APLIKASI AGENDA PRIBADI ENOT
BERBASIS ANDROID

SKRIPSI

disusun oleh

Umi Nur Fadhilah

09.11.3359

JURUSAN TEKNIK INFORMATIKA
SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER
AMIKOM YOGYAKARTA
YOGYAKARTA
2013

APLIKASI AGENDA PRIBADI ENOT BERBASIS ANDROID

SKRIPSI

untuk memenuhi sebagian persyaratan
mencapai derajat Sarjana S1
pada jurusan Teknik Informatika

disusun oleh

Umi Nur Fadhilah

09.11.3359

JURUSAN TEKNIK INFORMATIKA
SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER
AMIKOM YOGYAKARTA
YOGYAKARTA
2013

PERSETUJUAN

SKRIPSI

APLIKASI AGENDA PRIBADI ENOT BERBASIS ANDROID

yang dipersiapkan dan disusun oleh

Umi Nur Fadhilah

09.11.3359

telah disetujui oleh Dosen Pembimbing Skripsi
pada tanggal 24 Oktober 2012

Dosen Pembimbing,

M. Rudyanto Arief, MT
NIK. 190302098

PENGESAHAN

SKRIPSI

APLIKASI AGENDA PRIBADI ENOT BERBASIS ANDROID

yang dipersiapkan dan disusun oleh

Umi Nur Fadhilah

09.11.3359

telah dipertahankan di depan Dewan Penguji
pada tanggal 2 Juli 2013

Susunan Dewan Penguji

Nama Penguji	Tanda Tangan
Bambang Sudaryatno, Drs, MM **NIK : 190302029**	
Mei P. Kurniawan, M.Kom **NIK. 190302187**	
M. Rudyanto Arief, MT **NIK. 190302098**	

Skripsi ini telah diterima sebagai salah satu persyaratan
untuk memperoleh gelar Sarjana Komputer
Tanggal 15 Juli 2013

KETUA STMIK AMIKOM YOGYAKARTA

Prof. Dr. M. Suyanto, M.M.
NIK. 190302001

PERNYATAAN

Saya yang bertanda tangan di bawah ini menyatakan bahwa skripsi ini merupakan karya saya sendiri (ASLI), dan isi dalam skripsi ini tidak terdapat karya yang pernah diajukan orang lain untuk memperoleh gelar akademis di suatu Institusi Pendidikan, dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis dan/atau diterbitkan oleh orang lain kecuali yang secara tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka.

Yogyakarta, Juli 2013

Umi Nur Fadhilah
09.11.3359

HALAMAN MOTTO

“Jangan melihat suatu masalah dengan kaca mata kuda. Out Of The Box.” **M. Rudyanto Arief, MT**

“Kesuksesanmu di masa depan bukan (sepenuhnya) ditentukan oleh di mana sekolahmu, siapa gurumu, ilmu apa yang kamu pelajari. Tapi ditentukan oleh APA SIKAPMU di HARI INI.” **Melwin Syafrizal, S.Kom, M.Eng**

“Saat hati berkata ingin, namun Tuhan berkata TUNGGU. Saat air mata harus menetes, Tuhan berkata TERSENYUMLAH. Saat semua terasa membosankan, Tuhan berkata TERUSLAH MELANGKAH. Karena Tuhan selalu tepat waktu dalam memberikan pertolongan.” **DuniaPustaka.com**

“Satu-satunya cara untuk melakukan pekerjaan besar adalah mencintai apa yang anda lakukan. Jika anda belum menemukannya, teruslah mencari. Jangan menetap. Seperti dengan semua masalah hati, anda akan tahu bila anda telah menemukannya.” **Steve Jobs**

“Kehidupan tidak di alam rencana, tidak di alam keluhan. Tapi di alam tindakan.” **Mario Teguh**

“Ayah adalah seseorang yang selalu percaya pada kemampuanmu, orang pertama yang selalu yakin bahwa KAMU BISA.” **DuniaPustaka.com**

HALAMAN PERSEMBAHAN

Karya tulis ini penulis persembahkan untuk mereka yang telah berjasa dalam membantu penulis.

- Alloh SWT makasih untuk semua masalah yang benar-benar menakjubkan yang mendewasakan, yang menyabarkan, yang menjadikan pribadi yang kuat, yang selalu mengajari untuk menghargai yang dimiliki, mengajari selalu percaya kemampuan diri sendiri. Terima kasih selalu memberikan yang terbaik, walau bukan keinginanku. Terima kasih selalu mendengar doaku, walaupun aku tidak punya kata yang tepat saat berdoa. Makasih – I Love You GOD.
- Mom and Dad, my Bro and his wife. Aku Sarjana. Makasih buat vitaminnya, susunya, makanannya, kuenya, dukungannya, buat doa dan apapun itu.
- Special thanks to my lovely Catur Wahyu Karina and Fera Herlina Sari : thank you for always supported me, thank you strengthen me on weak moments. Always be on my side when I was in a most deplorable conditions. Thank You So Much.
- M. Rudyanto Arief, MT : terima kasih pak sudah menjadi Dosen Pembimbing yang hebat dan keren, makasih buat ilmunya. I like your style Pak.
- Kos Kinasih : Amel, Ditut, Yiyin, Mbak Puji, Mbak Putri, Mbak Siska, Mbak Uci, Mbak Yuli, Mbak Nur, Mbak Intun, Caca, Nana, Yusi, Mbak Na, Cici, Ladies, Aini, Om and anothers. Makasih semuanya, printernya, scanner-nya,

tintanya, minumnya, tebengannya, cemilannya, *mouse*-nya, kue-kuenya, cerita-ceritanya, dukungannya, waktunya, canda tawanya, poker'annya - Love you all.

- Makasih para mastah Mawaddah Aynurrohmah, Dikky Faiz Maulana, Pratama Nur Wijaya, Rizky Wais Kurniawan, Yogi Anggriawan, Puji Laksono, Nurlela Anggraini, Mohammed Irfan, Fendy Sapto : makasih buat pencerahannya, bimbingannya, bantuannya. Makasih.
- Kontrakan 211 & Kelas 09-S1TI-K & tetangga Mizwar : makasih ya buat internet gratisnya, ide-idenya. Yee kita LULUS, udah Sarjana.

KATA PENGANTAR

Puji dan syukur penulis panjatkan atas kehadirat Allah SWT, Tuhan semesta alam, atas segala nikmat, karunia, rahmat, serta hidayahnya sehingga penulis dapat menyelesaikan skripsi ini. Sholawat serta salam tak lupa hamba haturkan kepada Nabi besar Muhammad SAW yang telah membawa kemajuan besar kepada umat manusia di dunia.

Penulis menyadari bahwa penulisan skripsi ini masih banyak terdapat kekurangan baik dari segi teknik penulisan maupun dari segi kualitas skripsi. Meskipun demikian, penulis berharap semoga skripsi ini dapat bermanfaat bukan hanya bagi penulis tapi bagi teman-teman ataupun pihak-pihak yang ingin mendapatkan ilmu atau informasi mengenai pengembangan aplikasi pada *flatform* android.

Penulis juga menyadari bahwa tanpa bantuan dari berbagai pihak, penulisan skripsi ini tidak dapat berjalan lancar. Oleh sebab itu penulis ingin menyampaikan ucapan terima kasih yang sebesar-besarnya kepada :

1. Bapak M. Rudyanto Arief, MT selaku Dosen Pembimbing yang telah memberikan arahan, bimbingan dan masukan selama proses penyusunan Laporan Skripsi hingga selesai.
2. Bapak Prof. Dr. H. Mohammad Suyanto, MM, selaku Ketua Sekolah Tinggi Manajemen Informatika dan Komputer AMIKOM Yogyakarta.
3. Bapak Sudarmawan, MT, selaku Ketua Jurusan Teknik Informatika (S-1).
4. Bapak Bambang Sudaryatno, Drs, MM dan Bapak Mei P. Kurniawan, M.Kom selaku Dosen Penguji.
5. Staff, karyawan, dan Dosen di lingkungan STMIK AMIKOM Yogyakarta. Teman-teman mahasiswa dan mahasiswi Teknik Informatikan angkatan

2009 yang telah memberikan banyak dukungan dan semangat kepada Penulis.

6. 09-S1TI-11 Family dan semua teman – teman yang telah menginspirasi penulis untuk selalu bersemangat.

Terima kasih kepada semua orang yang penulis sebutkan diatas atas segala budi dan amal baiknya selama ini. Penulis hanya bisa mendoakan agar Allah SWT memberikan balasan yang berlipat ganda atas segala kebaikan kalian. Akhir kata, semoga skripsi ini dapat bermanfaat bagi kita semua. Amin.

