)*	*null*		

Tabel 3.9 Tabel otomatis_translate

field	tipe data	constraint	default	keterangan
idtranslate	*int(11)*	*not null*		
sourcetext	*varchar(200)*	*null*		
replacetext	*varchar(200)*	*null*		

Tabel 3.10 Tabel pagestatis

field	tipe data	constraint	default	keterangan
id	*int(11)*	*not null*		
title	*varchar(200)*	*null*		
slugurl	*varchar(200)*	*null*		
metakeyword	*varchar(200)*	*null*		
lokasi	*Enum ('howitworks','takeatour', 'footer','whatsme','bigboxhome', 'litleboxhomekanan','litleboxhomekiri')*	*null*		
content	*text*	*null*		
createdate	*datetime*	*null*		
totalview	*int(11)*	*null*		
urutan	*int(11)*	*null*		

Tabel 3.11 Tabel slidehome

field	tipe data	constraint	default	keterangan
id	*int(11)*	*not null*		
content	*varchar(200)*	*null*		
img	*varchar(200)*	*null*		

Tabel 3.12 Tabel template_email

field	tipe data	constraint	default	keterangan
id	*int(11)*	*not null*		
title_email	*varchar(100)*	*null*		
subject_email	*varchar(200)*	*null*		
content_email	*text*	*null*		

Tabel 3.13 Tabel template_notifikasi

field	tipe data	constraint	default	keterangan
id	*int(11)*	*not null*		
type_notif	*varchar(100)*	*null*		
title_notif	*varchar(200)*	*null*		
content_notif	*text*	*null*		

Tabel 3.14 Tabel user

field	tipe data	constraint	default	keterangan
username	*varchar(100)*	*not null*		
password	*varchar(50)*	*not null*		
fullname	*varchar(200)*	*null*		
pekerjaan	*varchar(200)*	*null*		
jenis_kelamin	*varchar(100)*	*null*		
usia	*varchar(255)*	*null*		
hobi	*text*	*null*		
kota_id	*int(10)*	*null*		
aktif	*int(1)*	*null*		
status	*varchar(50)*	*null*		
kodeaktifasi	*varchar(50)*	*null*		
last_login	*datetime*	*null*		
tgl_lahir	*date*	*null*		
status_nikah	*varchar(200)*	*null*		
pendidikan	*varchar(200)*	*null*		
keluarga_income	*varchar(200)*	*null*		
keluarga_status	*varchar(200)*	*null*		
keluarga_jmlanak	*int(2)*	*null*		
keluarga_kodepos	*int(6)*	*null*		
delete_akun	*varchar(100)*	*not null*		

Tabel 3.15 Tabel user_bank

field	tipe data	constraint	default	keterangan
userbank_id	*int(10)*	*not null*		
username	*varchar(255)*	*not null*		
usernameBank	*varchar(255)*	*not null*		
passwordBank	*varchar(255)*	*not null*		
bank_id	*varchar(255)*	*not null*		
saldoakhir	*double*	*not null*		
lastupdate	*datetime*	*null*		

Tabel 3.16 Tabel user_budget

field	tipe data	constraint	default	keterangan
idbudget	*int(11)*	*not null*		
username	*varchar(200)*	*null*		
idkategori	*int(11)*	*null*		
typebudget	*enum('everymonth', 'rangemonth','oncemonth')*	*null*		
jumlahbudget	*double*	*null*		
daterangemonthbegin	*char(7)*	*null*		
daterangemonthend	*char(7)*	*null*		
dateoncemonth	*char(7)*	*null*		
datecreate	*char(7)*	*null*		

Tabel 3.17 Tabel user_cashflow

field	tipe data	constraint	default	keterangan
cashflow_id	*int(11)*	*not null*		
cashflow_name	*varchar(255)*	*null*		
cashflow_desc	*varchar(255)*	*null*		
cashflow_date	*date*	*null*		
cashflow_date_bank	*varchar(255)*	*null*		
mutasi	*double*	*null*		

saldo	*double*	*null*		
jenis_mutasi	*varchar(2)*	*null*		
kategori_id	*int(5)*	*null*		
username	*varchar(255)*	*null*		
bank_id	*varchar(255)*	*null*		
userbank_id	*int(10)*	*null*		
rekening	*int(11)*	*null*		
credit	*double*	*null*		
debit	*double*	*null*		
type	*varchar(255)*	*null*		
tujuantransfer	*int(11)*	*not null*		

Tabel 3.18 Tabel user_kategori

field	tipe data	constraint	default	keterangan
id	*int(11)*	*not null*		
idkat	*int(11)*	*null*		
username	*varchar(200)*	*null*		
parent	*int(11)*	*null*		
namakat	*varchar(200)*	*null*		
jenismutasi	*varchar(2)*	*null*		

Tabel 3.19 Tabel user_notifikasi

field	tipe data	constraint	default	keterangan
idnotif	*int(11)*	*not null*		
username	*varchar(100)*	*null*		
type	*enum('uncategorized','welcome_user', 'overbudget','minimal_saldo','jatuh_tempo', 'tarik_tunai','transfer','pakai_kartu_debit', 'bayar_e_banking')*	*not null*		
cashflow_id	*int(11)*	*null*		

katoverbudget	*int(11)*	*null*		
blnoverbudget	*char(10)*	*null*		
jenistempo	*varchar(200)*	*null*		
content	*text*	*null*		
createdate	*datetime*	*null*		
statusbaca	*enum('read','unread')*	*null*		
statusclose	*enum('close','unclose')*	*null*		

Tabel 3.20 Tabel user_otomatis_kategori

field	Tipe data	constraint	default	keterangan
idotokat	*int(11)*	*not null*		
username	*varchar(200)*	*null*		
sourcetext	*varchar(200)*	*null*		
idkategori	*int(11)*	*null*		
idotokat	*int(11)*	*not null*		

Tabel 3.21 Tabel user_otomatis_sensor

field	tipe data	constraint	default	keterangan
idsensor	*int(11)*	*not null*		
username	*varchar(200)*	*null*		
sourcetext	*varchar(200)*	*null*		
replacetext	*varchar(200)*	*null*		

Tabel 3.22 Tabel user_rekening

field	tipe data	constraint	default	keterangan
idrek	*int(11)*	*not null*		
username	*varchar(200)*	*null*		
type	*enum('tunai','tabungan','online')*	*null*		
namarek	*varchar(200)*	*null*		
saldorek	*int(11)*	*not null*		

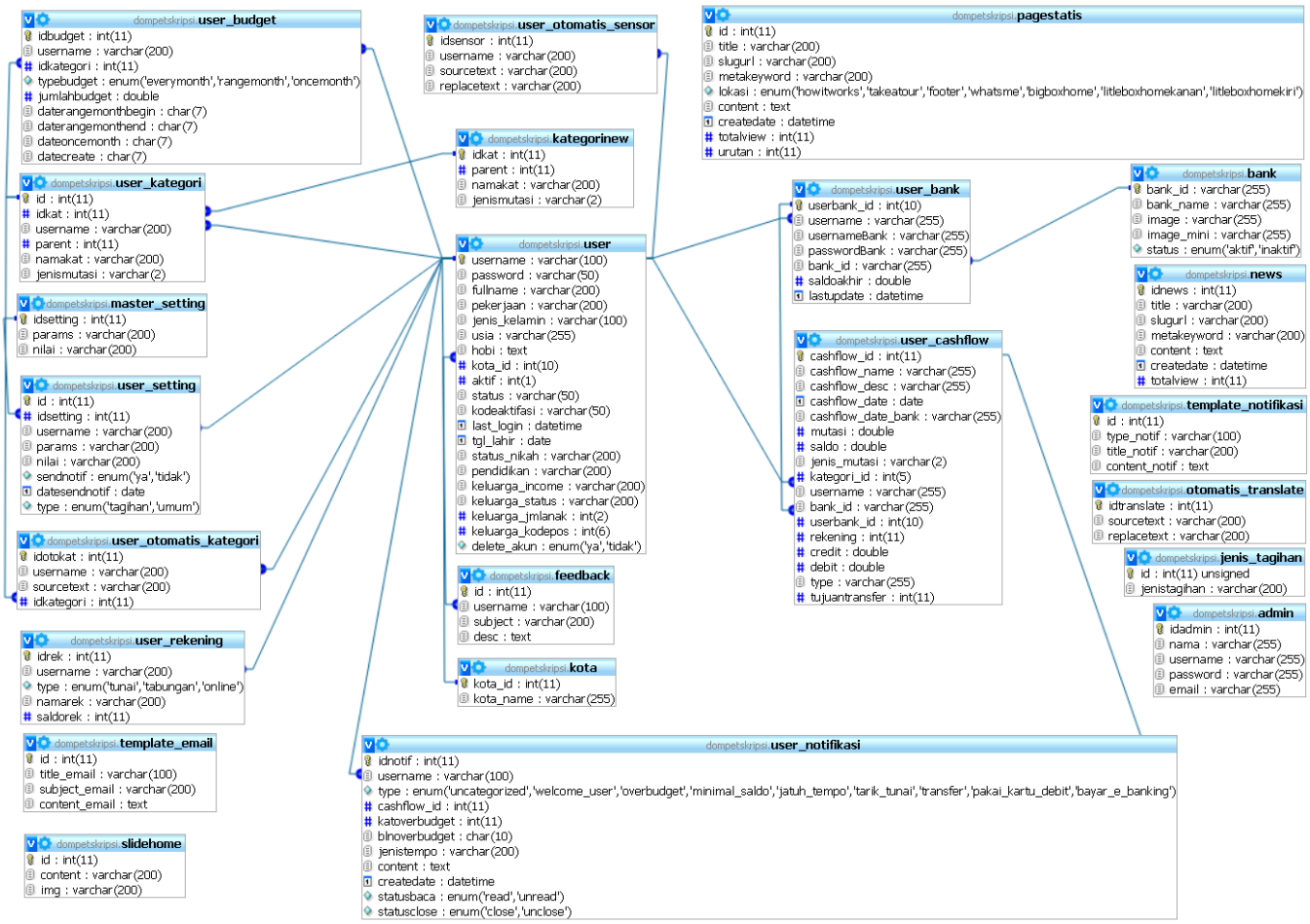
Tabel 3.23 Tabel user_setting

field	tipe data	constraint	default	keterangan
id	*int(11)*	*not null*		
idsetting	*int(11)*	*null*		
username	*varchar(200)*	*null*		
params	*varchar(200)*	*null*		
nilai	*varchar(200)*	*null*		
sendnotif	*enum('ya','tidak')*	*null*		
datesendnotif	*date*	*null*		
type	*enum('tagihan','umum')*	*null*		

Tabel 3.24 Tabel user_setting

field	tipe data	constraint	default	keterangan
id	*int(11)*	*not null*		
idsetting	*int(11)*	*null*		
username	*varchar(200)*	*null*		
params	*varchar(200)*	*null*		
nilai	*varchar(200)*	*null*		
sendnotif	*enum('ya','tidak')*	*null*		
datesendnotif	*date*	*null*		
type	*enum('tagihan','umum')*	*null*		

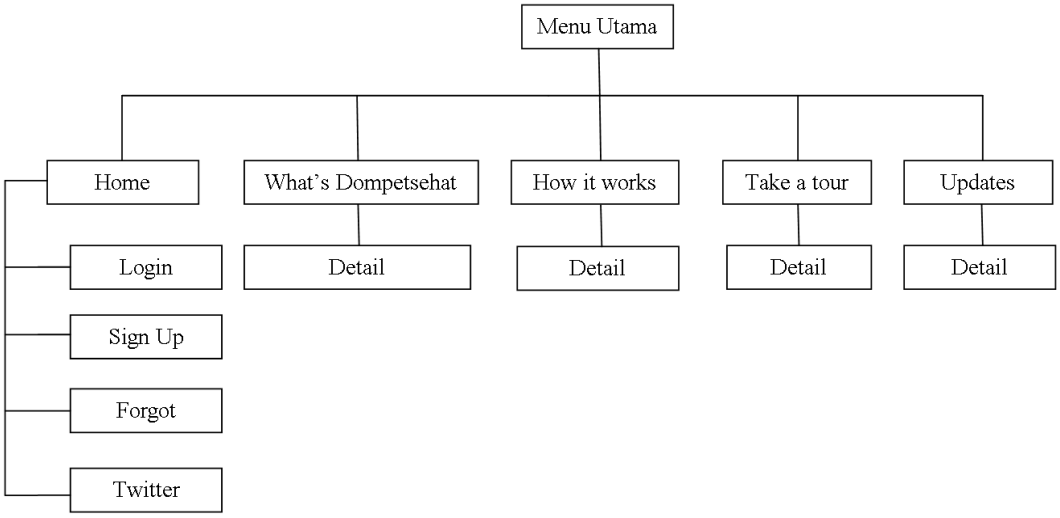
Dari tabel-tabel diatas dapat dibuat relasi antar tabel sebagai berikut:



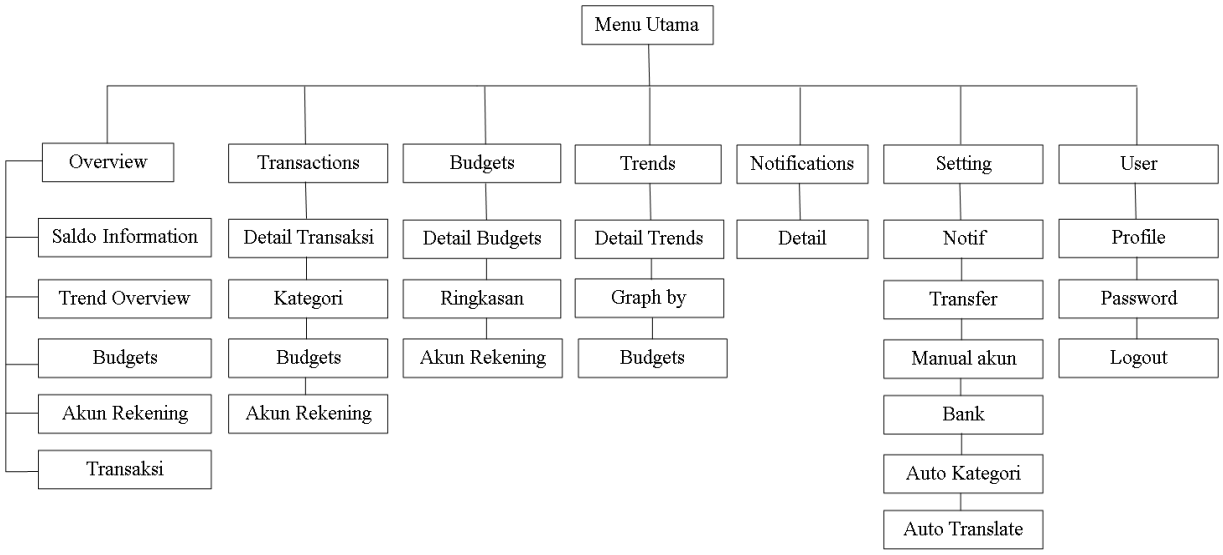
Gambar 3.7 Relasi Antar Tabel

3.2.3 Struktur Menu

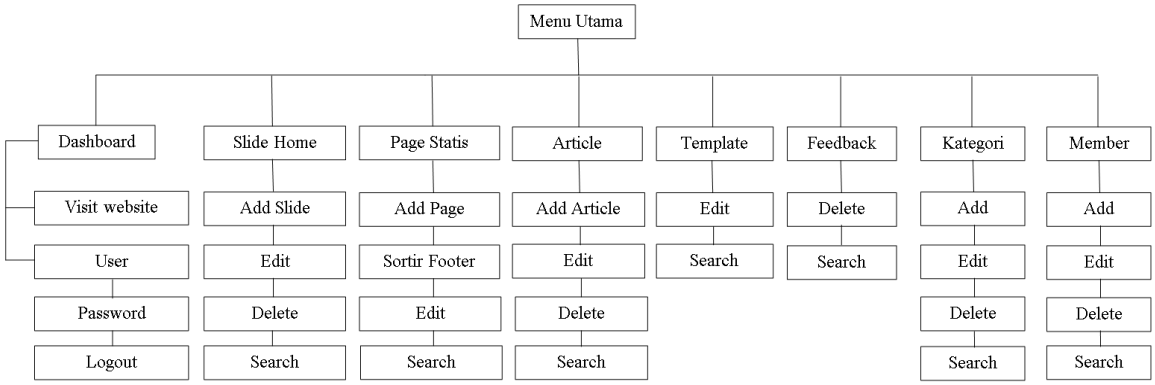
Perancangan struktur menu merupakan suatu kerangka prosedur dari suatu program, dimana tampilan arsitektur dalam suatu sistem perangkat lunak akan saling berhubungan dan membentuk suatu jaringan tampilan antar menu. Perancangan menu yang ada dalam Aplikasi Dompet Sehat adalah sebagai berikut :



Gambar 3.8 Perancangan Struktur Menu untuk *Visitor*



Gambar 3.9 Perancangan Struktur Menu untuk *User*



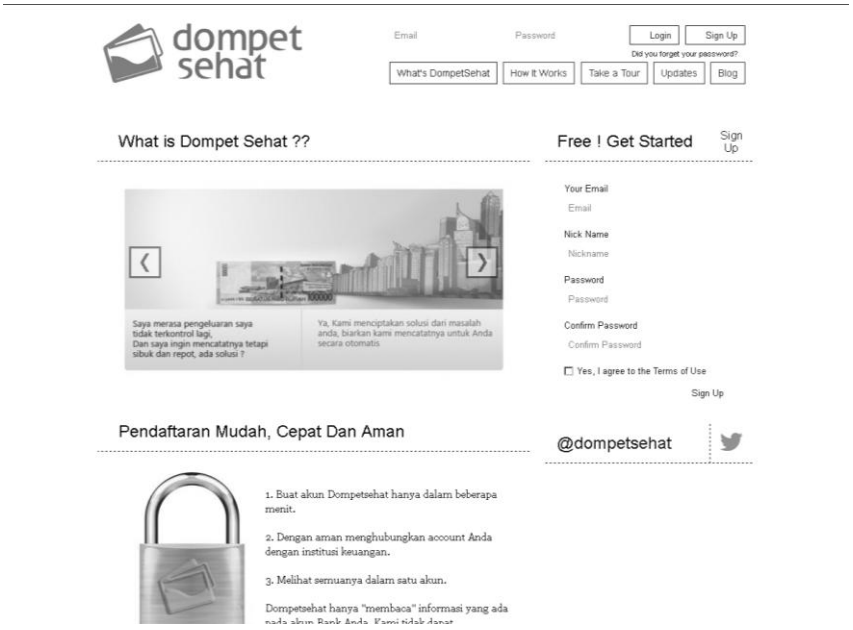
Gambar 3.10 Rancangan Struktur Menu untuk *Administrator*

3.2.4 Perancangan Antarmuka

Perancangan Antarmuka pada sistem ini terdiri dari beberapa antaramuka yang dibutuhkan dalam sitem informasi tersebut. Berikut adalah bagian dari perancangan antarmuka site tersebut :

1. Halaman Utama

   Halaman ini merupakan halaman utama / menu utama. Halaman pertama kali yang muncul setiap pengunjung yang mengunjungi *website* ini. Halaman ini menyediakan *link-link* ke halaman lainnya, sekaligus *form Login* dan *Sign Up* untuk menjadi *User*.