Yogyakarta, 31 Juli 2013

Penulis

Umi Nur Fadhilah

DAFTAR ISI

HALAMAN JUDUL ii
HALAMAN PENGESAHAN iii
HALAMAN PERSETUJUAN iv
HALAMAN PERNYATAAN KEASLIAN v
HALAMAN MOTO vi
HALAMAN PERSEMBAHAN vii
KATA PENGANTAR ix
DAFTAR ISI xi
DAFTAR TABEL xvii
DAFTAR GAMBAR xviii
INTISARI xxi
ABSTRACT xxii
BAB I PENDAHULUAN 1
1.1. Latar Belakang 1
1.2. Rumusan Masalah 1
1.3. Batasan Masalah 2
1.4. Tujuan Penelitian 2
1.5. Manfaat Penelitian 3
1.6. Metode Penelitian 3
1.7. Sistematika Penulisan 4
BAB II LANDASAN TEORI 6
2.1. Perkembangan Sistem Berbasis Mobile 6
2.2. Android 9
2.2.1. Definisi Android 9
2.2.2. Sejarah Android 10
2.2.3. Arsitektur Android 11
2.2.3.1. Application and Widgets 11
2.2.3.2. Application Framwork 11
2.2.3.3. Libraries 12

2.2.3.4. Android Run Time 13
2.2.3.5. Linux Kernel 14
2.2.4. Fundamental Aplikasi 14
2.2.5. DVM (Dalvik Virtual Machine) 16
2.2.6. Versi Android 17
2.2.6.1. Android Versi 1.1 17
2.2.6.2. Android Versi 1.5 (Cupcake) 17
2.2.6.3. Android Versi 1.6 (Donut) 18
2.2.6.4. Android Versi 2.0/2.1 (Enclair) 18
2.2.6.5. Android Versi 2.2 (Froyo) 19
2.2.6.6. Android Versi 2.3 (Ginggerbread) 19
2.2.6.7. Android Versi 3.0/3.1 (Honeycomb) 19
2.2.6.8. Android Versi 4.0 (Ice Cream Sandwich) 19
2.2.6.9. Android Versi 4.1 (Jelly Bean) 20
2.3. Pengertian Agenda 20
2.3.1. Macam-macam Contoh Aplikasi Agenda Pribadi 21
2.3.1.1. Evernote 21
2.3.1.2. Do It Tommorow 22
2.3.1.3. Spring Pad 23
2.3.1.4. Any Do 24
2.3.2. Agenda Pribadi Enot 25
2.4. Perangkat Lunak Yang Digunakan 26
2.4.1. Eclipse IDE 27
2.4.2. Java 28
2.4.2.1. Pengertian Java 28
2.4.2.2. Versi Awal Java 29
2.4.2.3. Kelebihan Java 30
2.4.3. Android SDK (Software Development Kit) 31
2.4.4. Inkscape 33
2.4.4.1. Sejarah Inkscape 33
2.2.5. SQLite 34

2.5. Pengenalan UML 36
2.5.1. Use Case Diagram 36
2.5.2. Class Diagram 38
2.5.3. Sequence Diagram 39
2.5.4. Activity Diagram 39
2.6. ERD 40
2.7. Database 42
2.8. Konsep Siklus Hidup Sistem 43
2.8.1. Analisis 44
2.8.2. Desain 44
2.8.2.1. Desain Logis 44
2.8.2.2. Desain Fisik 44
2.8.3. Implementasi 45
2.8.3.1. Testing 45
2.8.3.2. Instalasi 45
2.8.4. Pemeliharaan 45
2.9. Konsep Waterfall 46
2.10. Analisis SWOT 48
2.11. Pengujian Aplikasi Mobile 49
2.11.1. Code Development and Integration 49
2.11.1.1. Coding 49
2.11.1.2. Unit Testing 50
2.11.2. Integration and System Testing 50
2.11.2.1. Testing Proses 50
2.11.2.2. Testing Documentation 51
2.11.2.3. Testing Considerations 51
BAB III ANALISIS DAN PERANCANGAN SISTEM 52
3.1. Analisis Sistem 52
3.1.1. Analisis Kelemahan Sistem 52
3.1.2. Analisis Kebutuhan Sistem 55
3.1.2.1. Kebutuhan Fungsional 55

3.1.2.2. Kebutuhan Non Fungsional 56
3.2.2.3. Analisis Kelayakan Sistem 59
3.1.3. Analisis Kelayakan Sistem 57
3.1.3.1. Kelayakan Teknologi 57
3.1.3.2. Kelayakan Hukum 58
3.1.3.3. Kelayakan Operasional 58
3.2. Perancangan Sistem 58
3.2.1. Perancangan Proses 59
3.2.1.1. Use Case Diagram 59
3.2.1.2. Use Case Description 59
3.2.1.3. Activity Diagram 66
3.2.1.4. Class Diagram 71
3.2.1.5. Sequence Diagram 72
3.2.2. Perancangan Database 74
3.2.2.1. ERD 75
3.2.2.2. Struktur Tabel 75
3.2.3. Perancangan Interface / Antarmuka 78
BAB IV IMPLEMENTASI DAN PEMBAHASAN 83
4.1. Implementasi 83
4.1.1. Membuat Projek Enot 85
4.1.2. Cek SDK 85
4.1.3. Implementasi Interface 86
4.1.3.1. Splash Screen 87
4.1.3.2. Dashboard/Menu Utama Enot 88
4.1.3.3. Menu Foto 89
4.1.3.4. Menu Suara 90
4.1.3.5. Menu Ketik 91
4.1.3.6. Menu Kalender 92
4.1.3.7. List Kegiatan 93
4.1.3.8. Menu Edit 94
4.1.4. Membuat Database Enot 97

4.1.5. Cek Database Di SQLite Browser 98
4.1.6. Mencoba Insert Di Form Daftar 99
4.1.7. Mengakses Form Login 100
4.1.8. Memberikan Code Pada Menu Dashboard 101
4.1.8.1. Memberikan Code Pada Menu Foto 101
4.1.8.2. Memberikan Code Pada Menu Suara 101
4.1.8.3. Memberikan Code Pada Menu Ketik 102
4.1.8.4. Memberikan Code Pada Menu Kalender 103
4.1.8.5. Memberikan Code Pada Button Keluar 104
4.1.9. Membuat Class Model 105
4.1.10. Menampilkan Semua Kegiatan 106
4.1.11. Menampilkan Single Kegiatan 107
4.1.12. Membuat Layout Edit 108
4.1.13. Contoh Saat List Dipilih Menuju Ke Menu Edit 109
4.1.14. Memberikan 3 Button Pada Halaman Edit 109
4.1.15. Pengujian Program Pada Beberapa Smartphone 110
4.1.16. Pemeliharaan Program 111
4.2. Pengujian Program 111
4.2.1. Code Development and Integration 111
4.2.2. Integration and System Testing 114
4.3. Pembahasan 115
4.3.1. Pembahasan Listing Program 115
4.3.1.1. Pembahasan Daftar 116
4.3.1.2. Pembahasan Login 116
4.3.1.3. Pembahasan Simpan 117
4.3.1.4. Pembahasan Code Memanggil Foto 118
4.3.1.5. Pembahasan Code Pada Menu Suara 119
4.3.1.6. Pembahasan Set Waktu 120
4.3.1.7. Pembahasan Set Tanggal 121
4.3.1.8. Pembahasan Menu Kalender 122

4.3.1.9. Pembahasan Tampil Kegiatan Pada Menu Kalender 123
4.3.1.10. Pembahasan Tampil List Kegiatan 123
4.3.1.11. Pembahasan Button Berbagi 124
4.3.1.12. Pembahasan Button Hapus 124
4.3.2. Code Development and Integration 125
BAB V PENUTUP 126
5.1. Kesimpulan 126
5.2. Saran 126
DAFTAR PUSTAKA. 128

DAFTAR TABEL

Tabel 2.1. Perubahan Paradigma Dalam Telekomunikasi Indonesia 7
Tabel 2.2. Simbol Use Case Diagram 36
Tabel 2.3. Simbol Class Diagram 38
Tabel 2.4. Simbol Sequence Diagram 39
Tabel 2.5. Simbol Activity Diagram 40
Tabel 2.6. Simbol ERD 42
Tabel 3.1. Analisis SWOT 53
Tabel 3.2. Kebutuhan Perangkat Keras 56
Tabel 3.3. Kebutuhan Hardware Penerapan 56
Tabel 3.4. `Kebutuhan Perangkat Lunak 57
Tabel 3.5. Use Case Description Menginputkan Kegiatan Beserta Mengupload Foto..... 59
Tabel 3.6. Use case Description Menginputkan Kegiatan Dengan Mengetik..... 60
Tabel 3.7. Use Case Description Menginputkan Kegiatan Beserta Suara.....61
Tabel 3.8. Use Case Description Melihat List Kegiatan 61
Tabel 3.9. Use Case Description Melihat Isi Kegiatan 62
Tabel 3.10. Use Case Description Mengedit Isi Kegiatan 63
Tabel 3.11. Use Case Description Berbagi Isi Kegiatan 63
Tabel 3.12. Use Case Description Menghapus Isi Kegiatan 64
Tabel 3.13. Use Case Description Menyediakan Waktu Pengingat 65
Tabel 3.14. Struktur Tabel tb_daftar 75
Tabel 3.15. Struktur Tabel tb_foto 76
Tabel 3.16. Struktur Tabel tb_suara 77
Tabel 3.17. Struktur Tabel tb_ketik 77
Tabel 4.1. Uji Program Pada Beberapa Smartphone 110
Tabel 4.2. Kelemahan dan Keunggulan Enot 125

DAFTAR GAMBAR

Gambar 2.1. Arsitektur Android 14
Gambar 2.2. Agenda 21
Gambar 2.3. Tampilan Workbench Eclipse 28
Gambar 2.4. Tampilan Inkscape 34
Gambar 2.5. Konsep Waterfall 46
Gambar 2.6. Proses Pembangunan Sistem 47
Gambar 2.7. Diagram Ilustrasi Analisis SWOT 48
Gambar 2.8. Contoh Testing Black Box 51
Gambar 3.1. Use Case Diagram Enot 59
Gambar 3.2. Activity Diagram Registrasi User 66
Gambar 3.3. Activity Diagram Tampil Menu Foto 67
Gambar 3.4. Activity Diagram Tampil Menu Suara 68
Gambar 3.5. Activity Diagram Tampil Menu Ketik 69
Gambar 3.6. Activity Diagram Tampil Menu Kalender 70
Gambar 3.7. Class Diagram Enot 71
Gambar 3.8. Sequence Diagram Daftar 72
Gambar 3.9. Sequence Diagram Menu Foto 72
Gambar 3.10. Sequence Diagram Menu Suara 73
Gambar 3.11. Sequence Diagram Menu Ketik 73
Gambar 3.12. Sequence Diagram Menu Kalender 74
Gambar 3.13. Rancangan Model ERD 75
Gambar 3.14. Splash Screen Enot 78
Gambar 3.15. Login Enot 78
Gambar 3.16. Daftar Enot 79
Gambar 3.17. Menu Utama Enot 79
Gambar 3.18. Menu Foto Enot 80
Gambar 3.19. Menu Suara Enot 80
Gambar 3.20. Menu Ketik Enot 81
Gambar 3.21. Menu Kalender Enot 81

Gambar 3.22. List Kegiatan Enot 82
Gambar 4.1. Membuat Project Enot 85
Gambar 4.2. Cek SDK 86
Gambar 4.3. Implementasi Icon 87
Gambar 4.4. Tampilan Splash Screen 88
Gambar 4.5. Tampilan Dashboard / Menu Utama Enot 89
Gambar 4.6. Tampilan Menu Foto Enot 90
Gambar 4.7. Tampilan Menu Suara Enot 91
Gambar 4.8. Tampilan Menu Ketik Enot 92
Gambar 4.9. Tampilan Menu Kalender 93
Gambar 4.10. Tampilan List Kegiatan Enot 94
Gambar 4.11. Tampilan Menu Edit Foto 95
Gambar 4.12. Tampilan Menu Edit Suara 96
Gambar 4.13. Tampilan Menu Edit Ketik 97
Gambar 4.14. Code Database ENOT 98
Gambar 4.15. Tampilan Database ENOT Pada SQLite Browser 98
Gambar 4.16. Daftar Menjadi Member ENOT 99
Gambar 4.17. Data Berhasil Disimpan 99
Gambar 4.18. Menginputkan Data Pada Form Login 100
Gambar 4.19. Menu Utama ENOT 100
Gambar 4.20. Code Pada Menu Foto 101
Gambar 4.21. Code Pada Menu Suara 102
Gambar 4.22. Code Pada Menu Ketik 103
Gambar 4.23. Code Pada Menu Kalender 104
Gambar 4.24. Code Pada Button Keluar 105
Gambar 4.25. Class Model Untuk Penghubung 106
Gambar 4.26. Kegiatan Di Menu Kalender 107
Gambar 4.27. Single Kegiatan Di List 108
Gambar 4.28. Layout Menu Edit Foto, Suara, Ketik 108
Gambar 4.29. Menuju Ke Menu Edit 109
Gambar 4.30. Button Pada Halaman Edit 110

Gambar 4.31. Tampilan Database di SQLite Browser 112
Gambar 4.32. Tampilan Source Code Button Keluar 113
Gambar 4.33. Kesalahan Button Keluar Muncul di Logcat 113
Gambar 4.34. Tampilan Source Code Otentikasi Login 114
Gambar 4.35. Potongan Listing Code Daftar 116
Gambar 4.36. Potongan Listing Code Login 117
Gambar 4.37. Potongan Listing Code Simpan 118
Gambar 4.38. Potongan Listing Code Memanggil Foto 118
Gambar 4.39. Potongan Listing Code Mulai Merekam 119
Gambar 4.40. Potongan Listing Code Selesai Merekam 119
Gambar 4.41. Potongan Listing Code Memutar Hasil Rekaman 120
Gambar 4.42. Potongan Listing Code Selesai Memutar Rekaman 120
Gambar 4.43. Potongan Listing Code Set Waktu 121
Gambar 4.44. Potongan Listing Code Set Tanggal 122
Gambar 4.45. Potongan Listing Code Menampilkan Kalender 123
Gambar 4.46. Potongan Listing Code Cara Menampilkan Kegiatan 123
Gambar 4.47. Potongan Listing Code Cara Menampilkan List 124
Gambar 4.48. Potongan Listing Code Button Berbagi 124
Gambar 4.49. Potongan Listing Code Button Hapus 125

INTISARI

Mengabadikan momen merupakan suatu hal yang biasa dilakukan seseorang menggunakan *smartphone*. Mengelompokan beberapa momen menjadi satu akan memudahkan seseorang dalam mengorganisir banyak kegiatan. Tetapi banyak agenda yang rumit cara pengoperasiannya karena terlalu banyak fitur-fitur.

Berdasarkan masalah tersebut, maka diperlukan sebuah agenda pribadi yang lebih *usable* untuk bisa diakses dengan cepat dan mudah melalui perangkat *mobile*. Agenda harus bisa menjaga kerahasiaan member.

Hasil dari analisis ialah penulis membuat sebuah aplikasi yang lebih *usable* saat dioperasikan. Aplikasi dapat membantu menuliskan kegiatan seseorang. Waktu pengingat merupakan fitur yang bisa dimanfaatkan member.

Kata Kunci : aplikasi android, aplikasi enot, organisasi kegiatan

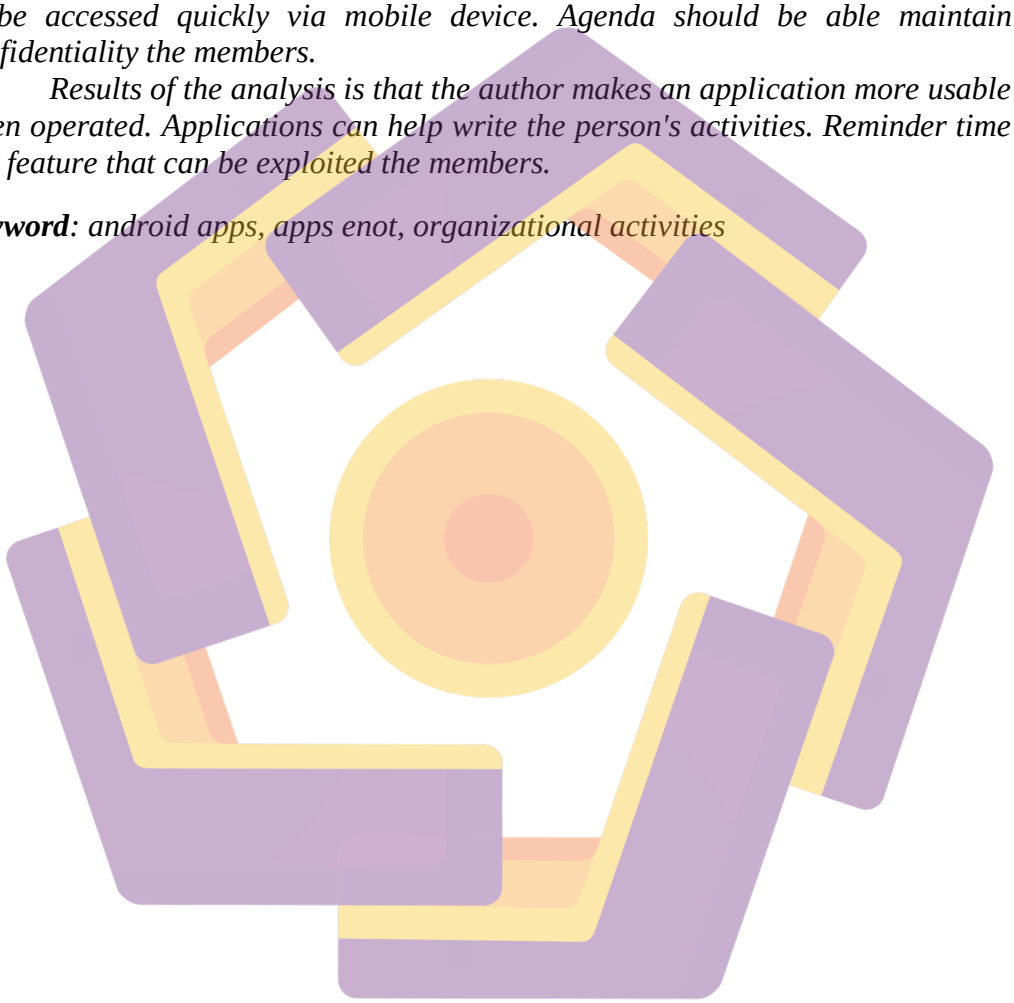
ABSTRACT

Capture the moment is a common way of someone using smartphone. Group some moments into one will help someone in organizing many activities. But a lot of complicated agendas how to operate because too many features.

Based on these issues, it would require personal agenda a more usable to be accessed quickly via mobile device. Agenda should be able maintain confidentiality the members.

Results of the analysis is that the author makes an application more usable when operated. Applications can help write the person's activities. Reminder time is a feature that can be exploited the members.

***Keyword**: android apps, apps enot, organizational activities*



BAB I

PENDAHULUAN

1.1. Latar Belakang

Di era moderen seperti ini *smartphone* adalah salah satu sarana yang memudahkan keseharian seseorang. Mengabadikan momen merupakan suatu hal yang biasa dilakukan oleh seseorang menggunakan *smartphone*. Mengelompokan beberapa momen menjadi satu akan memudahkan seseorang dalam mengorganisir banyak kegiatan. Tetapi banyak agenda yang rumit cara pengoperasiannya karena terlalu banyak fitur-fitur.