Gambar 3.11 Rancangan Halaman Utama *Visitor*

2. Halaman *What's DompetSehat*

   Halaman ini berisi tentang apa itu aplikasi DompetSehat dan terdapat tombol Daftar yang mengarahkan ke *form Sign Up*.



Gambar 3.12 Rancangan Halaman *What's DompetSehat*

3. Halaman *How it Works*

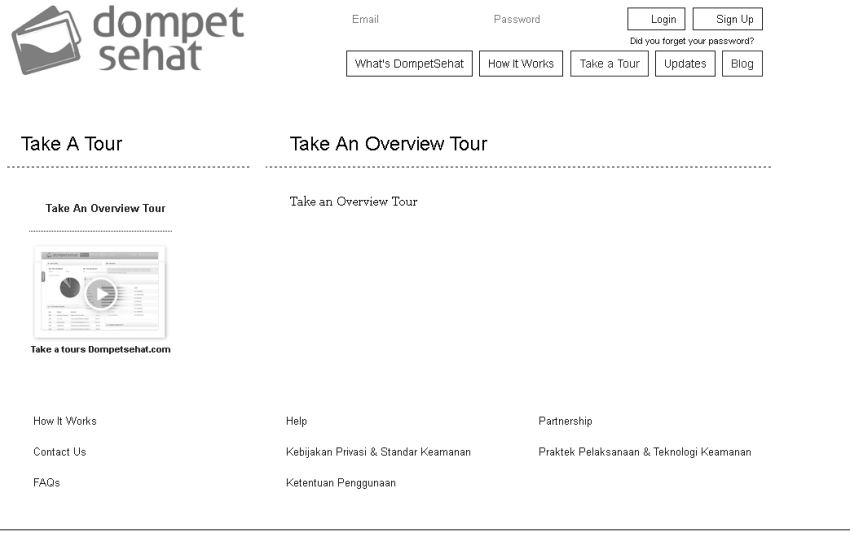
   Halaman ini berisi tentang informasi bagaimana aplikasi dompetsehat bekerja.



Gambar 3.13 Rancangan Halaman *How it works*

4. Halaman *Take a Tour*

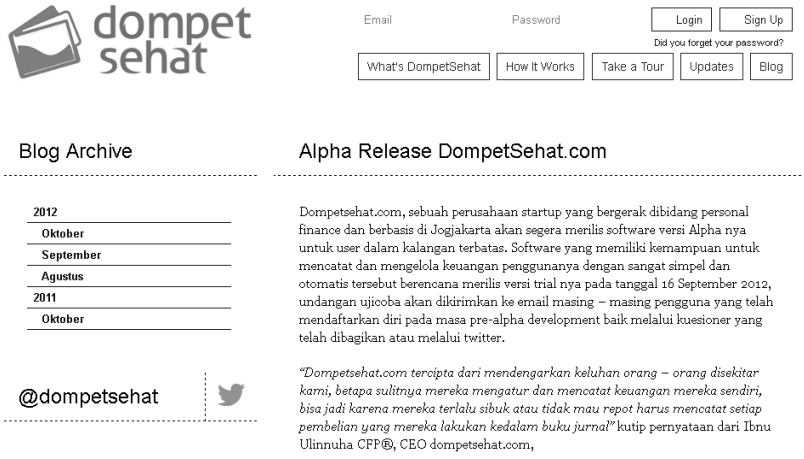
   Halaman ini berisi tentang informasi untuk *Visitor* agar bisa melihat demo aplikasi sebelum mendaftar.



Gambar 3.14 Rancangan Halaman *Take a tour*

5. Halaman Updates

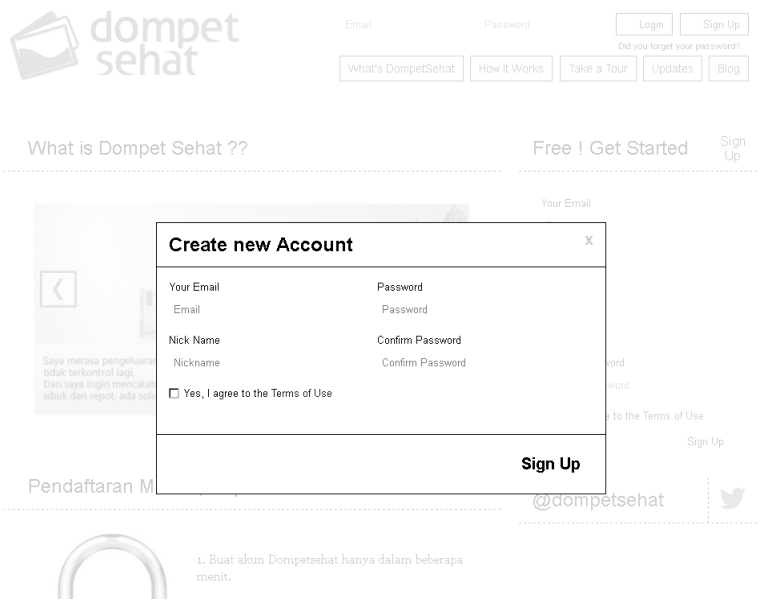
   Halaman Updates ini berisi tentang update aplikasi dan link ke blog.



Gambar 3.15 Rancangan Halaman *Updates*

6. Halaman *Sign Up*

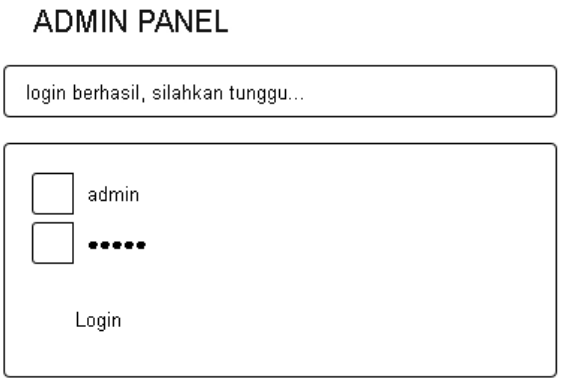
   Halaman ini berisi form untuk *visitor* yang ingin melakukan pendaftaran.



Gambar 3.16 Rancangan Halaman *Sign Up*

7. Halaman *Login Admin*

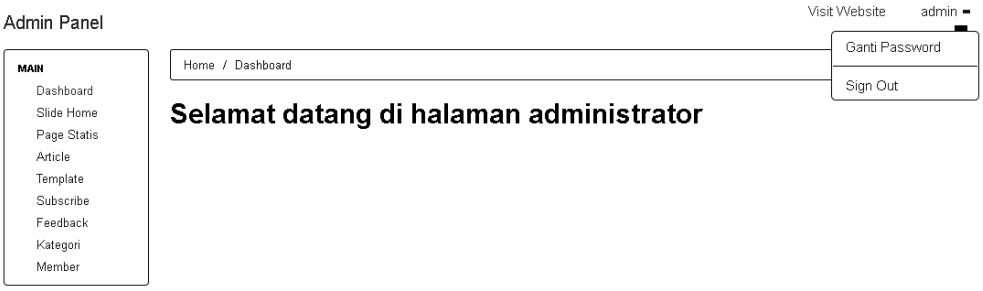
   Halaman ini khusus dibuat untuk *admin* yang mempunyai hak penuh untuk mengakses dan mengolah data.



Gambar 3.17 Rancangan Halaman *Login Admin*

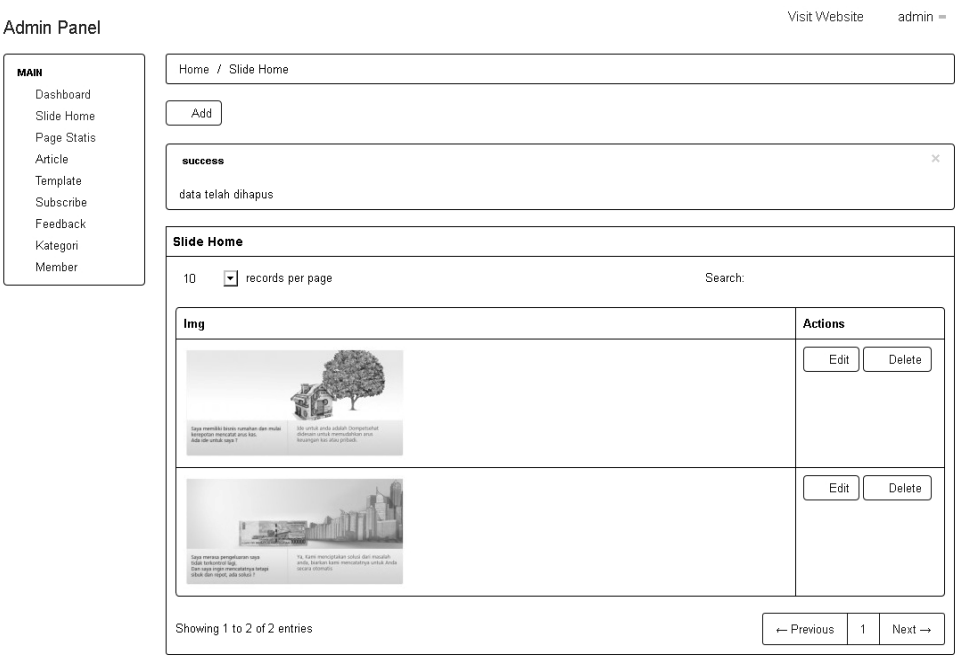
8. Halaman *Dashboard Admin*

Halaman ini adalah halaman *dashboard admin* yang pertama kali muncul setelah *admin login*.



Gambar 3.18 Rancangan Halaman *Dashboard Admin*

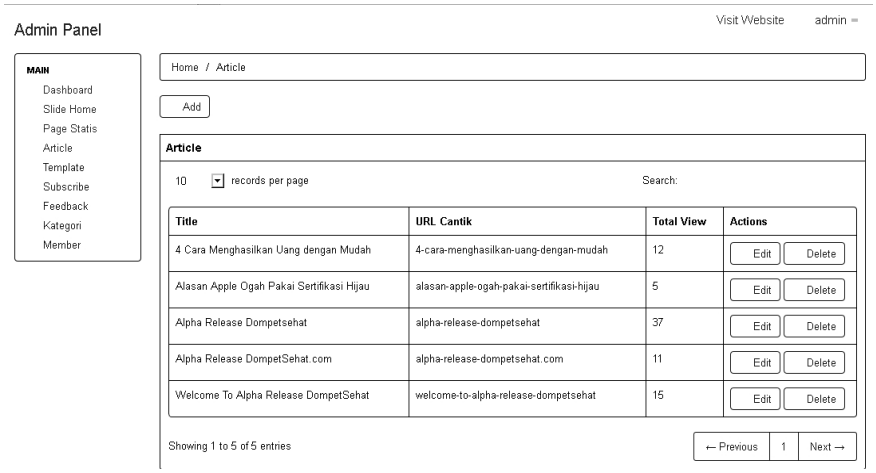
9. Halaman *Slide Home*



Gambar 3.19 Rancangan Halaman *Slide Home*

10. Halaman *Page Statis*

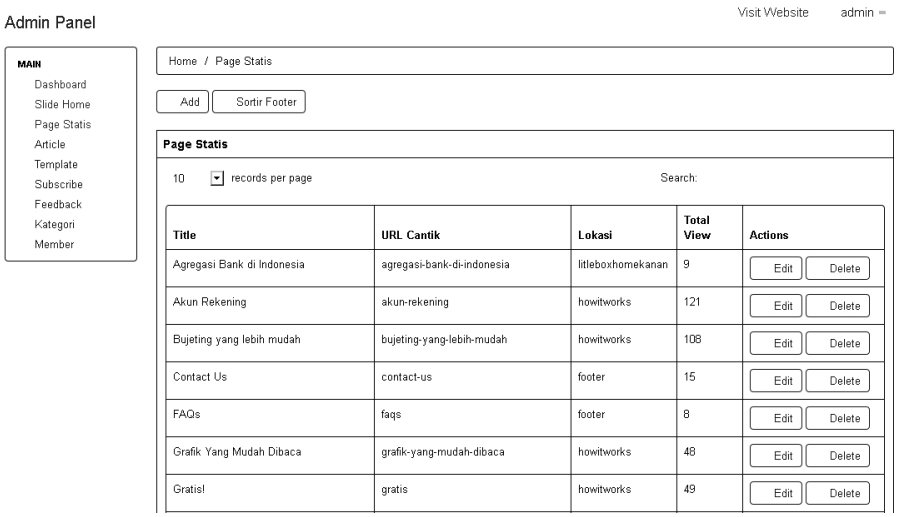
    Halaman ini menyediakan pilihan-pilihan untuk mengolah data *Page Statis*, mulai dari menambahkan, mengedit, menghapus dan mengurutkan halaman.



Gambar 3.20 Rancangan Halaman Olah Data Berita

11. Halaman *Article*

    Halaman ini menampilkan *article* yang telah di inputkan dan *admin* bisa mengolah *article* tersebut.



Gambar 3.21 Rancangan Halaman *Article*

12. Halaman Edit *Article*

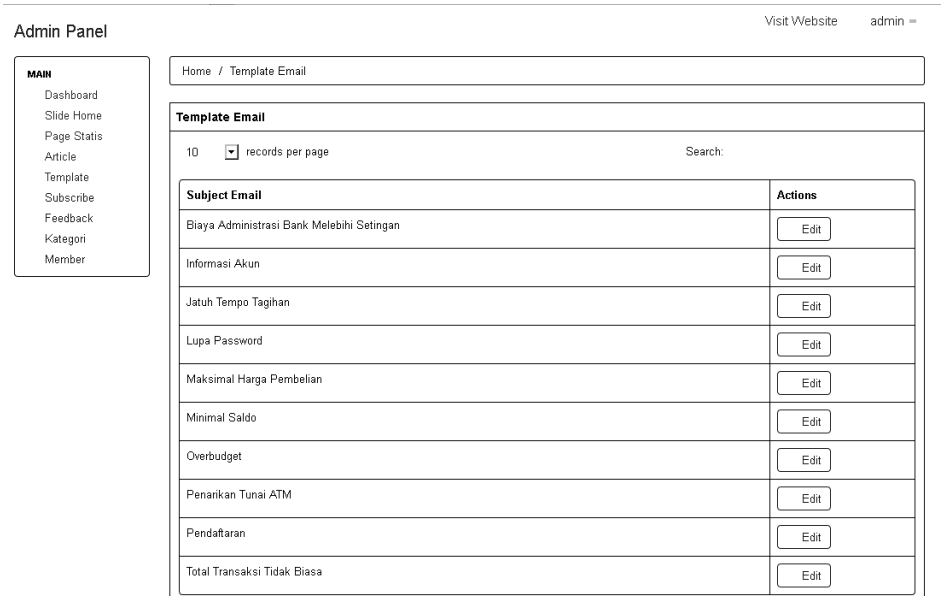
    Halaman ini berguna untuk mengedit halaman artikel, dengan menggunakan *text editor*.



Gambar 3.22 Rancangan Halaman *Edit Article*

13. Halaman *Template Email*

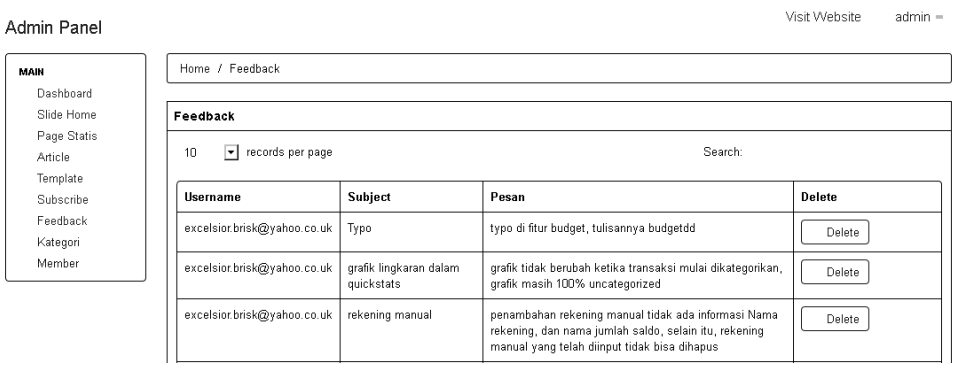
    Halaman ini berguna untuk mengedit *template email.*



Gambar 3.23 Rancangan Halaman *Template Email*

14. Halaman *Feedback*

    Halaman ini memberikan *admin* hak akses untuk melihat dan menghapus *feedback* yang ada.