Dengan adanya permasalahan di atas, maka diperlukan sebuah agenda pribadi yang lebih *use able* untuk bisa diakses dengan cepat dan mudah melalui perangkat *mobile*. Agenda pribadi "enot" merupakan aplikasi *mobile* berbasis android yang memberikan kemudahan dalam merancang dan membantu memanajemen keseharian seseorang. Dari kesimpulan di atas, ada sebuah inisiatif untuk membangun aplikasi berbasis android yang berjudul "APLIKASI AGENDA PRIBADI ENOT BERBASIS ANDROID".

1.2. Rumusan Masalah

Dari ulasan yang terdapat pada halaman latar belakang, maka didapat rumusan masalah, yaitu : "Bagimana membangun aplikasi agenda pribadi enot berbasis android?".

1.3. Batasan Masalah

Agar tidak menyimpang jauh dari permasalahan yang ada, maka perlu adanya batasan masalah, yaitu :

1. Aplikasi yang dibuat adalah media untuk menuliskan kegiatan seseorang.
2. Aplikasi ini dapat menginputkan foto suara dan teks.
3. Aplikasi ini dapat menampilkan *alert* dengan suara *default*.
4. Aplikasi ini menggunakan minimal SDK 3.0.
5. User mendaftar saat pertama kali menginstal secara otomatis menjadi member.
6. Software yang digunakan adalah Eclipse dengan Java (versi7 update 21) untuk mengkompilasi aplikasi Android.

1.4. Tujuan Penelitian

Memuat uraian yang menyebutkan secara spesifik maksud dari keinginan yang hendak dicapai, maka tujuan dari pembuatan Skripsi ini adalah sebagai berikut :

1. Sebagai salah satu syarat menyelesaikan pendidikan S1 pada jurusan Teknik Informatika STMIK “AMIKOM” YOGYAKARTA.
2. Dapat membuat aplikasi agenda pribadi “enot” berbasis android yang dapat membantu mengingat / menuliskan momen seseorang.
3. Menerapkan disiplin ilmu yang telah didapatkan selama menjalani perkuliahan.

1.5. Manfaat Penelitian

Hal yang di dapat dari hasil tujuan yang telah direncanakan, antara lain :

1. Memperoleh gelar Sarjana Komputer pada STMIK "AMIKOM" Yogyakarta.
2. Membantu mengembangkan teknologi informasi.
3. Memperdalam pengetahuan tentang pemrograman android, selanjutnya mengaplikasikan konsep dan teori mengenai aplikasi android pada *mobile*.
4. Dapat membangun sebuah aplikasi *mobile* bebasis android yang mampu membantu dalam keseharian seseorang.
5. Dapat mempermudah seseorang dalam mengabadikan momen atau kegiatan harian seseorang.

1.6. Metode Penelitian

Metodologi penelitian yang digunakan dalam penulisan ini baik dalam menyelesaikan penulisan maupun untuk menyelesaikan pembuatan program ditempuh melalui beberapa metode penelitian, yaitu :

1. Metode Observasi

   Melakukan pengamatan terhadap memo dan kalender di beberapa ponsel dan apa saja fitur-fitur menariknya.

2. Metode Kepustakaan

   Mengumpulkan sumber-sumber pendukung dari berbagai buku dan internet mengenai seluk beluk pemrograman android dan pemrograman java.

3. Analisis

Melakukan analisis terhadap data-data yang telah diperoleh, untuk mengidentifikasi masalah yang dihadapi. Dari hasil analisis dihasilkan gambaran kondisi dan masalah yang dihadapi sehingga menghasilkan penanganan yang tepat.

4. Perancangan Program

   Merancang proses aplikasi yang nantinya bisa mempermudah membuat program berikutnya.

5. Pembuatan Program

   Proses pembuatan program berdasarkan pada rancangan program yang sudah dibuat.

6. Tahap Uji Program

   Pada tahap ini membuktikan apakah aplikasi tersebut dapat berjalan dengan baik atau tidak, dan dapat digunakan sesuai dengan harapan.

1.7. Sistematika Penulisan

Penulisan skripsi ini disusun secara sistematis yang terdiri dari bagian-bagian yang saling berhubungan satu dengan yang lainnya. Adapun uraian singkat mengenai isi tulisan ini adalah sebagai berikut.

BAB I PENDAHULUAN

Pada bab ini menjelaskan mengenai latar belakang masalah, rumusan masalah, batasan masalah, tujuan dan manfaat, metode penelitian, dan sistematika penulisan laporan.

BAB II LANDASAN TEORI

Bab ini menguraikan tentang teori-teori yang digunakan dalam penelitian secara mendetail, berupa definisi dan model matematis yang berkaitan dengan ilmu dan masalah yang diteliti.

BAB III ANALISIS DAN PERANCANGAN SISTEM

Bab ini menguraikan tentang analisis terhadap semua permasalahan yang muncul dan penyelesaiannya. Pada bab ini juga berisikan rancangan terhadap penelitian yang dilakukan, baik rancangan secara umum dari sistem yang dibangun maupun perancangan secara spesifik.

BAB IV IMPLEMENTASI DAN PEMBAHASAN SISTEM

Bab ini menjelaskan tentang implementasi hasil serta pembahasan dari sebuah aplikasi yang telah dibuat dan sebagai gambaran bagaimana cara mengoperasikan serta pembahasan dari hasil implementasinya.

BAB V PENUTUP

Bab ini penulis membahas tentang kesimpulan dari perumusan masalah yang disampaikan, serta saran yang membangun bagi pengembangan diri.

DAFTAR PUSTAKA

Daftar pustaka berisi mengenai sumber-sumber materi dan data yang telah member masukan bagi penulis dalam mengerjakan Skripsi ini.

BAB II

LANDASAN TEORI

2.1. Perkembangan Sistem Berbasis Mobile

Kebutuhan perangkat telekomunikasi dewasa ini tidak hanya untuk komunikasi suara, tetapi sudah tuntutan untuk komunikasi data, gambar dan *video* membentuk komunikasi multimedia. Komunikasi multimedia sudah menjadi keharusan dan ini dimungkinkan karena telah terjadinya konvergensi beberapa layanan seperti *voice*, data, gambar dan *video*. Telah banyak aplikasi layanan telekomunikasi yang banyak dinikmati *user* akibat dari konvergensi layanan yang terjadi. Aplikasi layanan telekomunikasi yang pada awalnya hanya layanan *fixed* sekarang ini telah dituntut untuk dapat dinikmati menggunakan perangkat bergerak seperti PDA atau Laptop. Beberapa aplikasi layanan multimedia yang sekarang banyak dinikmati antara lain adalah *m-learning*, *m-banking*, *m-shopping*, *e-government*, *tele-medicine*, dan lain-lain.

Kemajuan teknologi telekomunikasi dan informatika biasa disebut Informatics Communications Technology (ICT) telah banyak membantu pengguna dalam kehidupan sehari-hari. Dengan kemajuan teknologi ICT telah mengubah paradigma hidup. Perubahan paradigma terbesar yang terjadi pada teknologi adalah dengan diperkenalkannya teknologi berbasis internet protokol (IP) di mana mampu menciptakan konvergensi yang seutuhnya dan memberikan banyak keuntungan. Dengan IP dimana komunikasi yang semula berbasis circuit switch berubah menjadi packet switch yang memberikan keuntungan terutama untuk efisiensi bandwidth.

Tabel 2.1 Perubahan Paradigma Dalam Telekomunikasi Indonesia

Perubahan Paradigma dalam Telekomunikasi	
Paradigma Lama	**Paradigma Baru**
▪ Pasar Monopolistik	▪ Pasar Kompetitif
▪ Regulasi Sangat Ketat	▪ Hampir Tanpa Regulasi
▪ Infrastruktur Telekomunikasi	▪ Infrastruktur Informasi
▪ Jasa Dasar dan Non-dasar	▪ Jaringan dan Jasa
▪ Informasi dengan format terpisah untuk suara, data, teks, gambar	▪ Informasi dalam format multimedia (konvergensi)
▪ Hybrid Analog / Digital	▪ Seluruhnya Digtal
▪ Circuit Switched	▪ IP (Packet Switch)
▪ Dominasi Saluran Kawat / Kabel	▪ Dominasi oleh Nir- Kabel dan Bergerak (mobile)
▪ Pentaripan sesuai jumlah ‘menit’	▪ Pentaripan sesuai jumlah ‘byte’
▪ Tergantung Jarak	▪ Tidak tergantung Jarak
▪ Dominasi badan usaha milik negara	▪ Dominasi oleh perush swasta dan perush publik
▪ Industrial Economy	▪ New Economy

Sumber : Wibisono, Gunawan dan Hantoro, Gunadi Dwi. 2008. Mobile Broadband. Bandung: Informatika

Konvergensi merupakan istilah yang dipergunakan untuk mendeskripsikan kecenderungan pasar dan teknologi, yang batas-batasnya mulai kabur. Lagipula regulasi juga makin kurang terkokontak-kotak menurut jenis palayanan yang

disediakan. Sebagai salah satu contoh konvergensi, batas antara segmen pasar kabel TV, telepon dan akses internet yang sekarang semuanya dapat berbasis pada platform apa saja, lambat laun mulai menghilang. Jadi konvergensi adalah gejala menyatunya kecenderungan teknologi, pasar dan regulasi sekaligus.

Konvergensi dari industri telekomunikasi, informatika dan multimedia termasuk penyiaran sudah akan menjadi ketiga bidang ini tidak terpisahkan / terbedakan satu dari lainnya, terutama dalam penggunaan infrastruktur, cara penyampaian muatan informasi (content) serta aplikasinya. Kenvergensi memungkinkan jaringan yang berbeda menyalurkan layanan yang sama dan beragam layanan dapat disalurkan melalui jaringan yang sama.

Namun dalam kenyataannya telekomunikasi di Indonesia diatur oleh Undang-undang Telekomunikasi No 36/1999 berikut PP dan KM yang terkait, sedangkan penyiaran dan informatika diatur atau akan diatur oleh perangkat Undang-undang berikut PP dan KM tersendiri secara terkotak-kotak. Sehingga perlu disususn langkah-langkah penyelarasan atau pengintregrasian UU, PP dan peraturan pelaksanaan lainnya menjadi perangkat peraturan baik di pusat maupun di daerah yang dapat diterapkan untuk industri yang semakin konvergen ini[1].

Kemajuan ICT yang menuju konvergensi telah mendorong perubahan tren gaya hidup dari user. Di mana gaya hidup user dewasa ini memiliki karakteristik sebagai berikut :

1. Ketergantungan akan telepon bergerak mulai tinggi,
2. Koneksi *online* mulai populer,

[1] Wibisono, Gunawan dan Hantoro, Gunadi Dwi. 2008. Mobile Broadband. Bandung: Informatika

3. *Mix content on-line* dan *off-line*. Sebagian besar *content* adalah *off-line* dalam bentuk CDROM (VCD/DVD, majalah dan buku),
4. *Contact List* ada pada aplikasi, perangkat,
5. *Broadband access* diperkenalkan dan digunakan,
6. *E-commerce*, *e-transaction* belum populer.

Di masa mendatang tren gaya hidup akan memiliki karakteristik sebagai berikut :

1. Setiap orang dan segala sesuatu terhubung secara di mana saja dan kapan saja (komunikasi dinamis tanpa batas).
2. Tempat utama mencari *content* adalah *on-line*.
3. Koneksi dengan pita lebar sudah umum baik *fixed* maupun bergerak.
4. *E-commerce*, *e-transaction* menjadi hal yang utama.

2.2. Android

2.2.1 Definisi Android

Android adalah sistem operasi untuk telepon seluler yang berbasis Linux[2]. Android menyediakan platform terbuka bagi para pengembang buat menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak. Awalnya, Google Inc. membeli Android Inc., pendatang baru yang membuat peranti lunak untuk ponsel. Kemudian untuk mengembangkan Android, dibentuklah *Open Handset Alliance*, konsorsium dari 34 perusahaan peranti keras, peranti lunak, dan telekomunikasi, termasuk Google, HTC, Intel, Motorola, Qualcomm, T-Mobile, dan Nvidia.

[2] Nasrudin Safaat H., *Pemrograman Aplikasi Mobile Smarthphone dan Tablet PC Berbasis Android* , Informatika, Bandung, 2011, Hal 1

2.2.2 Sejarah Android

Pada perilisan perdana Android, 5 November 2007, Android bersama *Open Handset Alliance* menyatakan mendukung pengembangan *open source* pada perangkat mobile. Di lain pihak, Google merilis kode-kode Android di bawah lisensi Apache, sebuah lisensi perangkat lunak dan *open platform* perangkat seluler.

Di dunia ini terdapat dua jenis distributor sistem operasi Android. Pertama yang mendapat dukungan penuh dari Google atau *Google Mail Services* (GMS) dan kedua adalah benar-benar bebas distribusinya tanpa dukungan langsung Google atau dikenal dengan *Open Handset Distribution* (OHD).

Sekitar September 2007 Google mengenalkan Nexus One, salah satu jenis *Smartphone* yang menggunakan Android sebagai system operasinya. Telepon seluler ini diproduksi oleh HTC Corporation dan tersedia di pasaran pada 5 Januari 2008. Pada Desember 2008, diumumkan anggota baru yang bergabung dalam program kerja Android ARM Holdings, Atheros, Softbank, Sony Ericson, Toshiba Corp, dan Vodafone Group Plc. Seiring pembentukan *Open Handset Alliance*, OHA mengumumkan produk perdana mereka Android, perangkat mobile yang merupakan modifikasi kernel Linux 2.6. Sejak android dirilis telah dilakukan berbagai pembaruan berupa perbaikan bug dan penambahan fitur baru.

Pada saat ini kebanyakan vendor-vendor *smartphone* sudah memproduksi *smarthphone* berbasis Android, vendor-vendor itu Antara lain HTC, LG, Motorola, Samsung, HKC, Huawei, Archos, Webstation Camangi, Dell, Nexus,

SciPhone, WayteQ, Sony Ericson, Acer, Philips, T-Mobile, Nexian, IMO, Asus, dan masih banyak lagi vendor *smartphone* didunia yang memproduksi Android. Hal ini karena Android adalah sistem operasi *open source* sehingga bebas didistribusikan dan dipakai oleh vendor manapun.

Tidak hanya menjadi sistem operasi di *smartphone*, saat ini Android menjadi pesaing utama dari Apple pada system operasi Tablet PC. Pesatnya pertumbuhan Android selain faktor yang disebutkan di atas adalah karena Android itu sendiri adalah platform yang sangat lengkap baik itu sistem operasinya, Aplikasi dan Tool pengembangan, market aplikasi Android serta dukungan yang sangat tinggi dari komunitas *open source* di Dunia, sehingga android terus berkembang pesat baik dari segi teknologi maupun dari segi jumlah *device* yang ada di dunia.

2.2.3 Arsitektur Android

Secara garis besar Arsitektur Android dapat dijelaskan dan digambarkan sebagai berikut:

2.2.3.1 *Appplication and Widgets*

Aplication dan *Widgets* ini adalah *layer* dimana kita berhubungan dengan aplikasi saja, di mana biasanya kita *download* aplikasi kemudian kita lakukan instalasi dan jalankan aplikasi tersebut. Di *layer* terdapat aplikasi inti termasuk klien email, program SMS, kalender, peta, browser, kontak, dan lain-lain. Semua aplikasi di tulis menggunakan bahasa pemrograman Java.

2.2.3.2 *Aplication Framework*

Android adalah "*Open Development Platform*" yaitu Android menawarkan kepada pengembang atau memberi kemampuan kepada pengembang untuk membangun aplikasi yang bagus dan inovatif. Pengembang bebas untuk mengakses perangkat keras, akses informasi *resources*, menjalankan *service background*, mengatur alarm, dan menambahkan status *notifications*, dan sebagainya. Pengembang memiliki akses penuh menuju API *framework* seperti yang dilakukan oleh aplikasi kategori inti. Arsitektur aplikasi dirancang supaya kita dengan mudah dapat menggunakan kembali komponen yang sudah digunakan (*reuse)*.

Sehingga bisa kita simpulkan *Aplications Framework* ini adalah *layer* di mana para pembuat aplikasi melakukan pengembangan/pembuatan aplikasi yang akan dijalankan di sistem operasi Android, Karena pada *layer* inilah aplikasi dapat dirancang dan dibuat, seperti *content-providers* yang berupa sms dan panggilan telepon.

Komponen-komponen yang termasuk di dalam applications Framework adalah sebagai berikut:

1. *views*
2. *content-providers*
3. *resource manager*
4. *notifications manager*
5. *activity manager*

2.2.3.3 *Libraries*

Libraries adalah *layer* di mana fitur-fitur Android berada, biasanya para pembuat aplikasi mengakses *libraries* untuk menjalankan aplikasinnya. Berjalan di atas kernel, *Layer* ini meliputi berbagai *library* C/C++ inti seperti Libc dan SSL, serta *libraries* media untuk pemutaran media audio dan video, *libraries* untuk manajemen tampilan, *libraries* Graphics mencakup SGL dan OpenGL untuk grafis 2D dan 3D, *libraries* SQLite untuk dukungan database, *libraries* SSL dan *WebKit* terintegrasi dengan *web broser* dan security, *libraries liveWebcore* mencakup *modern web browser* dengan *engine embedded web view*, *libraries* 3D yang mencakup implementasi OpenGL ES 1.0 API's.

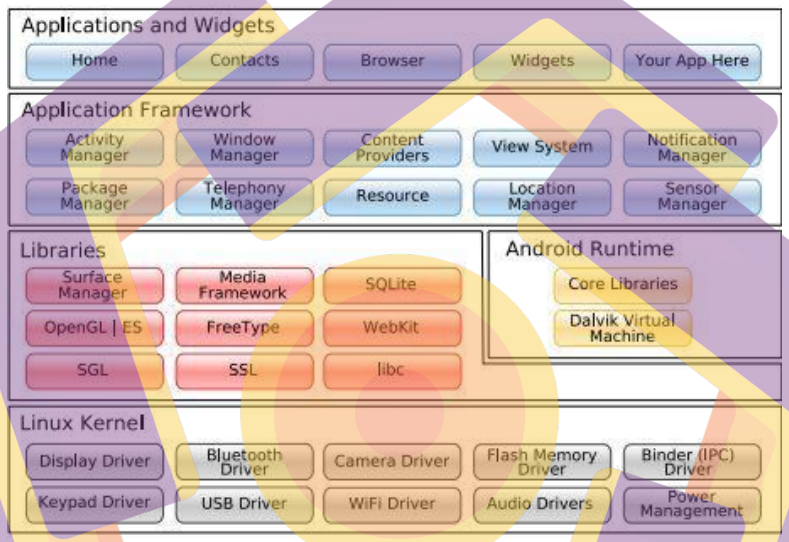
2.2.3.4 *Android Run Time*

Layer yang membuat aplikasi Android dapat dijalankan di manapun dalam prosesnya menggunakan implementasi Linux. *Dalvik Virtual Machine* (DVM)merupakan mesin yang membentuk dasar kerangka aplikasi Android. Di dalam Android Run Time dibagi menjadi dua bagian yaitu:

1. *Core Libraries*: Aplikasi Android dibangun dalam bahasa java, sementara Dalvik sebagai virtual mesinnya bukan *Virtual Machine Java*, sehingga diperlukan sebuah *libraries* yang berfungsi untuk menterjemahkan bahasa java/c yang ditangani oleh *core libraries*.
2. *Dalvik Virtual Machine*: Virtual mesin berbasis *register* yang dioptimalkan untuk menjalankan fungsi-fungsi secara efisien, dimana merupakan pengembangan yang mampu membuat linux kernel untuk melakukan *threading* dan manajemen tingkat rendah.