Gambar 3.24 Rancangan Halaman *Feedback*

15. Halaman Kategori

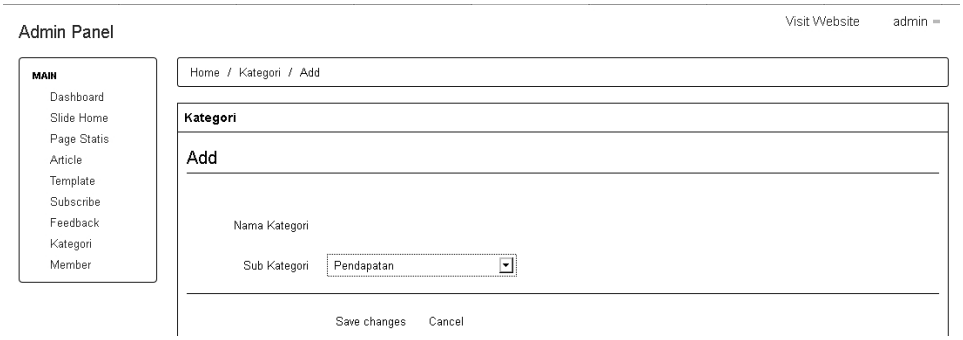
    Halaman ini memudahkan *admin* untuk mengolah kategori.



Gambar 3.25 Rancangan Halaman Kategori

16. Halaman *Add* Kategori

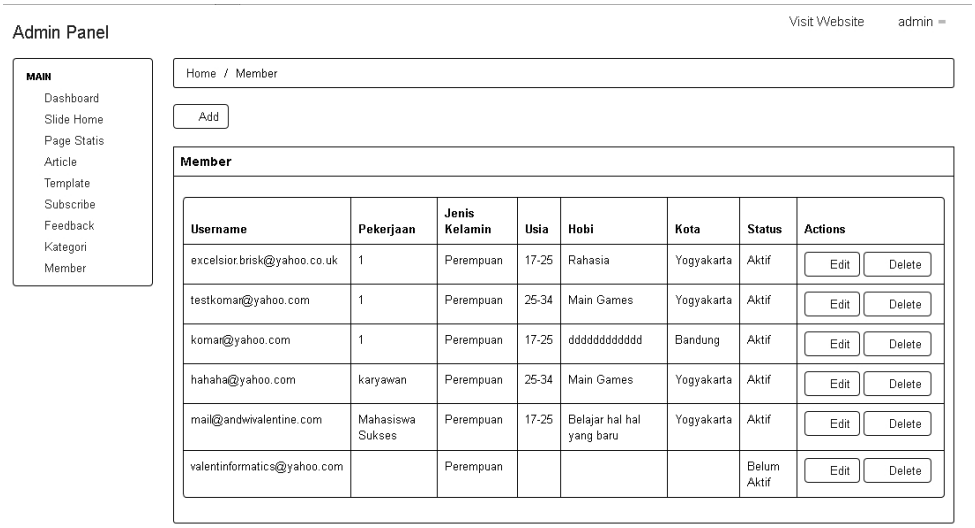
    Halaman ini berguna untuk menambahkan kategori .



Gambar 3.26 Rancangan Halaman *Add* Kategori

17. Halaman *Member*

    Halaman ini menampilkan data-data *user* yang terdaftar dalam aplikasi, sehingga *admin* dapat mengetahui semua data *user*.



Gambar 3.27 Rancangan Halaman Member

18. Halaman Ubah *Password*

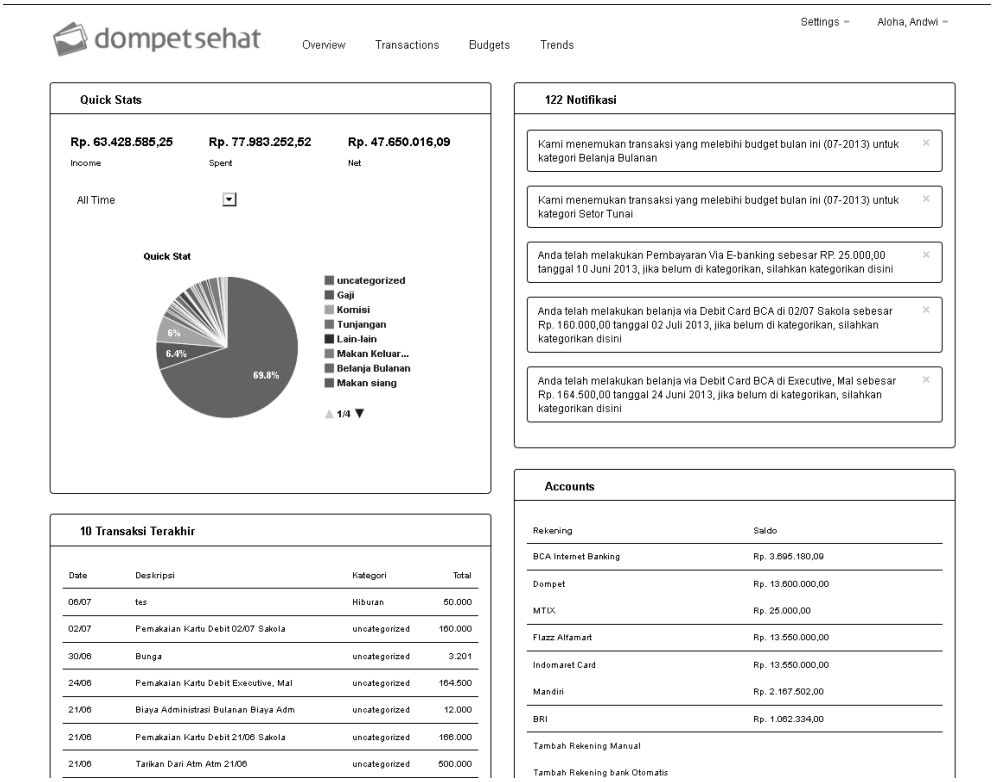
    Halaman ini digunakan untuk merubah *password admin.*



Gambar 3.28 Rancangan Halaman Ubah *Password*

19. Halaman *Dashboard User*

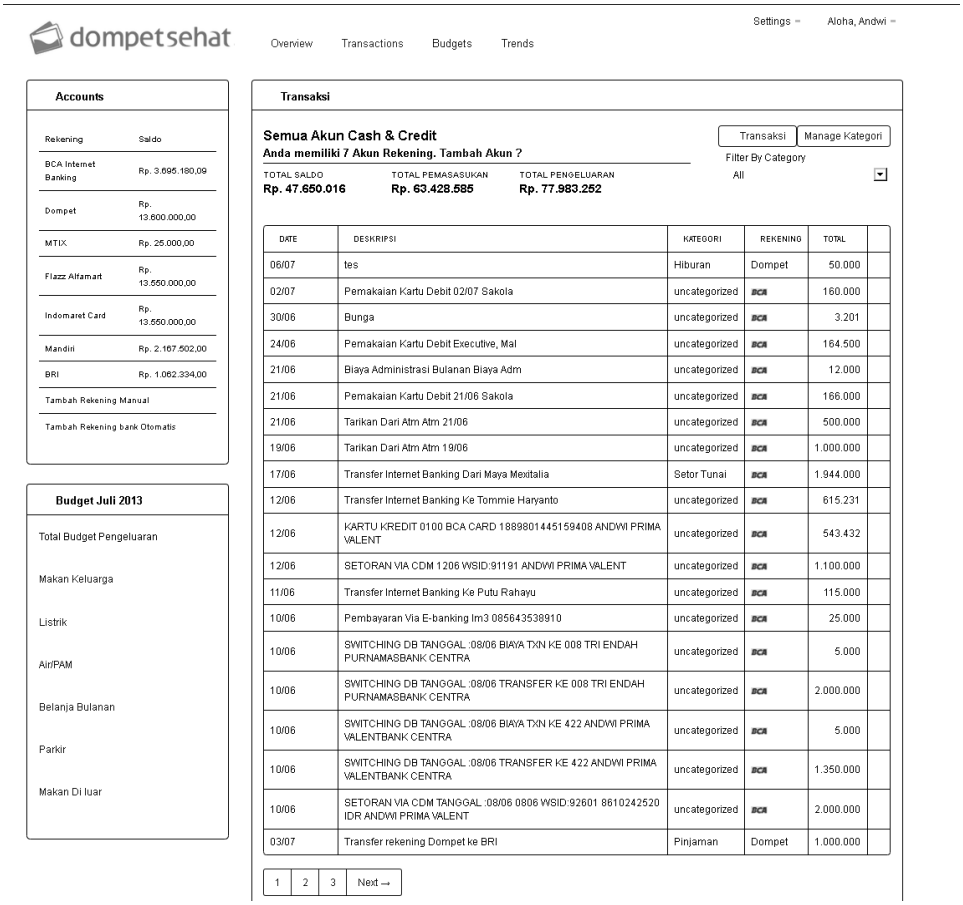
    Halaman ini adalah halaman utama bagi *user* yang sudah melakukan login. Terdiri dari *overview* dari semua fitur dan beberapa link ke halaman lainnya.



Gambar 3.29 Rancangan Halaman *Dashboard User*

20. Halaman *Transactions*

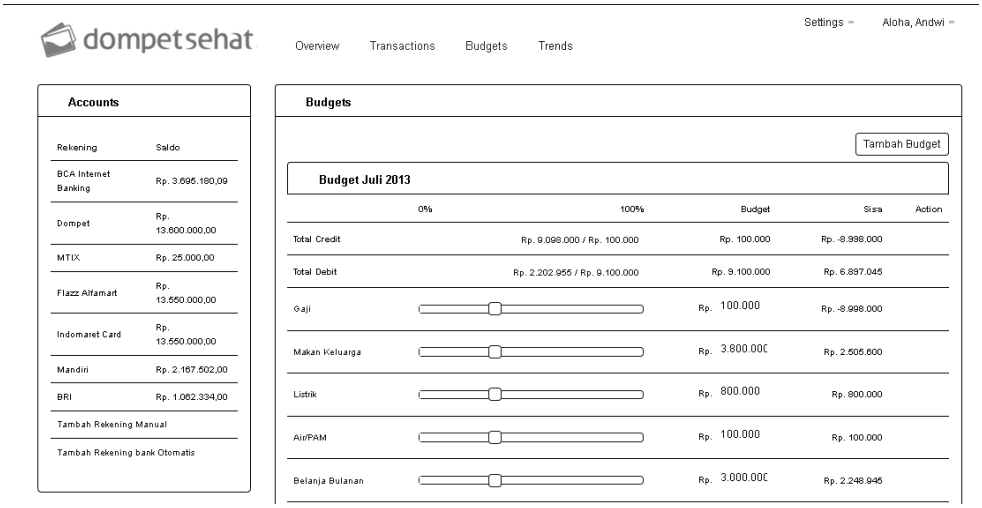
Halaman ini merupakan halaman fitur transaksi dimana user bisa menambahkan transaksi secara manual atau mengedit transaksi yang sudah di aggregasi.



Gambar 3.30 Rancangan Halaman *Transactions*

21. Halaman *Budgets*

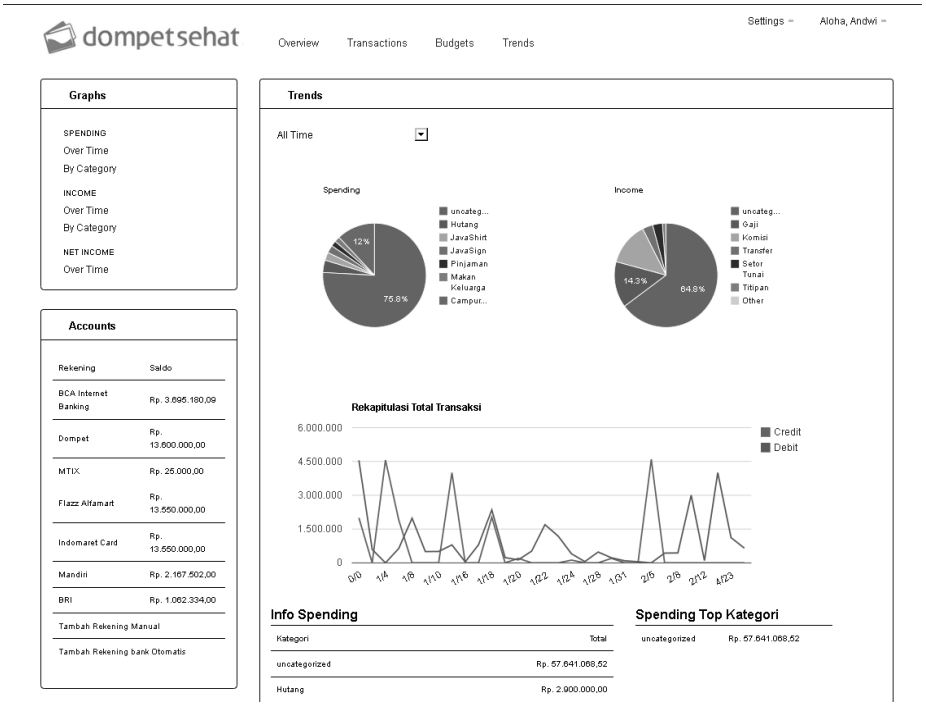
Halaman ini adalah *budgets* dimana user bisa menambahkan *budget*, mengedit dan melihat *budget* secara detail.



Gambar 3.31 Rancangan Halaman *Budgets*

22. Halaman *Trends*

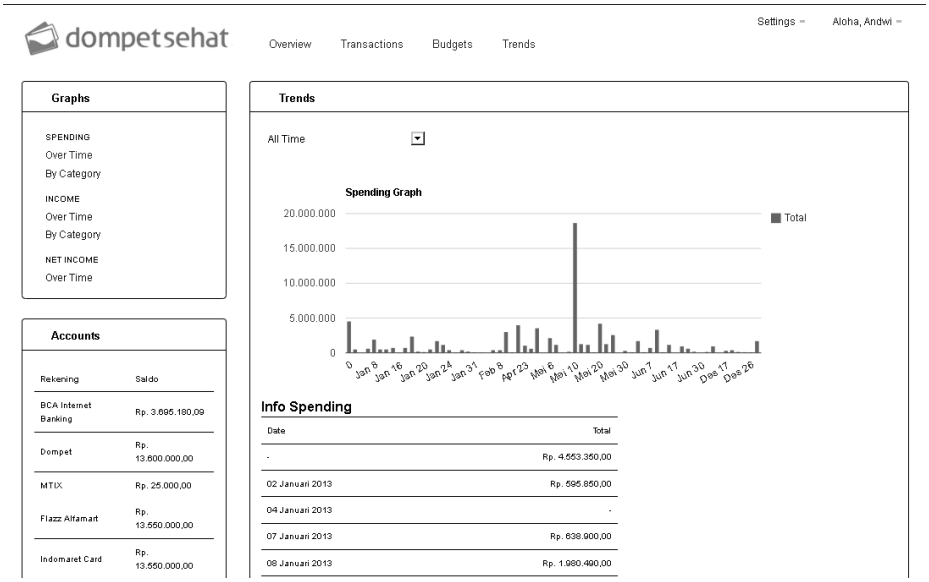
Halaman ini menampilkan informasi tentang *Trends*.



Gambar 3.32 Rancangan Halaman *Trends*

23. Halaman *Trends By Overtime*

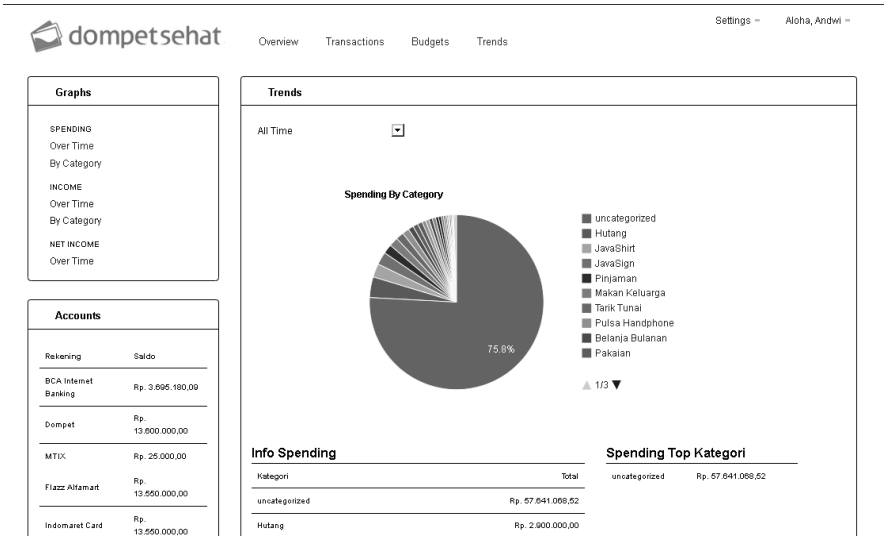
    Halaman ini merupakan halaman *trends* yang di *filter* berdasarkan waktu



Gambar 3.33 Rancangan Halaman *Trends by overtime*

24. Halaman *Trends By Category*

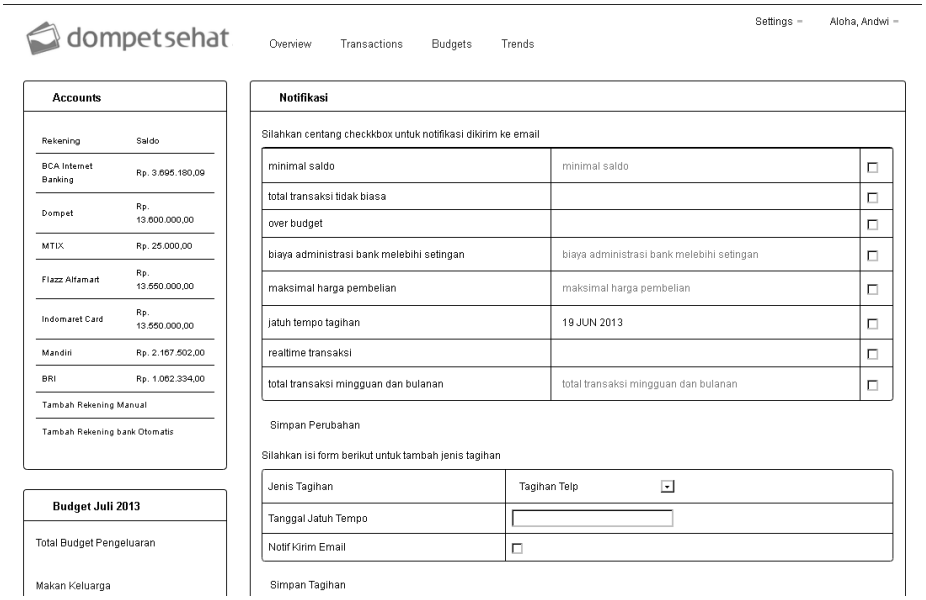
    Halaman ini merupakan halaman *trends* yang di *filter* berdasarkan kategori



Gambar 3.34 Rancangan Halaman *Trends by category*

25. Halaman Notifikasi

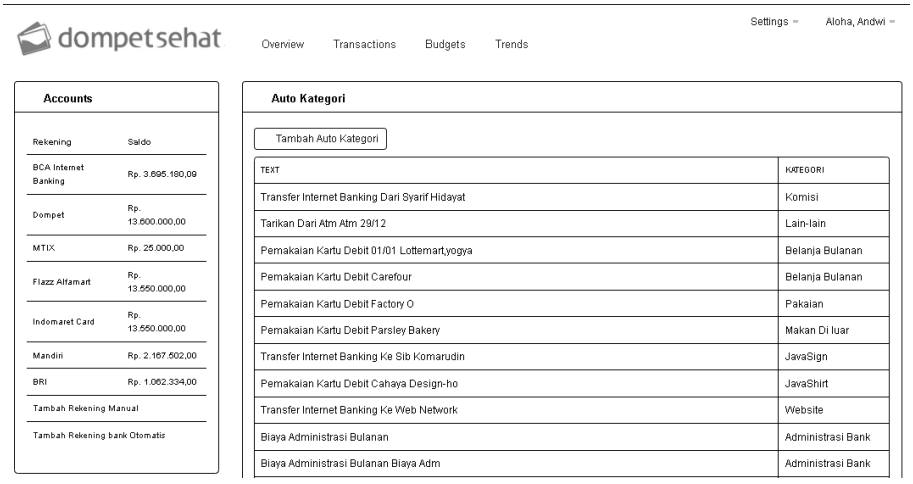
Halaman ini menyediakan *setting* pada notifikasi.



Gambar 3.35 Rancangan Halaman Notifikasi

26. Halaman Auto Kategori

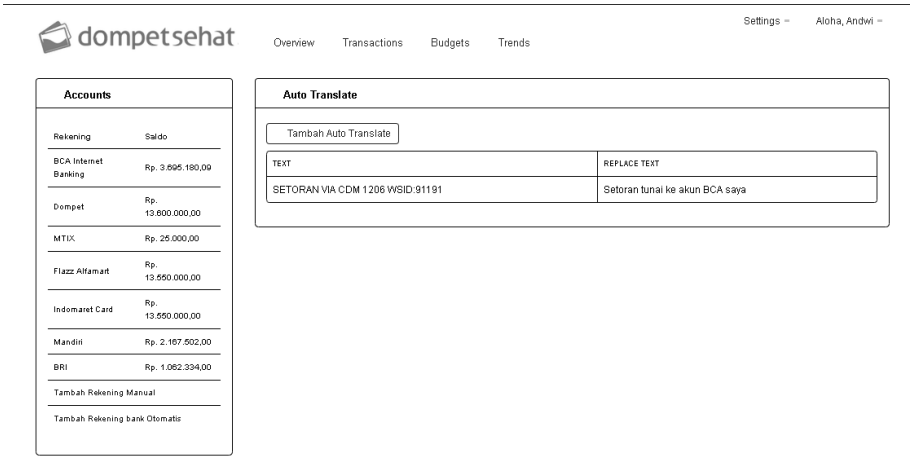
Halaman ini merupakan halaman untuk meginputkan transaksi yang akan di kategorikan secara otomatis



Gambar 3.36 Rancangan Halaman Auto Kategori

27. Halaman *Auto Translate*

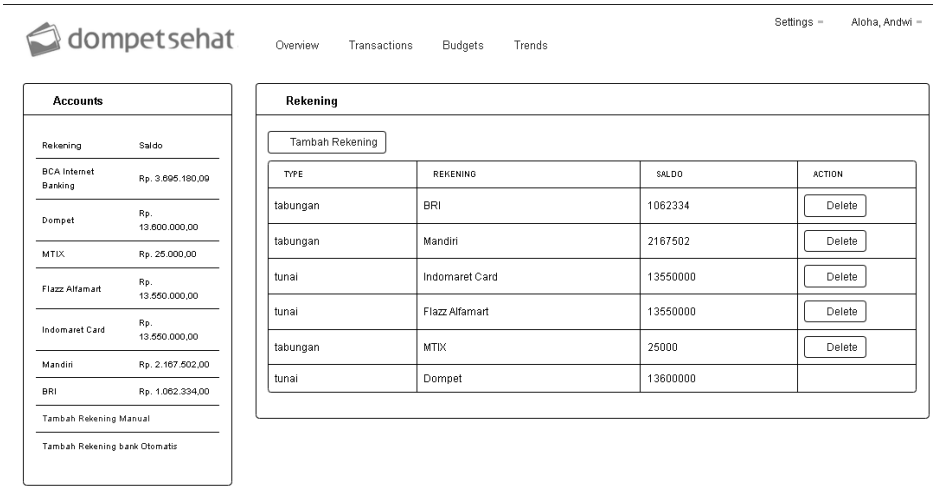
    Halaman ini merupakan halaman untuk mengiputkan deskripsi yang berasal dari bank akan di *translate* secara otomatis



Gambar 3.37 Rancangan Halaman *Auto Translate*

28. Halaman Rekening

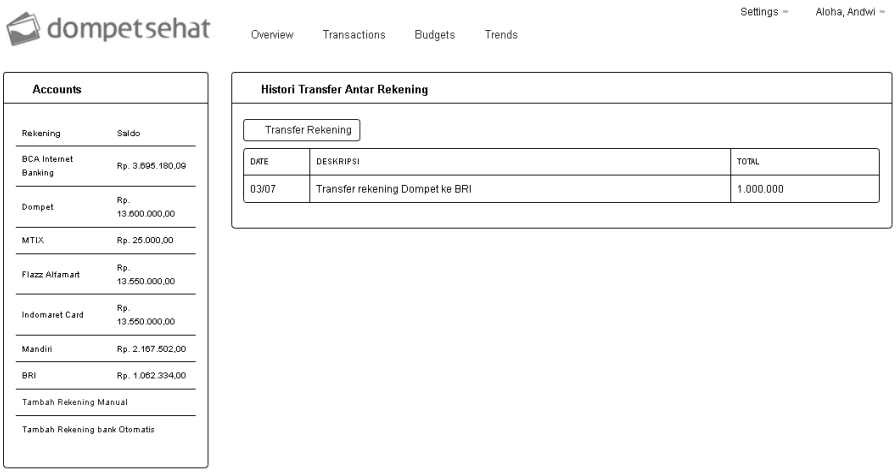
    Halaman ini merupakan halaman *management* rekening manual.



Gambar 3.38 Rancangan Halaman Rekening

29. Halaman *Transfer* Rekening

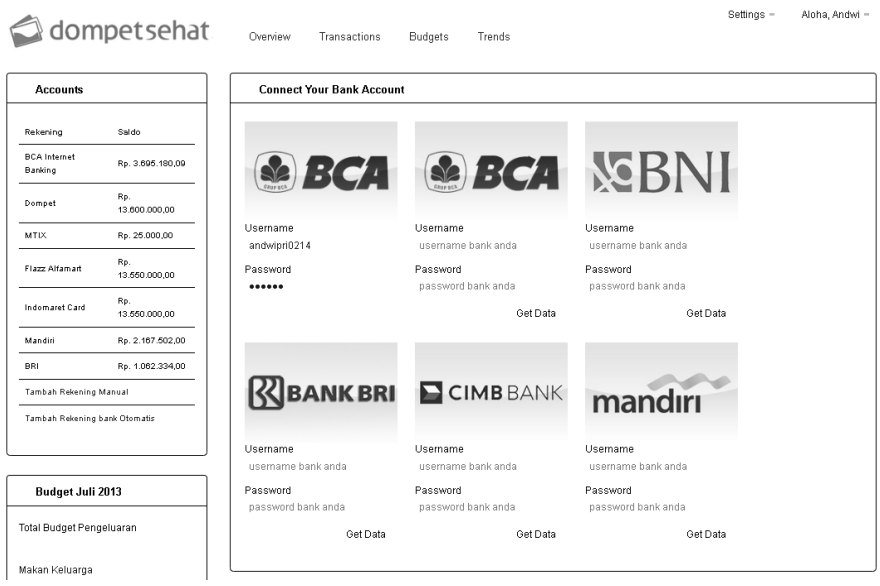
    Halaman ini merupakan halaman *transfer* dana melalui rekening manual.



Gambar 3.39 Rancangan Halaman *Transfer* Rekening

30. Halaman *Connect your Bank*

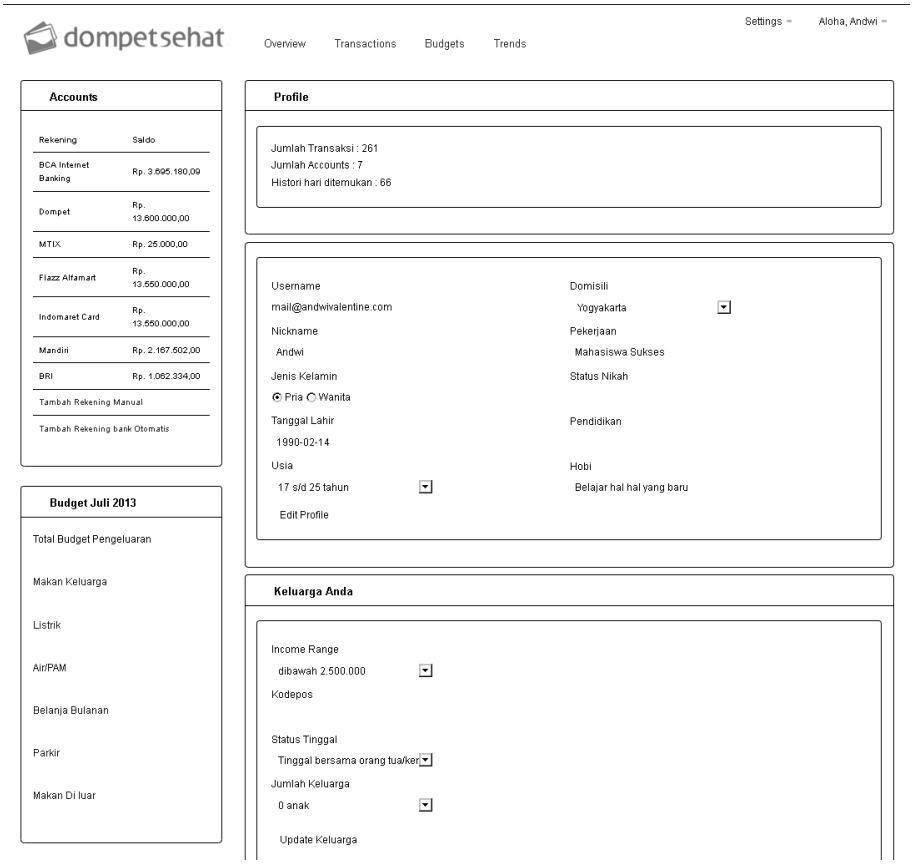
    Halaman ini merupakan halaman aggregasi ke *Bank Central Asia.*



Gambar 3.40 Rancangan Halaman *Connect your Bank*

31. Halaman *Profile*

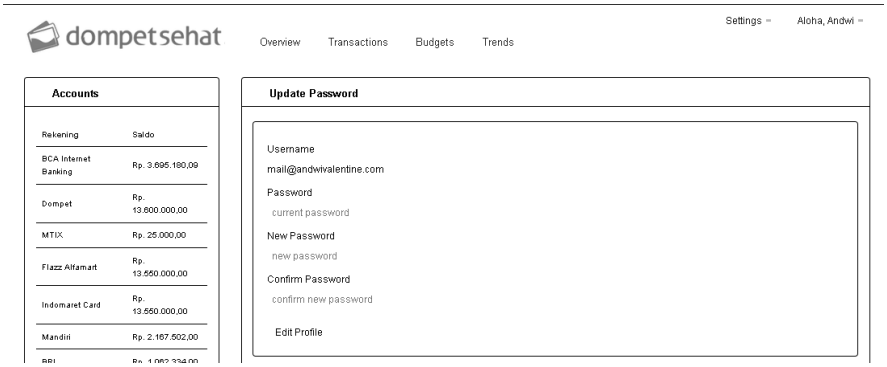
Halaman ini merupakan halaman *profile* dari *user*.



Gambar 3.41 Rancangan Halaman *Profile*

32. Halaman *Password*

Halaman ini merupakan *Update password.*



Gambar 3.42 Rancangan Halaman *Update Password*

BAB IV
HASIL DAN PEMBAHASAN

Dalam bab ini akan dibahas bagian pengembangan (*development*) dalam *grapple*. Tahap implementasi ini dilakukan setelah tahap perancangan dengan tujuan mempermudah menjabarkan perangkat lunak. Pada tahap implementasi akan menjelaskan tentang perangkat keras dan perangkat lunak pendukung sistem, dan tampilan antarmukanya.

4.1 Perangkat Keras

Perangkat keras yang digunakan dalam pembangunan aplikasi ini adalah sebuah *Notebook ASUS* dengan spesifikasi sebagai berikut :

1. Intel Core i5
2. Memori 2 GB
3. Harddisk 500 GB

4.2 Perangkat Lunak

Perangkat lunak yang digunakan dalam pembangunan aplikasi ini adalah sebagai berikut :

1. Microsoft Windows 8
2. XAMPP 1.8.2
3. PHP 5.4.2
4. MySQL 5.0.37
5. Rational Rose 2002
6. Notepad ++

4.3 Tampilan Antarmuka

Pada implementasi antarmuka ini terdapat tiga sisi antarmuka, yaitu :

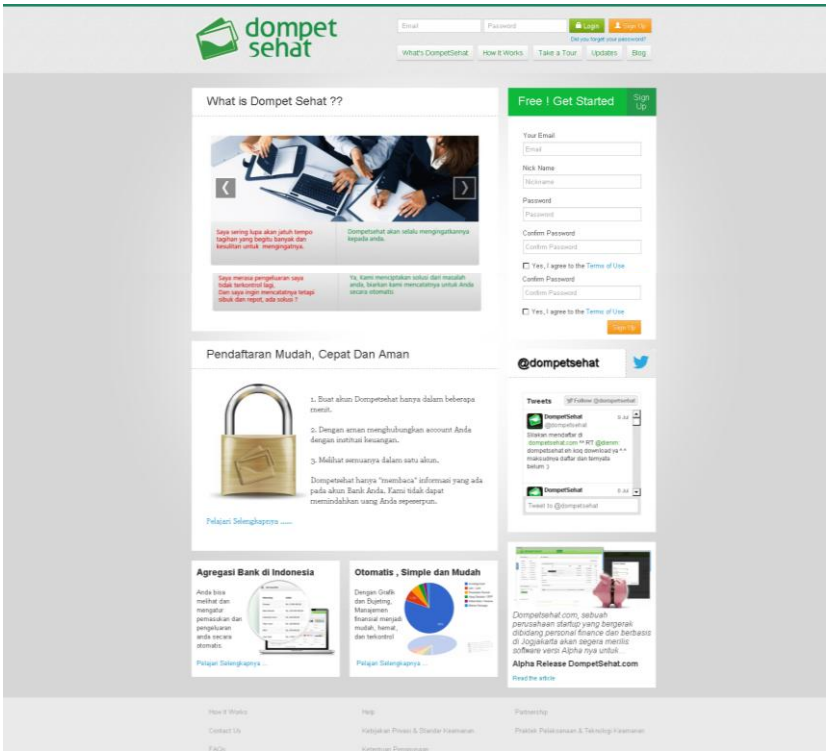
1. Halaman antarmuka *visitor*
2. Halaman antarmuka *user*
3. Halaman antarmuka *administrator*

4.3.1 Halaman Antarmuka Visitor

Halaman antarmuka *visitor* ini merupakan suatu halaman yang dapat diakses oleh siapa saja. Halaman ini terdiri dari beberapa tautan halaman yang mengarah ke halaman lainnya. Halaman yang termasuk dalam halaman visitor antara lain :

1. Halaman *Homepage*

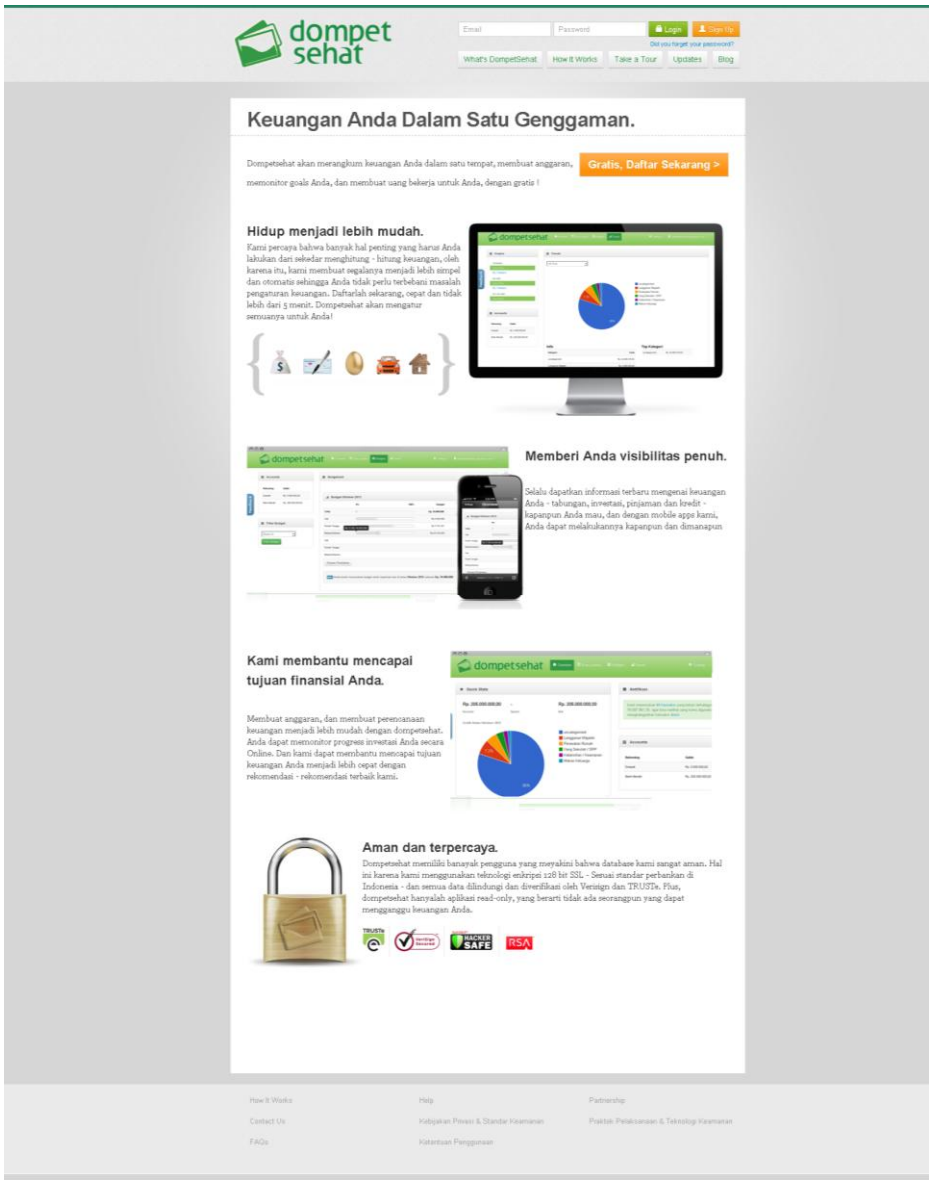
   Halaman ini adalah halaman pertama yang dikunjungi oleh visitor untuk mengetahui lebih dalam tentang aplikasi DompetSehat. Halaman *homepage* ini dapat dilihat pada gambar 4.1 dibawah ini.