2.2.3.5 *Linux Kernel*

Linux kernel adalah *layer* dimana inti dari operating sistem dari Android itu berada. Berisi file-file system yang mengatur sistem *processing, memory, resource, drivers,* dan sistem-sistem operasi android lainnya. Linux Kernel yang digunakan Android adalah Linux-Kernel release 2.6.



Gambar 2.1 Arsitektur Android

2.2.4 Fundamental Aplikasi

Aplikasi Android ditulis dalam bahasa pemrograman java. Kode java dikompilasi bersama dengan data *file resource* yang dibutuhkan oleh aplikasi, di mana prosesnya di-*package* oleh *tools* yang dinamakan "apt-tools" ke dalam paket Android sehingga menghasikan file dengan ekstensi apk. File apk itulah yang di sebut dengan aplikasi, dan nantinya dapat diinstall pada perangkat mobile.

Ada empat jenis komponen pada aplikasi Android yaitu:

1. ***Activities***

Suatu *activity* akan menyajikan *user interface* (UI) kepada pengguna, sehingga pengguna dapat melakukan interaksi. Sebuah aplikasi Android bisa jadi hanya memiliki satu *activity*, tetapi umumnya aplikasi memiliki banyak *activity* tergantung pada tujuan aplikasi dan desain dari aplikasi tersebut. Satu *activity* biasanya akan dipakai untuk menampilkan aplikasi atau yang bertindak sebagai *user interface* (UI) saat aplikasi diperlihatkan kepada *user*. Untuk pindah dari satu *activity* ke *activity* lain dapat dilakukan dengan satu *event*, misalnya klik tombol, memilih opsi atau *triggers* tertentu. Secara hirarki sebuah *windows activity* dinyatakan sebagai *method* Activity.setContentView(). ContentView adalah objek yang berada pada root hirarki.

2. *Servive*

Service tidak memiliki *Graphic User Interface* (GUI), tetapi *service* berjalan secara *background*, sebagai contoh dalam memainkan musik, *service* mungkin memainkan musik atau mengambil data dari jaringan, tetapi setiap *service* harus berada dalam kelas induknya. Misalnya, *media player* sedang memutar lagu dari *list* yang ada, aplikasi ini akan memiliki dua atau lebih *activity* yang memungkinkan *user* untuk memilih lagu misalnya, atau menulis sms sambil *player* sedang jalan. Untuk menjaga musik tetap dijalankan, *activity player* dapat menjalankan *service*. *Service* dijalankan pada *thread* utama dari proses aplikasi.

3. *Broadcast Receiver*

Broadcast receiver berfungsi menerima dan bereaksi untuk menyampaikan notifikasi. Contoh *broadcast* seperti notifikasi zona waktu berubah, baterai *low*, gambar telah selesai, diambil oleh kamera, atau pengubahan referensi bahasa yang

digunakan. Aplikasi juga dapat menginisiasi *broadcast* misalnya memberikan informasi pada aplikasi lain bahwa ada data yang telah diunduh ke perangkat dan siap untuk digunakan. *Broadcast receiver* tidak memiliki *user interface* (UI), tetapi memiliki sebuah *activity* untuk merespon informasi yang mereka terima, atau mungkin menggunakan *notifications manager* untuk memberitahu kepada pengguna, seperti lampu latar atau *vibrating* (getaran) perangkat, dan lain sebagainya.

4. *Content Provider*

Content provider membuat kumpulan aplikasi data secara spesifik sehingga bisa digunakan oleh aplikasi lain. Data disimpan dalam file sistem seperti database SQLite. *Content provider* menyediakan cara untuk mengakses data yang dibutuhkan oleh suatu *activity*, misalnya ketika kita menggunakan aplikasi yang membutuhkan peta (*map*), atau aplikasi yang membutuhkan akses data kontak dan navigasi, maka di sinilah fungsi *content provider*.

2.2.5 DVM (*The Dalvik Virtual Machine*)

Salah satu elemen kunci dari Android adalah *Dalvik Virtual Machine* (DVM). Android berjalan pada *Dalvik Virtual Machine* (DVM) bukan di *Java Virtual Machine* (JVM), sebenarnya banyak persamaan dengan *Java Virtual Machine* (JVM) seperti Java ME (*Java Mobile Edition*), tetapi Android menggunakan *Virtual Machine* sendiri yang dikustomisasi dan dirancang untuk memastikan bahwa beberapa *feature* berjalan lebih efisien pada perangkat *mobile*.

Dalvik Virtual Machine (DVM) adalah "*register based*" sementara *Java Virtual Machine* (JVM) adalah "*stack based*", DVM didesain dan ditulis oleh Dan

Bornstern dan beberapa *engineers* Google lainnya. *Dalvik Virtual Machine* menggunakan kernel Linux untuk menangani fungsionalitas tingkat rendah termasuk keamanan, *threading*, dan proses serta manajemen memori. Ini memungkinkan kita untuk menulis Aplikasi C/C++ sama halnya seperti pada OS Linux kebanyakan. Meskipun dalam kenyataan kita harus banyak memahami Arsitektur dalam proses sistem dari kernel linux yang digunakan dalam android tersebut.

Semua *hardware* yang berbasis android dijalankan dengan menggunakan *Virtual Machine* untuk eksekusi aplikasi, pengembang tidak perlu khawatir tentang implementasi perangkat keras tertentu. *Dalvik Virtual Machine* mengeksekusi *executable file*, sebuah format yang dioptimalkan untuk memastikan memori yang digunakan sangat kecil. *The executable file* diciptakan dengan mengubah kelas bahasa Java dan dikompilasi menggunakan *tools* yang disediakan dalam SDK Android.

2.2.6 Versi Android

Secara garis besar Arsitektur Android dapat dijelaskan dan digambarkan sebagai berikut:

2.2.6.1 Android Versi 1.1

Pada 9 Maret 2009, Google merilis Android versi 1.1. Android versi ini dilengkapi dengan pembaruan estetis pada aplikasi, jam alarm, *voice search* (pencarian suara), pengiriman pesan dengan Gmail, dan pemberitahuan email.

2.2.6.2 Android Versi 1.5 (Cupcake)

Pada pertengahan Mei 2009, Google kembali merilis telepon seluler dengan menggunakan Android dan SDK (*Software Development Kit*) dengan versi 1.5 (Cupcake). Terdapat beberapa pembaruan termasuk juga penambahan beberapa fitur dalam seluler versi ini yakni kemampuan merekam dan menonton video dengan modus kamera, mengunggah video ke Youtube dan gambar ke Picasa langsung dari telepon, dukungan Bluetooth A2DP, kemampuan terhubung secara otomatis ke headset *Bluetooth*, animasi layar, dan *keyboard* pada layar yang dapat disesuaikan dengan sistem.

2.2.6.3` Android Versi 1.6 (Donut)

Donut (versi 1.6) dirilis pada September dengan menampilkan proses pencarian yang lebih baik dibanding sebelumnya, penggunaan baterai indikator dan kontrol applet VPN. Fitur lainnya adalah galeri yang memungkinkan pengguna untuk memilih foto yang akan dihapus; kamera, *camcorder* dan galeri yang dintegrasikan, CDMA / EVDO, 802.1x, VPN, Gestures, dan Text-to-speech engine; kemampuan dial kontak; teknologi *text to change speech* (tidak tersedia pada semua ponsel, pengadaan resolusi VWGA.

2.2.6.4 Android Versi 2.0/2.1 (Éclair)

Pada 3 Desember 2009 kembali diluncurkan ponsel Android dengan versi 2.0/2.1 (Eclair), perubahan yang dilakukan adalah pengoptimalan *hardware*, peningkatan Google Maps 3.1.2, perubahan UI dengan *browser* baru dan dukungan HTML5, daftar kontak yang baru, dukungan *flash* untuk kamera 3,2 MP, *digital Zoom*, dan *Bluetooth* 2.1.

2.2.6.5 Android Versi 2.2 (Froyo: Frozen Yogurt)

Pada 20 Mei 2010, Android versi 2.2 (Froyo) diluncurkan. Perubahan-perubahan umumnya terhadap versi-versi sebelumnya antara lain dukungan Adobe Flash 10.1, kecepatan kinerja dan aplikasi 2 sampai 5 kali lebih cepat, intergrasi V8 JavaScript *engine* yang dipakai Google Chrome yang mempercepat kemampuan *rendering* pada *browser*, pemasangan aplikasi dalam SD *Card*, kemampuan WiFi *Hotspot portabel*, dan kemampuan *auto update* dalam aplikasi Android Market.