Gambar 4.1 Halaman *Homepage*

2. Halaman *What's DompetSehat*

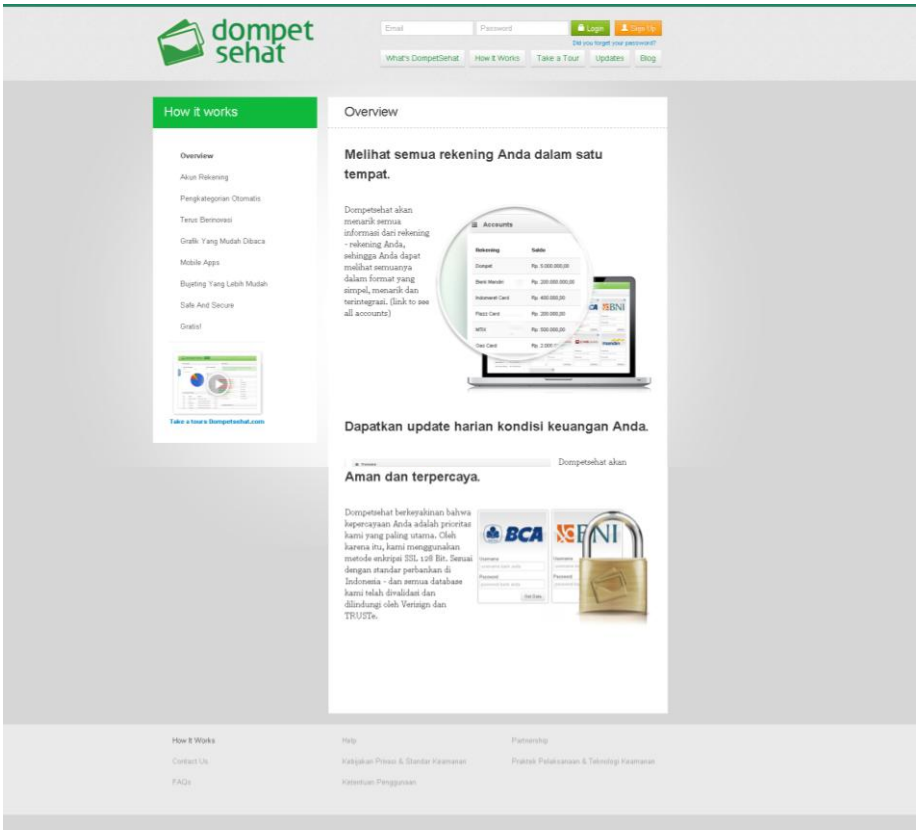
   What's DompetSehat adalah halaman informasi tentang apa itu aplikasi DompetSehat. Halaman *What's DompetSehat* ini dapat dilihat pada gambar 4.2 dibawah ini.



Gambar 4.2 Halaman *What's Dompetsehat*

3. Halaman *How it works*

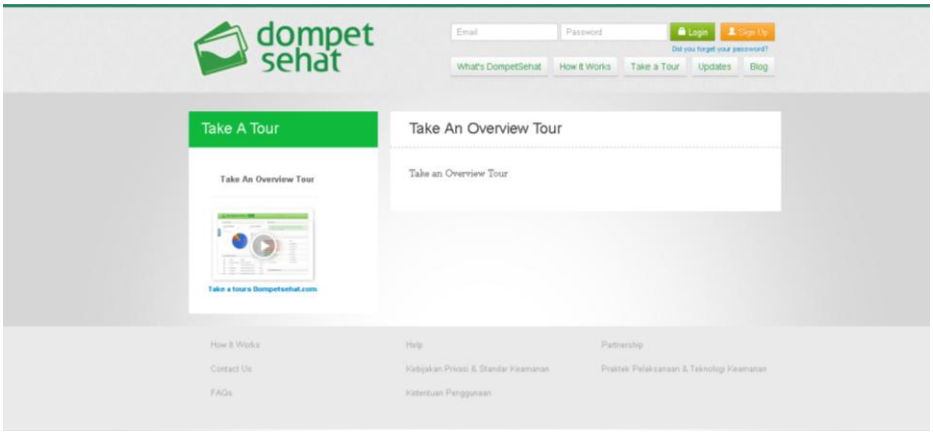
   Halaman *How it works* adalah halaman yang menampilkan informasi bagaimana aplikasi dompetsehat bekerja, terdiri dari beberapa sub halaman, antara lain : Overview, Akun rekening, Pengkategorian otomatis, Terus Berinovasi, Grafik yang mudah dibaca, Bujeting yang lebih mudah, Safe and Secure, Gratis. Halaman *How it works* ini dapat dilihat pada gambar 4.3 dibawah ini.



Gambar 4.3 Halaman *How it works*

4. Halaman *Take a tour*

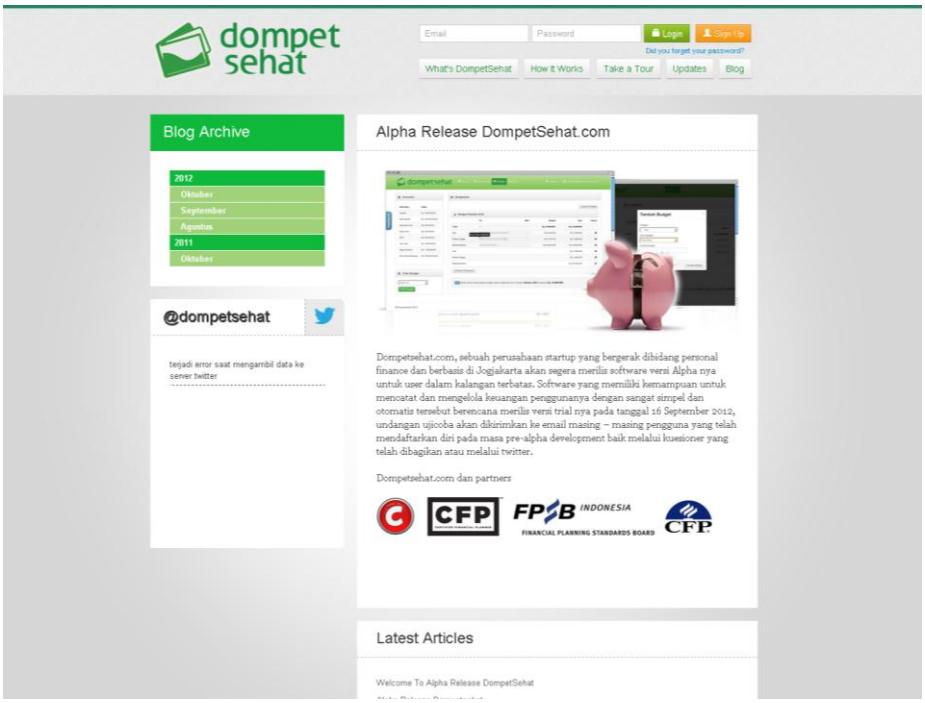
   Halaman *Take a tour* adalah halaman informasi mengenai demo aplikasi, sehingga *user* yang belum melakukan registrasi bisa melihat halaman *user*. Halaman *Take a tour* dapat dilihat pada gambar 4.4 di bawah ini.



Gambar 4.4 Halaman *Take a tour*

5. Halaman *Updates dan Blog*

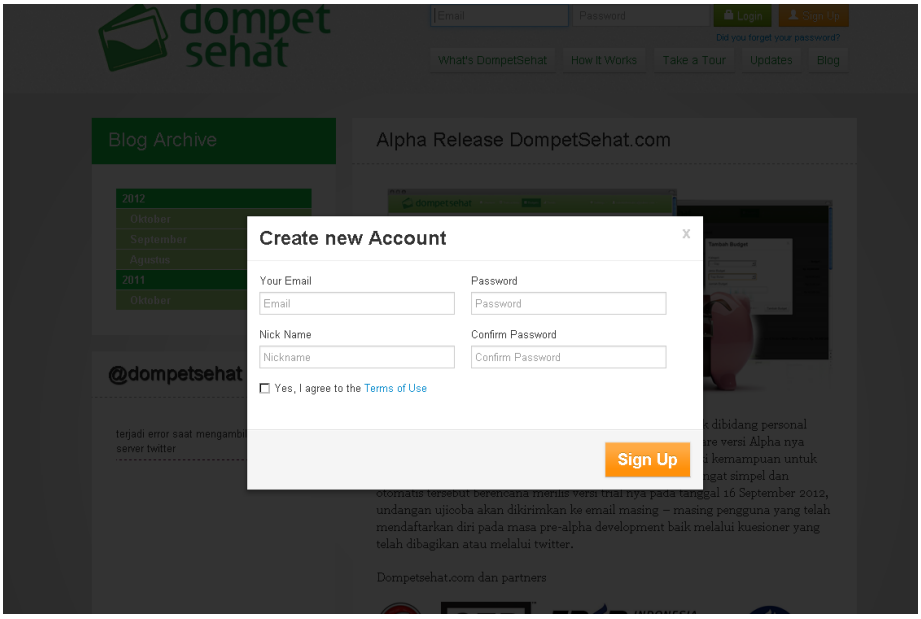
   Halaman *Updates dan Blog* merupakan halaman yang berisi informasi mengenai update terbaru tentang aplikasi DompetSehat dan Berisi blog yang menampilkan artikel. Halaman *Updates dan Blog* dapat dilihat pada gambar 4.5 dibawah ini.



Gambar 4.5 Halaman *Updates dan Blog*

6. Halaman *Sign Up*

   Halaman *Sign Up* merupakan sebuah halaman yang berisi *form* registrasi untuk menjadi *user*. Halaman *Sign Up* dapat dilihat pada gambar 4.6 di bawah ini.



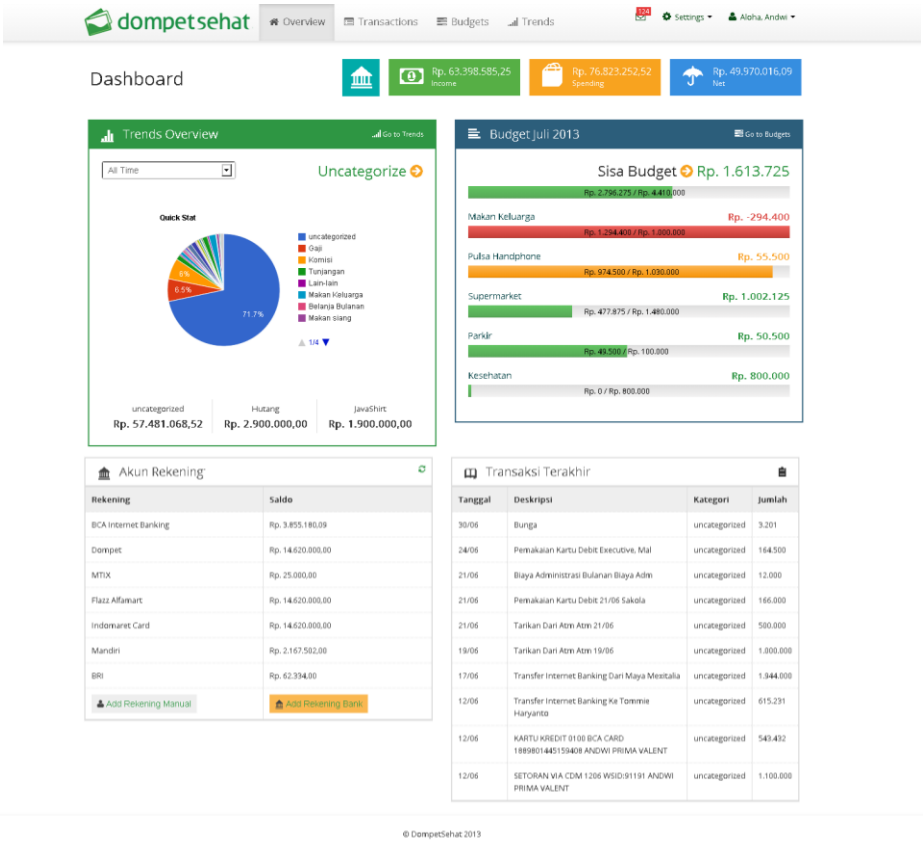
Gambar 4.6 Halaman *Sign Up*

4.3.2 Halaman Antarmuka *User*

Halaman antarmuka *user* ini merupakan suatu halaman yang dapat diakses oleh *User* saja. Pada halaman ini mempunyai beberapa halaman yang dapat digunakan *user* untuk mencoba fitur-fitur dari Aplikasi. Halaman yang termasuk dalam halaman *user* antara lain :

1. Halaman *Overview*

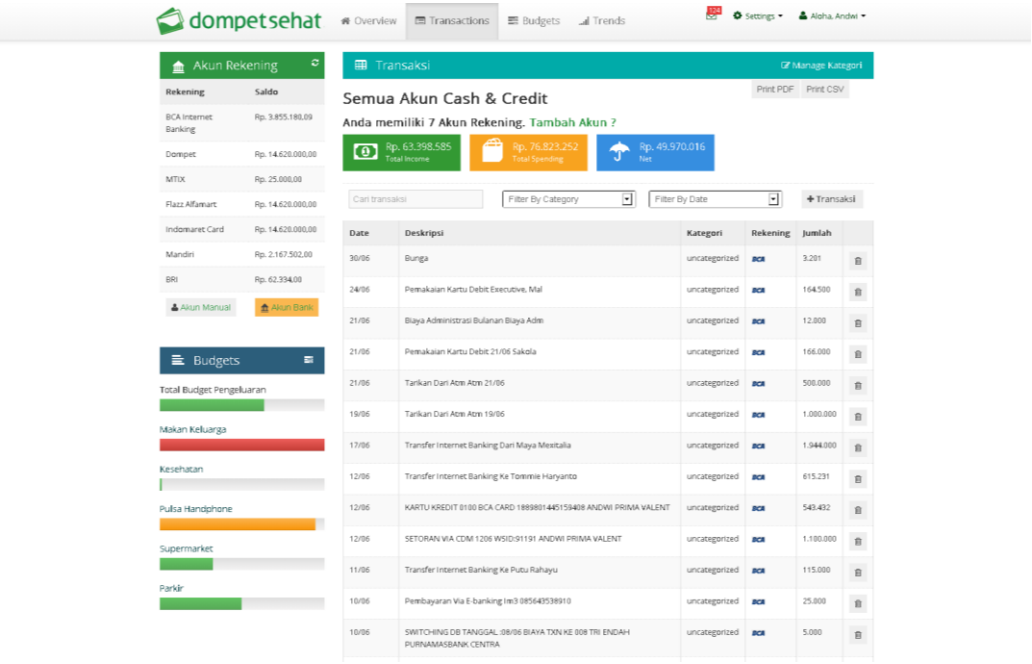
   Halaman *Overview* merupakan halaman pertama yang akan diakses oleh *user* setelah *user* login ke aplikasi. Halaman ini terdiri dari beberapa singkatan dari semua fitur dan beberapa tautan ke halaman yang lain. Halaman *Overview* ini dapat dilihat pada gambar 4.7 di bawah ini.



Gambar 4.7 Halaman *Overview*

2. Halaman *Transaction*

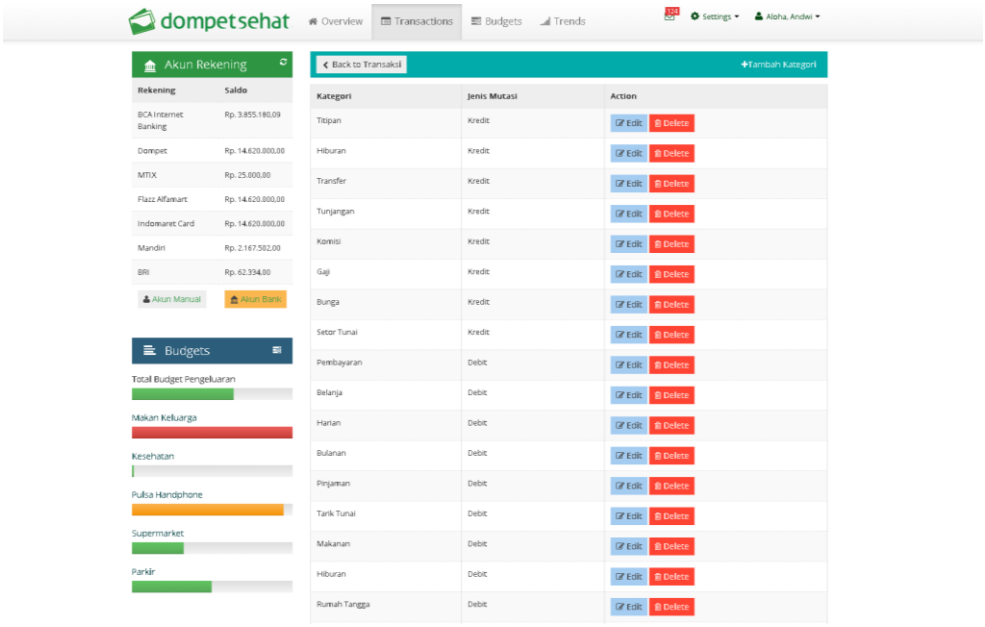
   Halaman *Transaction* adalah yang berisi fitur transaksi, fitur-fitur yang terdapat pada transaksi antara lain, Informasi saldo, *Search* transaksi, *Filter by Category*, *Filter by Date*, Tambah Transaksi, *Print Pdf* dan *Print CSV* dan link menuju halaman *manage category*. Pada bagian sidebar kiri terdapat *widgets* akun rekening dan *budgets*. Halaman *Transaction* ini dapat dilihat pada gambar 4.8 di bawah ini.



Gambar 4.8 Halaman *Transaction*

3. Halaman *Manage Category*

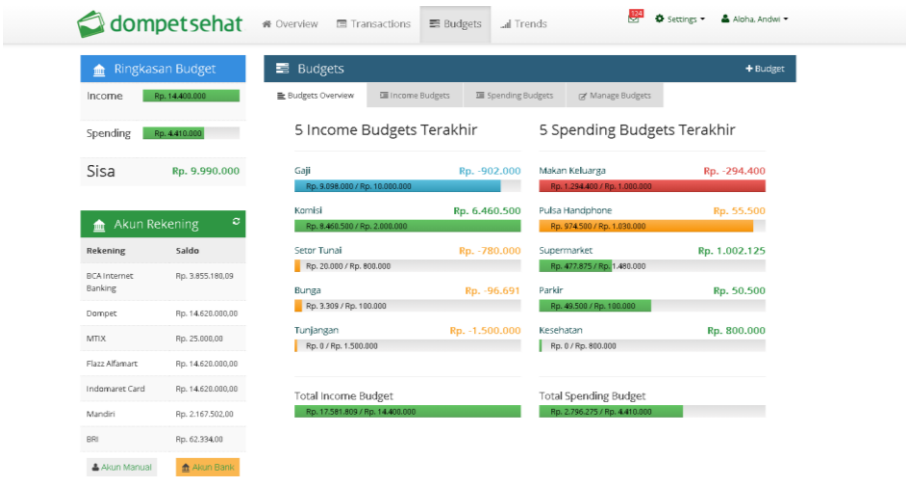
   Halaman *Manage Category* ini digunakan untuk mengolah kategori. User dapat menambahkan atau mengedit kategori. Halaman *Manage Category* dapat dilihat pada gambar 4.9 di bawah ini.



Gambar 4. 9 Halaman *Manage Category*

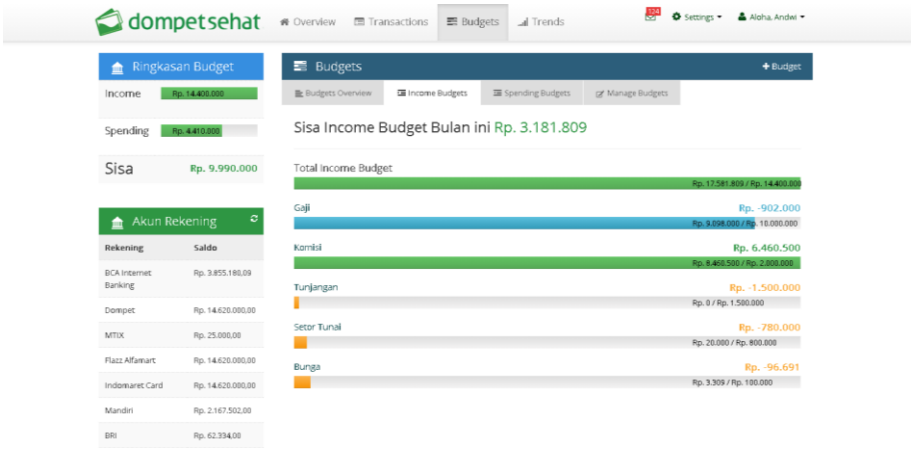
4. Halaman *Budgets*

Halaman *Budgets* adalah halaman fitur *budget* yang memiliki dua *widgets* yaitu Ringkasan *Budget* dan Akun Rekening serta memiliki beberapa bagian fitur yang dibatasi dengan *tab* diantaranya *Budgets Overview*, *Income Budgets*, *Spending Budgets*, dan *Manage Budgets*. *Tab Budgets Overview* menampilkan 5 *income budgets* terakhir dan 5 *spending budgets* terakhir dan juga total masing masing *budget*. *Tab Budgets Overview* dapat dilihat pada gambar 4.10 di bawah ini.



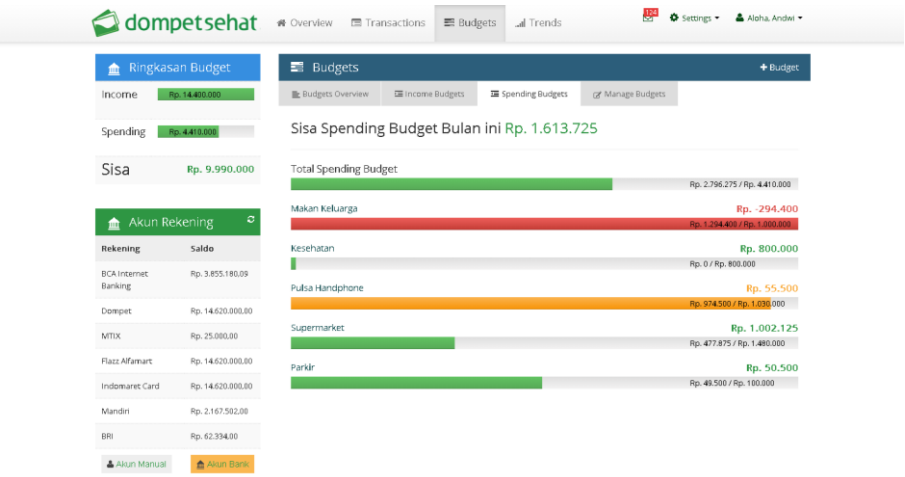
Gambar 4.10 Halaman *Budgets Overview*

Tab Income Budgets menampilkan seluruh budgets pemasukan yang telah di buat. *Tab Income Budgets* dapat dilihat pada gambar 4.11 di bawah ini.



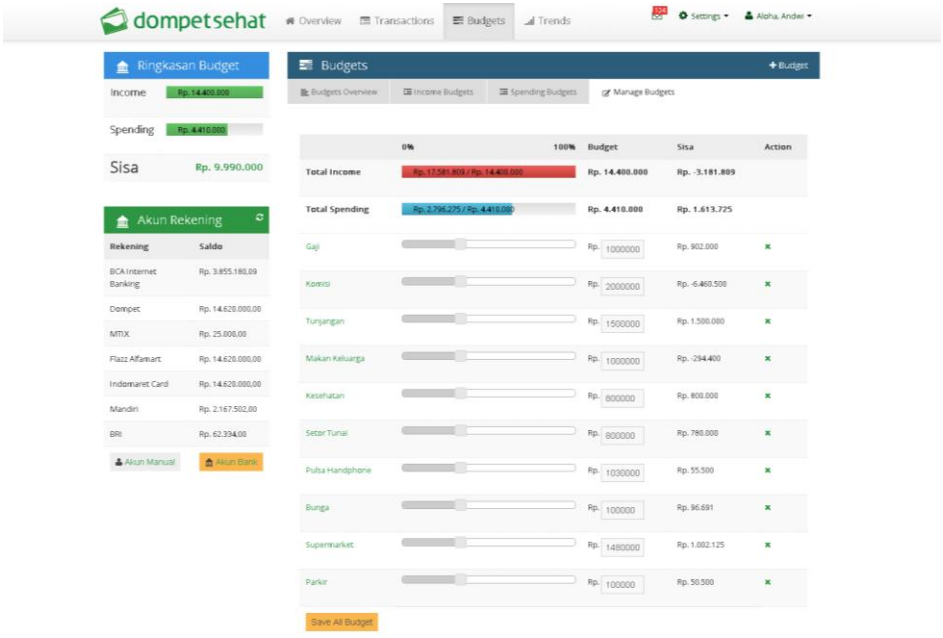
Gambar 4.11 Halaman *Income Budgets*

Tab Spending Budgets menampilkan seluruh budgets pengeluaran yang telah di buat. *Tab Spending Budgets* dapat dilihat pada gambar 4.12 di bawah ini.



Gambar 4.12 Halaman *Spending Budgets*

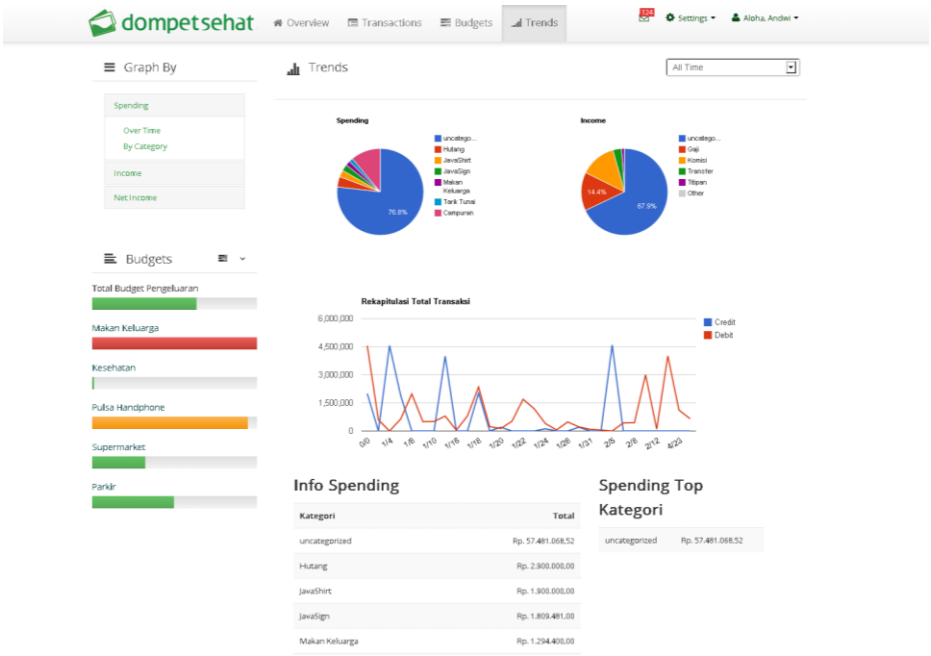
Tab Manage Budgets merupakan fitur untuk mengolah *budgets*, mengedit dan menghapus *budgets*. *Tab Income Budgets* dapat dilihat pada gambar 4.13 di bawah ini.



Gambar 4.13 Halaman *Manage Budgets*

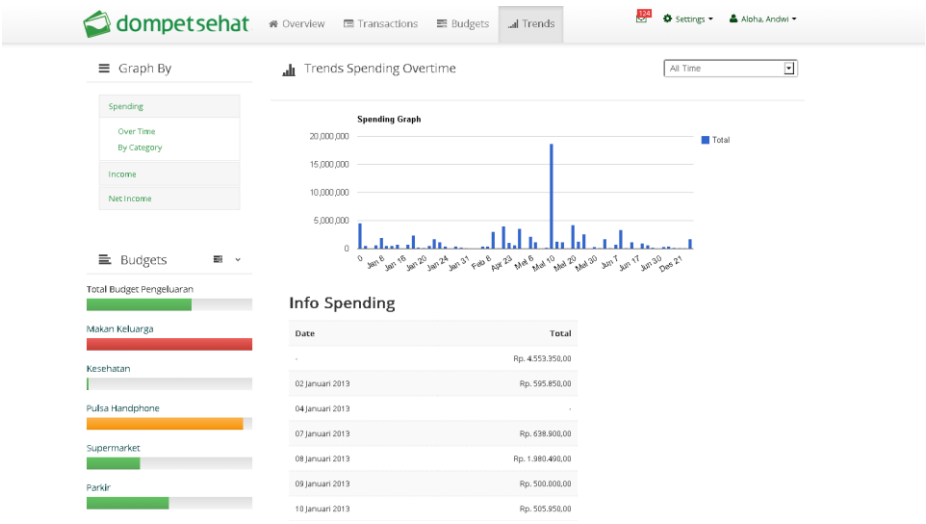
5. Halaman *Trends*

Halaman *Trends* merupakan halaman yang menampilkan detail fitur *trends*, fitur *trends* adalah fitur yang resprensentasikan histori transaksi melalui diagram, baik diagram batang, diagram garis ataupun diagram lingkaran. Fitur ini terbagi menjadi beberapa submenu antara lain *spending by overtime*, *spending by category*, *income by overtime*, *income by categori* dan *net income*. Halaman *Trends Overview* dapat dilihat pada gambar 4.14 di bawah ini.



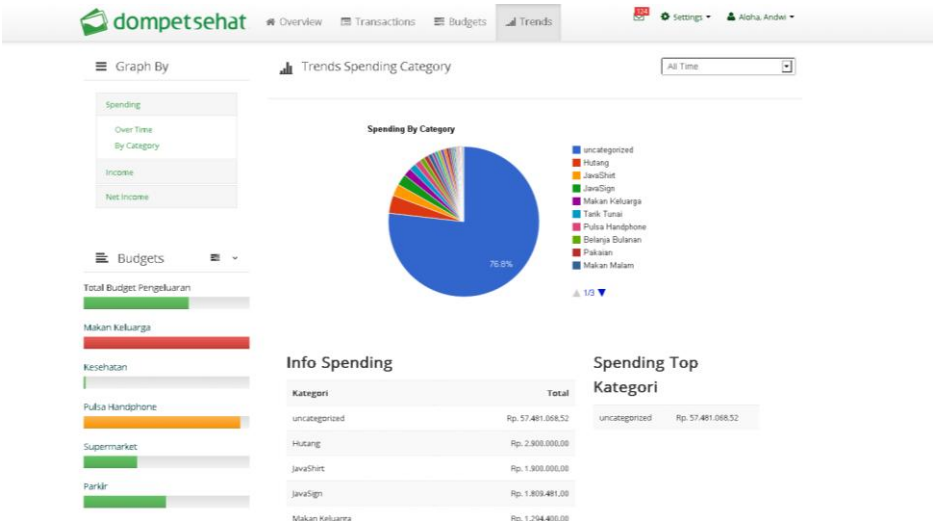
Gambar 4.14 Halaman *Trends*

Halaman *Spending by Overtime* adalah halaman *trends* yang menampilkan histori transaksi pengeluaran berdasarkan waktu dengan menggunakan diagram batang. *Spending by Overtime* dapat dilihat pada gambar 4.15 di bawah ini.



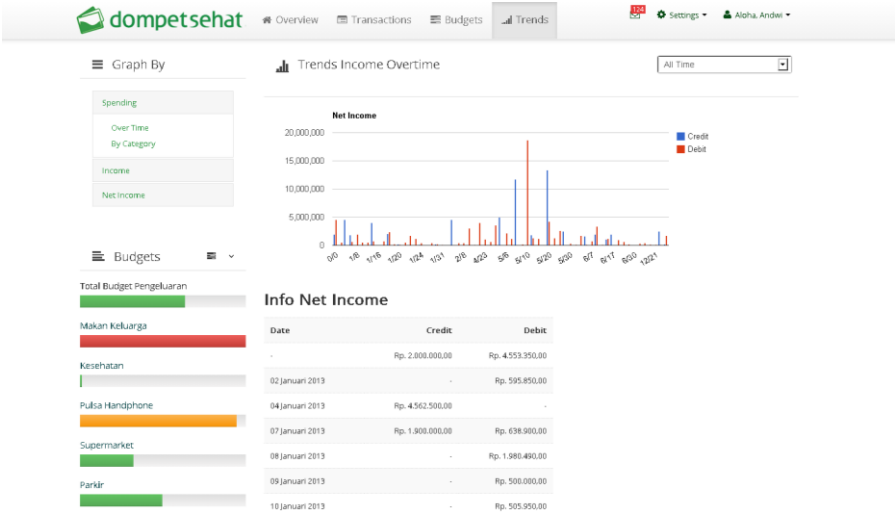
Gambar 4.15 Halaman *Spending by Overtime*

Halaman *Spending by Category* adalah halaman *trends* yang menampilkan histori transaksi pengeluaran berdasarkan kategori dengan menggunakan diagram lingkaran. *Spending by Category* dapat dilihat pada gambar 4.16 di bawah ini.



Gambar 4.16 Halaman *Spending by Category*

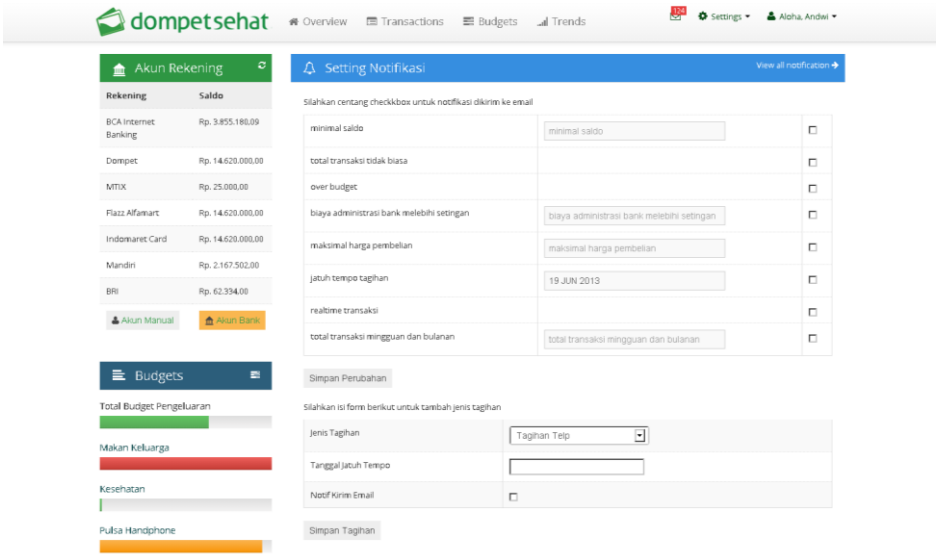
Halaman *Net Income* menampilkan perbandingan histori transaksi pengeluaran dan pemasukan berdasarkan waktu dengan menggunakan diagram batang. Halaman *Net Income* dapat dilihat pada gambar 4.17 di bawah ini.