2.2.6.6 Android Versi 2.3 (Ginggerbread)

Pada 6 Desember 2010, Android versi 2.3 (Gingerbread) diluncurkan. Perubahan-perubahan umum yang didapat dari Android versi ini antara lain peningkatan kemampuan permainan (*gaming*), peningkatan fungsi *copy paste*, layar antar muka (*User Interface*) didesain ulang, dukungan format video VP8 dan WebM, efek audio baru (*reverb, equalization, headphone virtualization*, dan *bass boost*), dukungan kemampuan *Near Field Communication* (NFC), dan dukungan jumlah kamera yang lebih dari satu.

2.2.6.7 Android Versi 3.0/3.1 (Honeycomb)

Android Honeycomb dirancang khusus untuk tablet. Android versi ini mendukung ukuran layar yang lebih besar. *User Interface* pada Honeycomb juga berbeda karena sudah didesain untuk tablet. Honeycomb juga mendukung multi prosesor dan juga akselerasi perangkat keras (*hardware*) untuk grafis.

2.2.6.8 Android Versi 4.0 (ICS: Ice Cream Sandwich)

Android ICS membawa fitur Honeycomb untuk *smartphone* dan menambah fitur baru termasuk membuka kunci dengan pengenal wajah, jaringan data pemantauan penggunaan dan kontrol, terpadu kontak jaringan sosial, perangkat tambahan fotografi, mencari email secara *offline*, dan berbagi informasi menggunakan NFC.

2.2.6.9 Android Versi 4.1 (Jelly Bean)

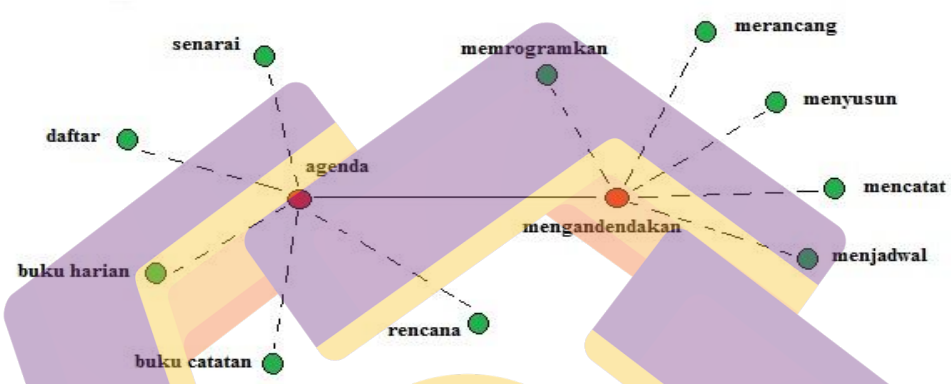
Google secara resmi telah memamerkan *Jelly Bean* sebagai generasi terbaru dari Android. Sistem operasi ini pun disebut-sebut punya sejumlah fitur yang telah disempurnakan dibandingkan *Ice Cream Sanwich*. Salah satunya *Project Butter* dalam *Jelly Bean* tak hanya mengubah tampilan Android lebih menarik, namun diklaim juga dirancang untuk mengoptimalkan kemampuan *System on Chip* (SoC) pada tiap-tiap ponsel.

Perbedaan lain dari *Jelly Bean* dengan sistem operasi sebelumnya adalah penanaman sistem *Adress Space Layout Randomization* (ASLR). Bersama dengan fitur keamanan lain yang disebut *Data Executive Prevention* (DEP), sistem ini dapat mengacak lokasi pada memori perangkat. Ketika ASLR dan DEP dikombinasikan, serangan dengan modus ini dapat dipentalkan karena perentas tidak dapat meletakan program jahat dalam memori perangkat. *Jelly Bean* memiliki pertahanan terhadap upaya pembocoran informasi, *buffer overflow*, dan beberapa celah lain yang berkaitan dengan memori[3].

2.3. Pengertian Agenda

[3] "Jelly Bean Versi Android Teraman." Tempo.co. http://www.tempo.co/read/news/2012/07/18/072417787/Jelly-Bean-Versi-Android-Teraman [18 Desember 2012]

Agenda adalah buku catatan yang bertanggal untuk satu bulan atau satu tahun. Isinya sesuatu yang akan dilakukan, sesuatu yang akan dibicarakan, sesuatu yang sedang terjadi dan sesuatu yang sudah terjadi.



Gambar 2.2 Agenda

Sumber: http://www.artikata.com/arti-3738-agenda.htm [19 Desember 2012]

2.3.1 Macam-macam Contoh Aplikasi Agenda Pribadi

2.3.1.1 Evernote

Aplikasi Evernote sangat *powerful* digunakan sebagai catatan *portable* pada perangkat yang anda gunakan. Kelebihan yang ditawarkan adalah dapat disinkronisasikan pada berbagai perangkat yang didukung oleh Evernote, dan catatan yang dibuat kemudian akan disimpan dalam *server* (data awan) Evernote. Kelebihan lain yang ditawarkan antara lain (Mulyana, D. Hendrik: 2012:32) :

1. Tersedia di berbagai perangkat smartphone, di antaranya iPhone, iPod, Blackberry, Android, iPad, Palm WebOS, Windows Mobile dan tersedia juga pada perangkat aplikasi dekstop Windows, Mac serta Evernote Web Application (IE, Firefox, Chrome, Safari).

2. Mendukung penyimpanan gambar, foto, file audio, dan file-file lainnya seperti file Office (Word, Excel, PowerPoint, PDF khusus pengguna akun Premium).
3. Mendukung pencarian note dengan mudah, bahkan Evernote dapat menemukan teks yang ada pada gambar.
4. Note tersinkronisasi ke semua jenis *platform* dan *device* yang didukung oleh Evernote.
5. Mendukung Share Note (berbagi catatan) via Facebook dan Twitter.
6. Dilindungi dengan enkripsi SSL yang akan menjamin keamanan data note.

Dengan Evernote, kita dapat mencatat momen, dan note akan langsung tersimpan di server Evernote dengan aman. Note tersebut bisa kita akses kapan pun serta si manapun kita berada, selama ada koneksi internet dan device, serta platform yang didukung oleh Evernote.

2.3.1.2 Do It Tomorrow

Aplikasi DO IT Tomorrow adalah aplikasi sederhana pengingat pesan atau reminder. Aplikasi ini sangat nyaman digunakan dengan tampilan layaknya catatan notebook, namun dalam bentuk virtual perangkat ponsel android.

Kelebihan yang ditawarkan pada aplikasi sederhana ini adalah (Mulyana, D. Hendrik: 2012:33) :

1. Tampilan User Interface layaknya catatan notebook.
2. Cepat dan mudah membuat catatan.
3. Mudah memeriksa berbagai *task* yang telah selesai dikerjakan atau menghapusnya dalam list to do.

4. Tiga *widget* yang dapat digunakan di layar *homepage* (khusus untuk versi berbayar).
5. Mendukung sinkronisasi catatan Do It Tomorrow versi website.

Aplikasi ini hanya menyediakan aplikasi pembuat catatan pengingat untuk dilakukan saat ini atau esok hari sesuai dengan nama aplikasinya, Do It Tomorrow. Aplikasi ini juga tersedia untuk perangkat iPhone, iPad dan aplikasi web.

2.3.1.3 Spring Pad

Spring pad adalah aplikasi multifungsi yang tidak hanya dapat digunakan untuk membuka catatan saja, seperti slogan aplikasinya *"the smart note app"* – aplikasi note cerdas. Aplikasi ini dapat anda gunakan untuk menyimpan catatan pendek, seperti catatan resep, belanjaan, reminder dan lainnya. Anda juga dapat menggunakan aplikasi ini bukan hanya sekedar menyimpan catatan saja, anda dapat mencari berbagai informasi, misal mencari nama restoran, judul film, resep masakan hingga produk dalam menu *look up*, dan dapat diintegrasikan ke dalam sebuah note yang anda buat, hingga foto, audio dapat dimasukan ke dalam note.

Berikut beberapa fitur yang dapat digunakan pada aplikasi Springpad (Mulyana, D. Hendrik: 2012:36) :

1. Mudah membuat catatan dan beerbagai informasi dalam satu aplikasi.
2. Mendukung membuat *task* dan *list to do (reminder)*.
3. Mencari informasi secara instan menggunakan menu look up (tempat, judul film, resep, produk dan lain-lain) dan dapat diintegrasikan dalam sebuah catatan.

4. Mudah mengintegrasikan foto atau file audio dalam note.
5. Mudah menambahkan tag, flags dan alert.
6. Scan barcode secara instan dalam satu aplikasi dan mendukung proses sinkronikasi.

Aplikasi Springpad juga tersedia untuk versi Tablet, iPad, iPhone hingga versi website. Aplikasi Springpad juga dapat menyimpan semua data catatan anda ke dalam server cloud miliknya, sehingga dapat memudahkan anda untuk selalu terkoneksi dengan semua catatan, baik perangkat ponsel lain atau website yang telah didukung oleh aplikasi tersebut. Kelebihan lainnya adalah berbagai catatan dapat diintegrasikan dengan berbagai file, baik foto, audio (memo) link, hingga video, dan aplikasi ini juga dapat digunakan sebagai catatan *to do* (*reminder*).