Gambar 4.17 Halaman *Net Income*

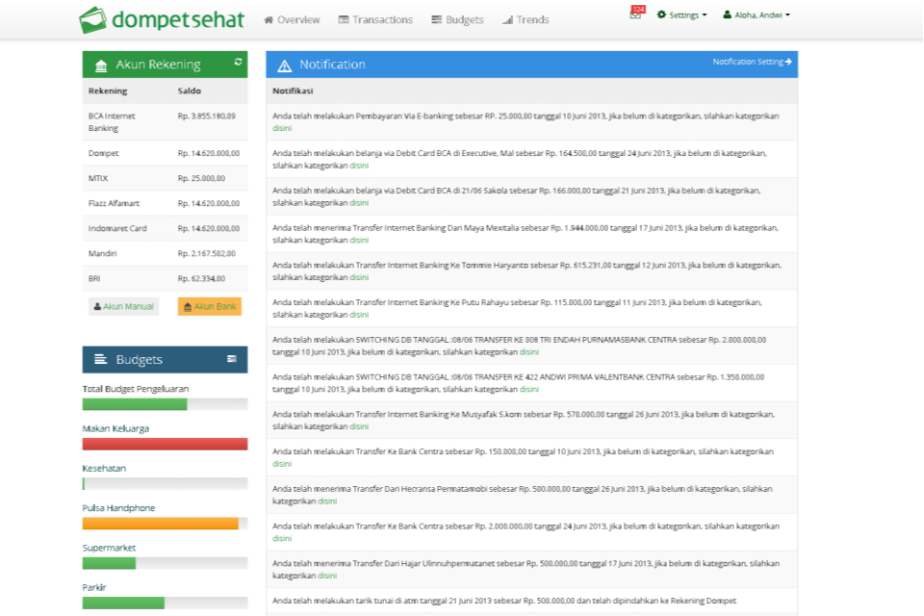
6. Halaman Notifikasi

Halaman Notifikasi merupakan halaman untuk mengatur notifikasi. Halaman ini memiliki tautan ke halaman *all notification* untuk melihat semua notifikasi. Halaman Notifikasi dapat dilihat pada gambar 4.18 di bawah ini.



Gambar 4.18 Halaman Notifikasi

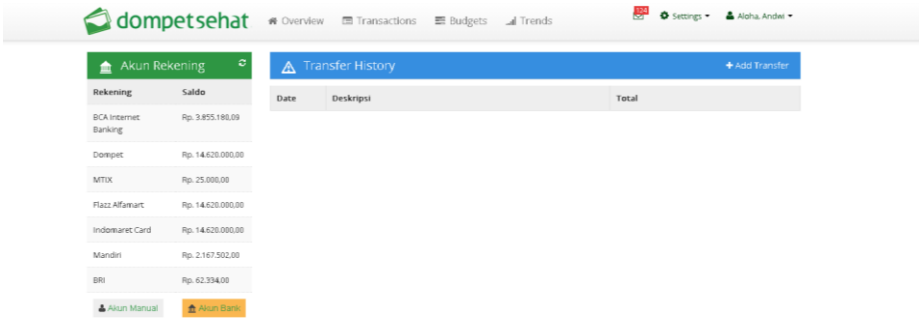
Halaman *All Notification* merupakan halaman yang menampilkan semua notifikasi yang masuk. Halaman *All Notification* ini dapat dilihat pada gambar 4.19 di bawah ini.



Gambar 4.19 Halaman *All Notification*

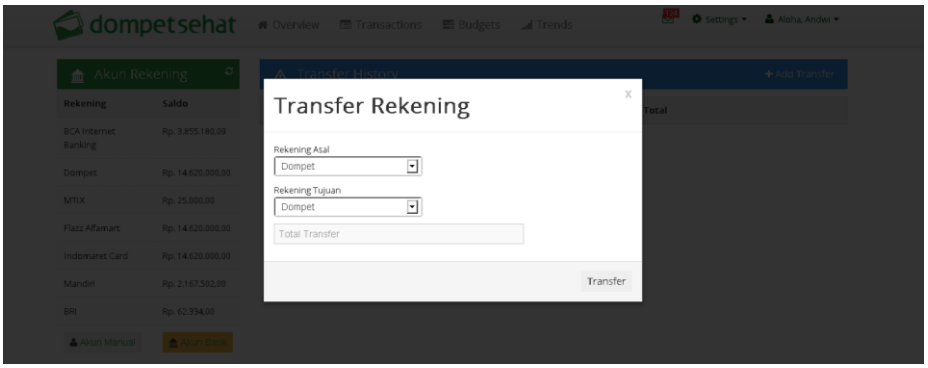
7. Halaman *Transfer History*

   Halaman *Transfer History* merupakan halaman yang memberikan *user* fasilitas *transfer* antar rekening manua. Pada halaman ini juga terdapat tautan *Add transfer* untuk menambahkan *transfer dana* . Halaman *Transfer History* ini dapat dilihat pada gambar 4.20 di bawah ini.



Gambar 4.20 Halaman *Transfer History*

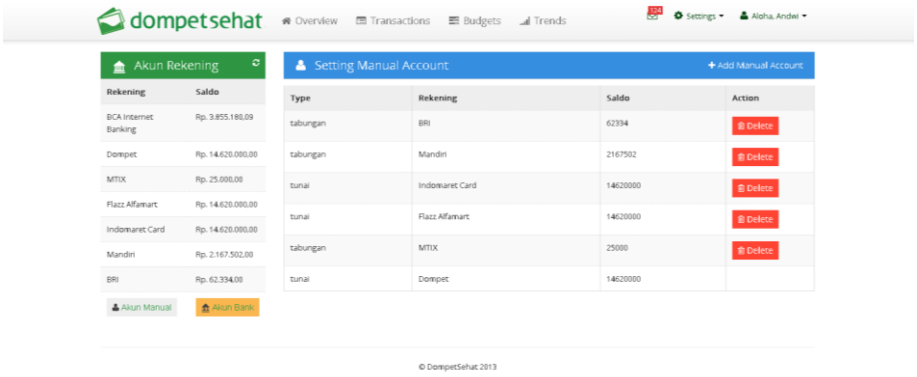
Halaman *Add Transfer* merupakan halaman untuk menambahkan tranfer dana. Halaman *Add Transfer* dapat dilihat pada gambar 4.21 di bawah ini.



Gambar 4.21 Halaman *Add Transfer*

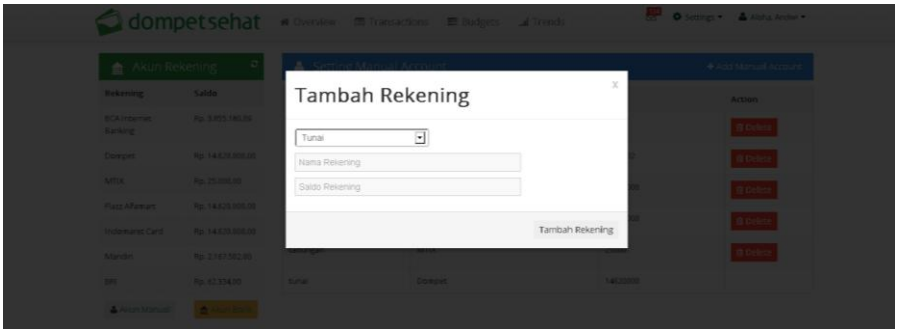
8. Halaman Rekening

Halaman Rekening ini berisi mengenai daftar rekening manual yang telah diinputkan oleh *user*. Halaman Rekening dapat dilihat pada gambar 4.22 dibawah ini.



Gambar 4.22 Halaman Rekening

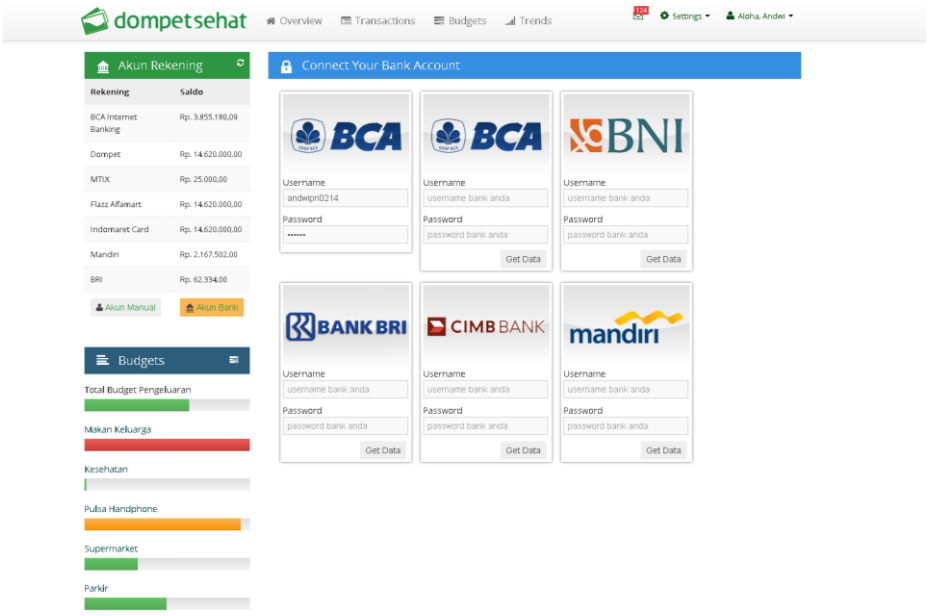
Pada halaman ini user juga bisa menghapus dan menambahkan rekening dengan cara klik tombol *Add Manual Account.* Halaman Tambah Rekening dapat dilihat pada gambar 4.23 dibawah ini.



Gambar 4.23 Halaman Tambah Rekening

9. Halaman *Connect Bank Account*

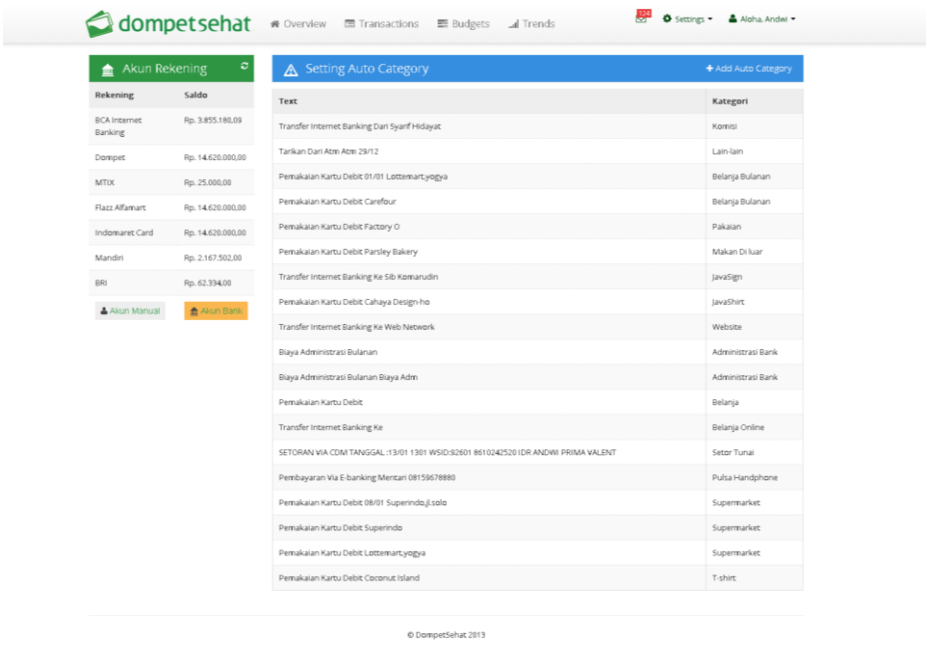
   Halaman *Connect Bank Account* ini berisi form untuk mengkoneksikan akun user dengan akun internet banking. Halaman *Connect Bank Account* dapat dilihat pada gambar 4.24 dibawah ini.



Gambar 4.24 Halaman *Connect Bank account*

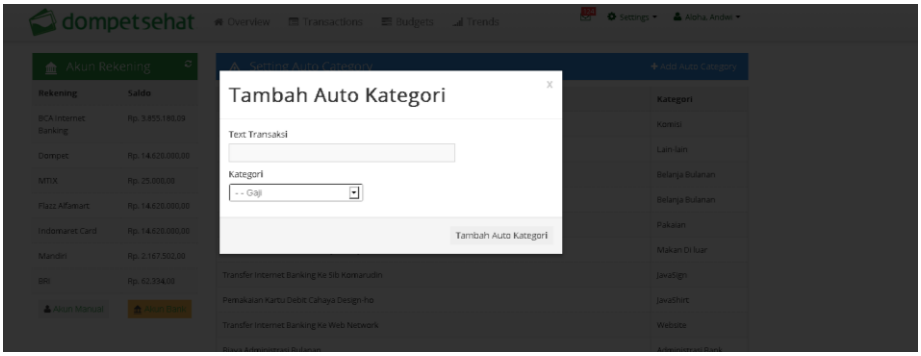
10. Halaman *Setting Auto Category*

    Halaman ini merupakan halaman untuk setting kategori secara otomatis, kategori disesuaikan berdasarkan deskripsi inputan. Halaman *Setting Auto Category* dapat dilihat pada gambar 4.25 dibawah ini.



Gambar 4.25 Halaman *Setting Auto Category*

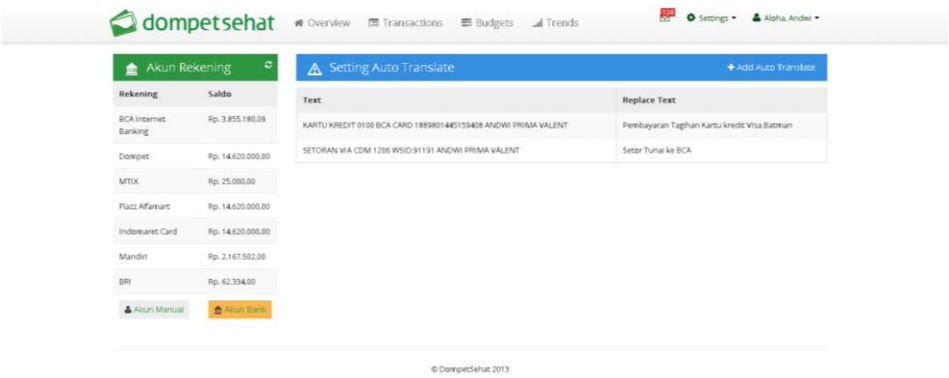
User juga bisa menambahkan auto kategori dengan menekan tombol *Add Auto Category*. Halaman *Add Auto Category* dapat dilihat pada gambar 4.26 dibawah ini.



Gambar 4.26 Halaman *Setting Auto Category*

11. Halaman *Setting Auto Translate*

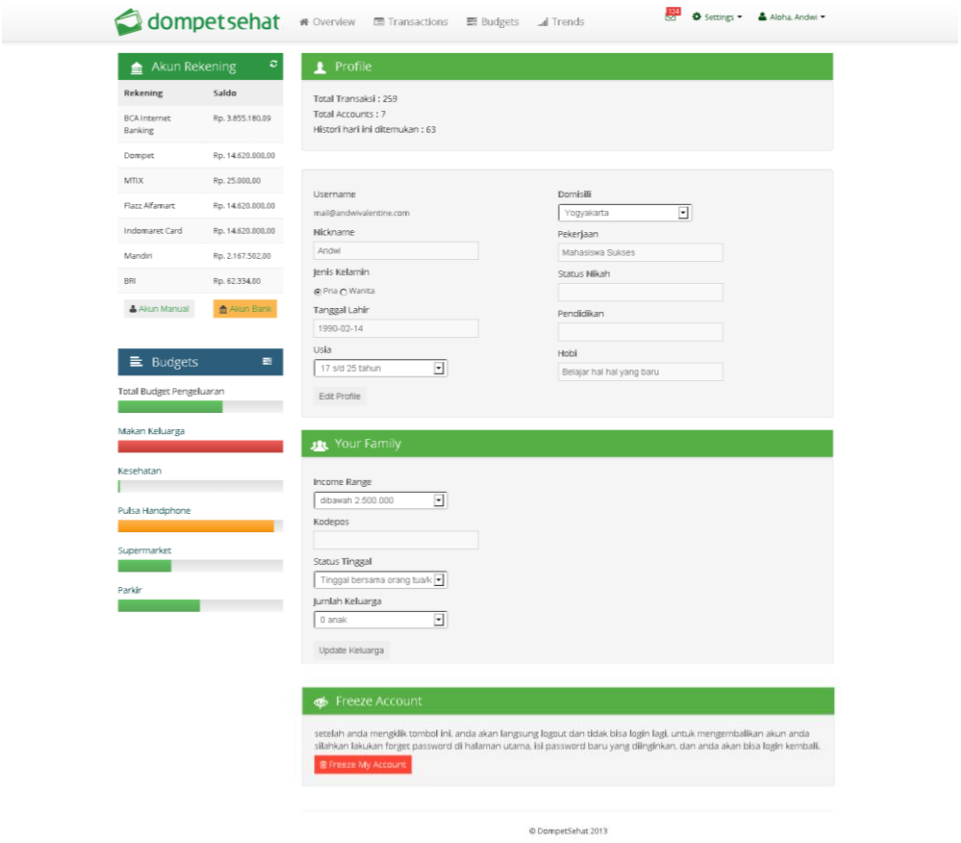
    Halaman ini merupakan halaman untuk setting dekripsi secara otomatis, deskripsi yang berasal dari data aggregasi bank akan diterjemahkan berdasarkan inputan. Halaman *Setting Auto Translate* dapat dilihat pada gambar 4.27 dibawah ini.



Gambar 4.27 Halaman *Setting Auto Translate*

12. Halaman *Profile User*

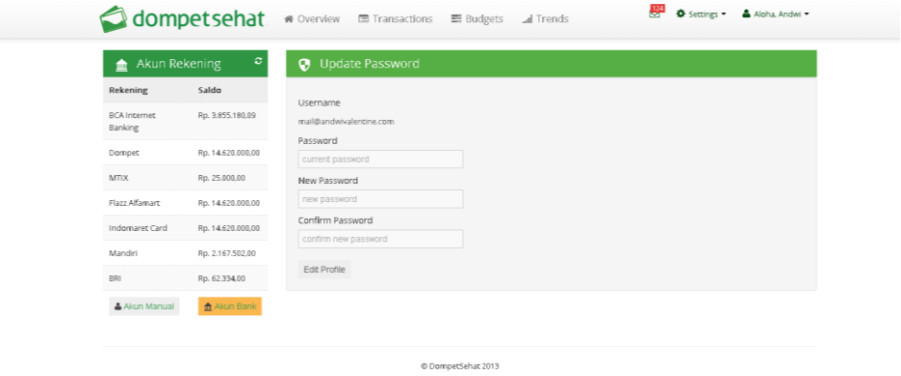
Halaman *Profile User* merupakan halaman yang berguna untuk mengubah profil pengguna. Halaman ini dapat dilihat pada gambar 4.28 dibawah ini.



Gambar 4.28 Halaman *Profile User*

13. Halaman *Update Password*

    Halaman ini merupakan halaman yang bisa memberikan user hak untuk mengubah password user tersebut. Halaman ini dapat dilihat pada gambar 4.29 dibawah ini.



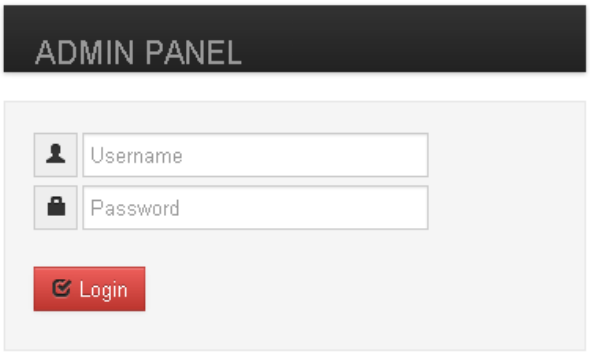
Gambar 4.29 Halaman *Update Password*

4.3.1 Halaman Antarmuka Administrator

Halaman antarmuka *administrator* ini terdiri dari beberapa halaman yang hanya dapat diakses oleh *administrator* saja. Halaman ini terdiri dari beberapa tautan halaman yang mengarah ke halaman lainnya. Halaman yang termasuk dalam halaman tersebut antara lain :

1. Halaman *Login Admin*

   Halaman ini adalah halaman yang berisi form untuk masuk kedalam sistem *administrator* . Halaman *Login Admin* ini dapat dilihat pada gambar 4.30 dibawah ini.



Gambar 4.30 Halaman *Login Admin*

2. Halaman *Dashboard Administrator*

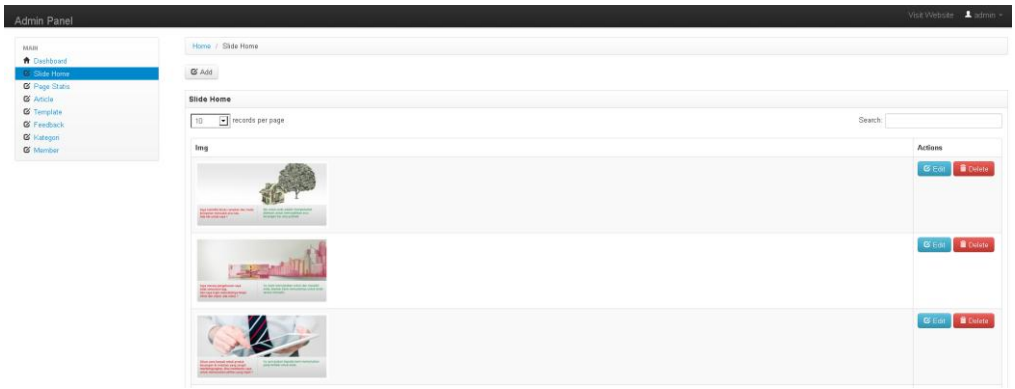
   Halaman *Dashboard Administrator* adalah halaman yang pertama kali muncul ketika berhasil login sebagai *Administrator*. Halaman *Dashboard Administrator* ini dapat dilihat pada gambar 4.31 dibawah ini.



Gambar 4.31 Halaman *Dashboard Administrator*

3. Halaman *Slidehome*

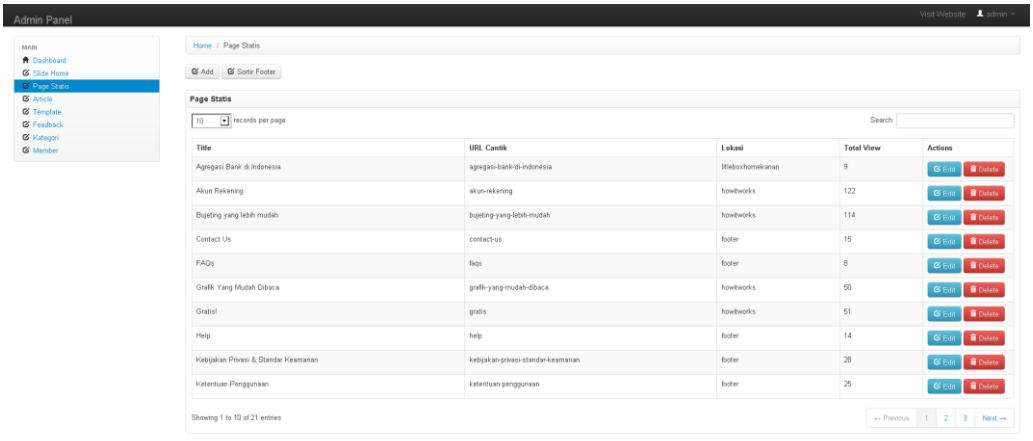
   Halaman *Slidehome* adalah halaman yang beguna untuk manajemen.slider yang akan ditampilkan pada halaman utama visitor . Halaman *Slidehome* ini dapat dilihat pada gambar 4.32 dibawah ini.



Gambar 4.32 Halaman *Slidehome*

4. Halaman *Page Statis*

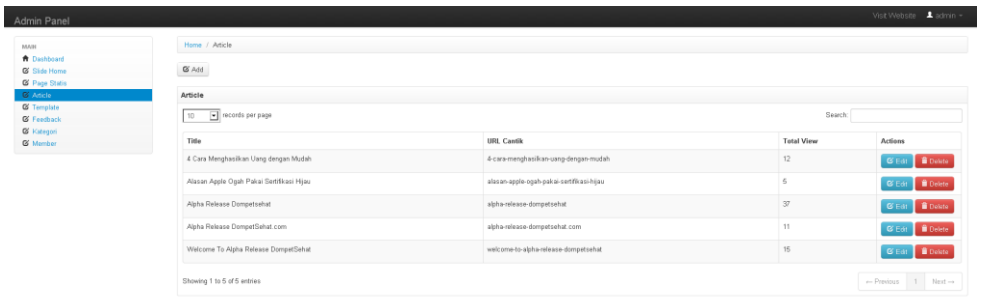
   Halaman *Page Statis* adalah halaman untuk menajemen halaman statis yang akan ditampilkan pada halaman utama aplikasi. Halaman *Page Statis* dapat dilihat pada gambar 4.33 di bawah ini.



Gambar 4.33 Halaman *Page Statis*

5. Halaman *Article*

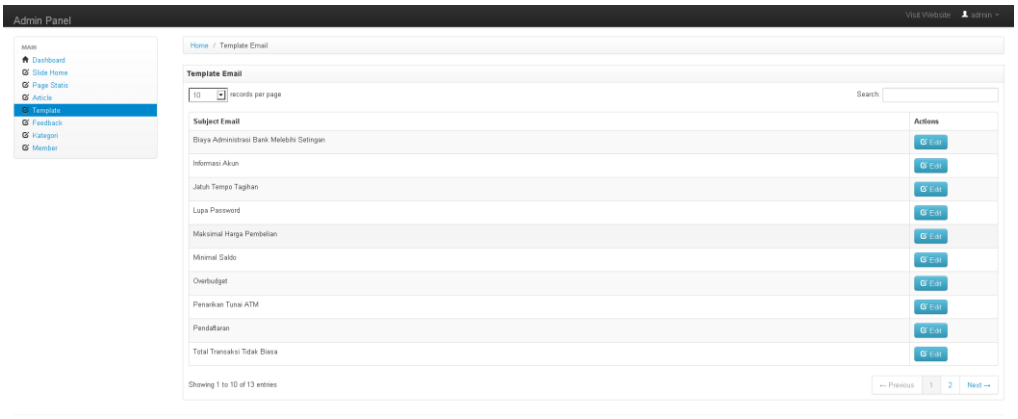
   Halaman *Article* merupakan halaman yang bisa memberikan hak akses untuk admin dalam mengolah artikel. Halaman *Article* dapat dilihat pada gambar 4.34 dibawah ini.



Gambar 4.32 Halaman *Article*

6. Halaman *Template*

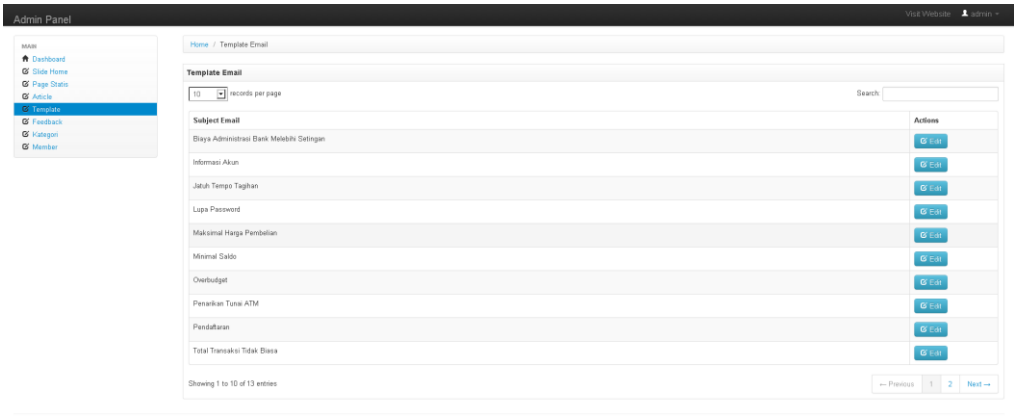
   Halaman *Template* akan memudakan admin untuk mengolah template yang akan terkirim secara otomatis ke user.. Halaman *Template* dapat dilihat pada gambar 4.35 di bawah ini.



Gambar 4.35 Halaman *Template*

7. Halaman *Feedback*

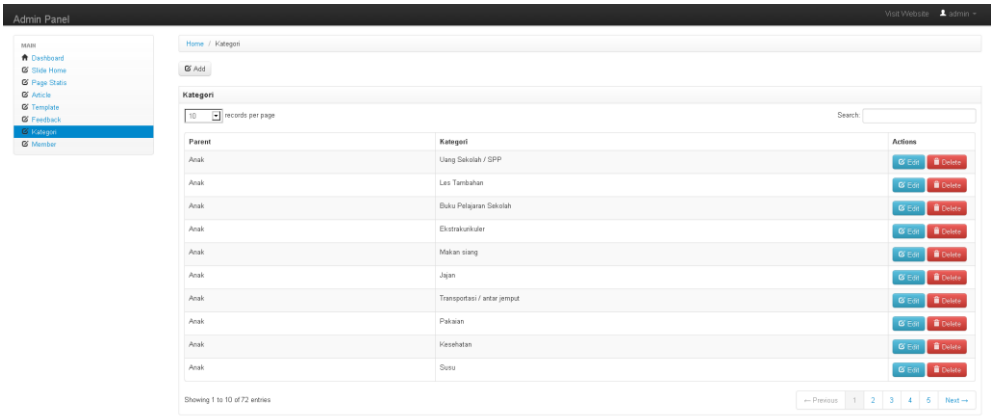
   Halaman ini memberikan hak akses kepada admin untuk melihat pesan yang dikirimkan user lewat form feedback. Halaman *Feedback* dapat dilihat pada gambar 4.36 di bawah ini.



Gambar 4.36 Halaman *Feedback*

2. Halaman Kategori

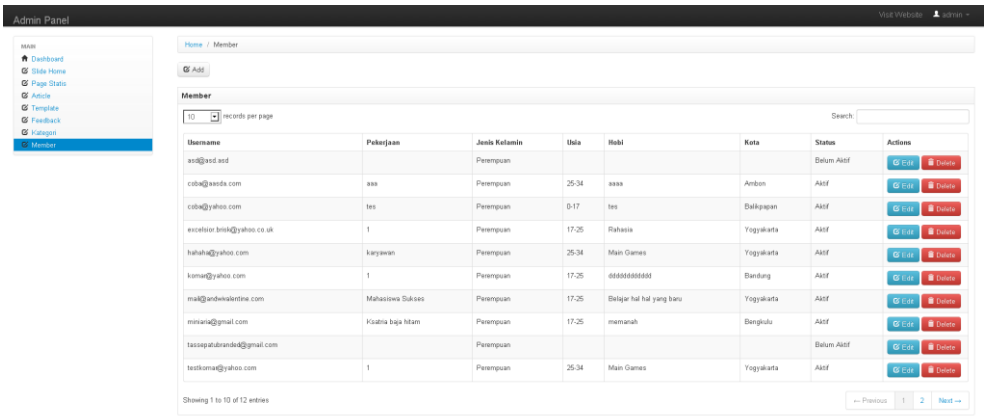
Halaman ini memberikan hak akses kepada admin untuk menambahkan serta mengedit kategori yang akan di tampilkan pada transaksi user . Halaman Kategori dapat dilihat pada gambar 4.37 di bawah ini.



Gambar 4.37 Halaman Kategori

2. Halaman *Member*

Halaman *member* merupakan halaman yang memudahkan admin untuk melihat member yang sudah terdaftar serta m ini memberikan hak akses kepada admin untuk mengubah status user tersebut. Halaman *member* dapat dilihat pada gambar 4.38 di bawah ini.



Gambar 4.38 Halaman *member*

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil dari penelitian dan pembahasan tentang pembuatan aplikasi Dompet Sehat sebagai *Personal Finance* dengan *Account Aggregation* disimpulkan bahwa :

1. Dompet Sehat menggunakan konsep account aggregation dengan Bank BCA.
2. Dompet Sehat mentranslasi bahasa Bank ke bahasa natural dalam membaca transaksi yang sudah diagregasi.
3. Dompet Sehat adalah sebuah aplikasi untuk mengelola dan mengatur keungan pribadi seseorang.

5.2 Saran

Setelah melihat hasil yang dicapai dalam pembuatan aplikasi Dompet Sehat , maka ada beberapa saran yang dapat penulis berikan berkaitan dengan aplikasi yang dibuat ini, antara lain :

1. Menambahkan *Application Programming Interface* (API) pada aplikasi Dompet Sehat agar dengan mudah dikembangkan ke dalam aplikasi mobile seperti Android, IOS, dan lainnya.
2. Memanfaatkan fitur-fitur aplikasi Dompet Sehat untuk mengelola keuangan pribadi mungkin akan lebih efisien dari pada mencatat pemasukan dan pengeluaran secara tertulis.
3. Menambahkan agregasi ke Bank lokal lainnya seperti Bank Mandiri , BRI dan BNI.

DAFTAR PUSTAKA

Adi Nugroho. 2005. *Rational Rose untuk Pemodelan Berorientasi Objek*. Informatika. Bandung.

Bastian, Indra. 2007. *Sistem Akuntansi Sektor Publik*. Salemba Empat Standar Akuntansi. Jakarta

Edy Winarno , Ali Zaki, SmiDev Community. 2010. *Easy Web Programming with PHP plus HTML 5.* Kompas Gramedia. Jakarta

Elmblad, Shelley. 2004. *What is Account Aggregation*. (Online), (http://financialsoft.about.com/od/onlinesoftware/f/Account_Aggrega.htm, diakses 12 Maret 2013)

Fathansyah. 2002. *Basis Data*. Informatika. Bandung.

Hartono, Jogiyanto. 1999. *Analisis & Desain Sistem Informasi*. Andi.Yogyakarta.

Id, Ibnu Daqiqil M.Ti. 2011. *Framework Codeigniter (Sebuah Panduan dan Best Practice)*.(Online),(http://pinguin.ittelkom.ac.id/download/location/File%20Artikel/ Tutorial%20Web/ Framework_Codeigniter.pdf, diakses 23 Febuari 2013)

Ikatan Akuntan Indonesia. 2002. *Keuangan dalam Kerangka Dasar Penyusunan dan Penyajian Laporan keuangan.* Jakarta

Kadir, Abdul. 2002. *Dasar Pemrograman Web Dinamis Menggunakan PHP*. Andi. Yogyakarta.

Nugroho, Adi. 2005. *Analisis dan Perancangan Sistem Informasi dengan Metodologi Berorientasi Objek (Edisi Revisi).* Informatika. Bandung.

Nugroho, Bunafit. 2004. *Database Relasional dengan MySQL*. Andi. Yogyakarta.

Rolliawati. 2012. *Use Case Diagram*. (Online), (http://rolliawati.dosen.narotama.ac.id/files/2012/05/Ansis-8.ppt, diakses 16 Maret 2013)

Schmuller, Joseph. 1999. *Teach Yourself UML in 24 Hours*. Sams Publishing.

Sidik, Betha. 2004. *Pemrograman WEB dengan PHP*. Informatika. Bandung.

Suhendra, Adang. 2008. *Budgeting*. (Online), (http://adang.staff.gunadarma.ac.id/Downloads/files/6108/BUDGETING.pdf, diakses 24 Febuari 2013).

Sumirat, Lambang Probo. 2012. *Membuat Class Diagram*. (Online), (http://karepku. com/wp-content/uploads/2012/06/2.class-diagram.pdf, diakses 28 Febuari 2013)