2.3.1.4 Any.Do

Make Things happen begitulah slogan dari aplikasi bernama Any.Do. aplikasi Any.Do adalah aplikasi To Do list (Reminder) yang dapat digunakan untuk keperluan pengingat pesan, baik sebagai catatan singkat ataupun pengingat janji. Aplikasi ini dapat diperoleh secara gratis di anroid store. Aplikasi ini sangat simpel, sederhana dan mudah digunakan.

Kesan pertama pada saat menggunakan aplikasi ini adalah sangat elegan, tidak terdapat banyak menu yang terdapat di dalamnya. Di bagian atas aplikasi terdapat form seperti kotak pencarian yang berfungsi untuk membuat catatan singkat yang bertuliskan *i want to* dan dilengkapi dengan sebuah *microphone* yang dapat digunakan untuk membuat pesan menggunakan *microphone* pada ponsel. Bagian bawah terdapat kategori penjadwalan menu, seperti *Today*,

Tomorrow, *This Week* dan *Later* yang dapat digunakan untuk mengelola *to do list* yang akan dibuat.

Berikut beberapa fitur menarik pada palikasi Any.DO (Mulyana, D. Hendrik: 2012:38) :

1. Akses task lebih cepat, dilengkapi dengan fitur *Voice Recognition*.
2. Mendukung *Drag & Drop Task* Agenda.
3. Mendukung penyortiran berdasarkan tanggal, folder atau berdasarkan prioritas.
4. Mendukung penambahan catatan pada sub-task.
5. Mendukung penghapusan catatan dengan *Shake to Task*.
6. Dapat mem-backup task pada SD Card.
7. Mendukung sinkronisasi dengan akun Google.

2.3.2 Agenda Pribadi "enot"

Enot adalah aplikasi agenda pribadi yang *use able* dengan fitur-fitur sederhana dan praktis yang akan memberi kemudahan bagi seseorang mengorganisasikan maupun mengabadikan momen atau kejadian. Enot dilengkapi dengan pengaman (*ussername* dan *password*) yang memang mendukung *privacy* seseorang, jadi tidak semua ornag bisa mengakses agenda pribadi *user*.

Ada 4 menu utama yang bisa user gunakan, berikut :

1. Menu Foto

   User dapat memilih foto yang ada di Gallery, kemudian dapat langsung mendeskripsikan gambar.

2. Menu Suara

User bisa menggunakan menu ini dengan inputan berupa suara yang nantinya akan disimpan di *sdcard*.

3. Text

Teks bisa menjadi pilihan praktis saat user ingin menulis sendiri agenda yang ingin disusun.

4. Calender

Di menu kalender, user bisa melihat 1 bulan penuh kalender dan susunan kegiatan yang sudah diinputkan. Susunan kegiatan tersusun berdasarkan waktu penyimpanan. User bisa juga mengedit, menghapus dan berbagi. Koneksi internet dapat dimanfaatkan user untuk berbagi beberapa kegiatan atau momen di beberapa jejaring sosial.

2.4. Perangkat Lunak Yang Digunakan

2.4.1 Eclipse IDE

Eclipse adalah sebuah IDE (*Integrated Development Environment*) untuk mengembangkan perangkat lunak dan dapat dijalankan di semua platform (*platform-independent*). Berikut ini adalah sifat dari Eclipse:

1. *Multi-Platform*: Target sistem operasi Eclipse adalah Microsoft Windows, Linux, Solaris, AIX, HP-UX dan Mac OS X.
2. *Multi-language*: Eclipse dikembangkan dengan bahasa pemrograman Java, akan tetapi Eclipse mendukung pengembangan aplikasi berbasis bahasa pemrograman lainnya, seperti C/C++, Cobol, Python, Perl, PHP, dan lain sebagainya.

3. *Multi-Role*: Selain sebagai IDE untuk pengembangan aplikasi, Eclipse pun bisa digunakan untuk aktivitas dalam siklus pengembangan perangkat lunak, seperti dokumentasi, *test* perangkat lunak, pengembangan web, dan lain sebagainya.

Eclipse pada saat ini merupakan salah satu IDE favorit dikarenakan gratis dan *open source*, yang berarti setiap orang boleh melihat kode pemrograman perangkat lunak ini. Selain itu, kelebihan dari Eclipse yang membuatnya populer adalah kemampuannya untuk dapat dikembangkan oleh pengguna dengan komponen yang dinamakan *plug-in*.

Sejak versi 3.0, Eclipse pada dasarnya merupakan sebuah kernel, yang mengangkat *plug-in*. Apa yang dapat digunakan di dalam Eclipse sebenarnya adalah fungsi dari *plug-in* yang sudah diinstal. Ini merupakan basis dari Eclipse yang dinamakan *Rich Client Platform* (RCP). Berikut ini adalah komponen yang membentuk RCP:

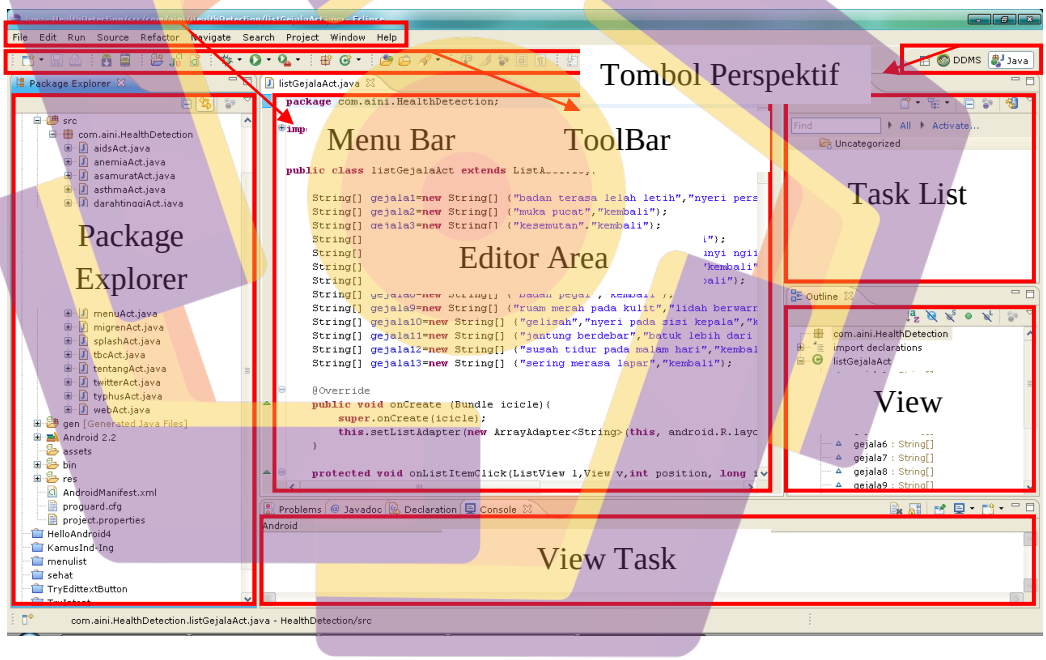
1. *Core Platform*
2. OSGi
3. SWT (*Standard Widget Toolkit*)
4. JFace
5. *Eclipse Workbench*

Secara standar Eclipse selalu dilengkapi dengan JDT (*Java Development Tools*), *plug-in* yang membuat Eclipse kompatibel untuk mengembangkan program Java, dan PDE (*Plug-in Development Environment*) untuk

mengembangkan *plug-in* baru. Eclipse beserta *plug-in* nya diimplementasikan dalam bahasa pemrograman Java.

Konsep Eclipse adalah IDE yang terbuka (*open*), mudah diperluas (*extensible*) untuk apa saja, dan tidak untuk sesuatu yang spesifik. Jadi, Eclipse tidak saja untuk mengembangkan program Java, akan tetapi dapat digunakan untuk berbagai macam keperluan, cukup dengan menginstal plug-in yang dibutuhkan.

Berikut ini adalah tampilan interface workbench eclipse.



Gambar 2.3 Tampilan Workbench Eclipse

2.4.2 Java

2.4.2.1 Pengertian Java

Java adalah bahasa pemrograman yang dapat dijalankan di berbagai komputer termasuk telepon genggam. Bahasa ini awalnya dibuat oleh James Gosling saat masih bergabung di Sun Microsystems saat ini merupakan bagian

dari Oracle dan dirilis tahun 1995. Bahasa ini banyak mengadopsi sintaksis yang terdapat pada C dan C++ namun dengan sintaksis model objek yang lebih sederhana serta dukungan rutin-rutin aras bawah yang minimal. Aplikasi-aplikasi berbasis java umumnya dikompilasi ke dalam p-code (*bytecode*) dan dapat dijalankan pada berbagai Mesin Virtual Java (JVM). Java merupakan bahasa pemrograman yang bersifat umum/non-spesifik (*general purpose*), dan secara khusus didisain untuk memanfaatkan dependensi implementasi seminimal mungkin. Karena fungsionalitasnya yang memungkinkan aplikasi java mampu berjalan di beberapa *platform* sistem operasi yang berbeda, java dikenal pula dengan slogannya, "Tulis sekali, jalankan di mana pun". Saat ini java merupakan bahasa pemrograman yang paling populer digunakan, dan secara luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak aplikasi ataupun aplikasi berbasis web.

2.4.2.2 Versi Awal Java

Versi awal Java ditahun 1996 sudah merupakan versi *release* sehingga dinamakan Java Versi 1.0. Java versi ini menyertakan banyak paket standar awal yang terus dikembangkan pada versi selanjutnya:

1. java.lang: peruntukan kelas elemen-elemen dasar.
2. java.io: peruntukan kelas *input* dan *output*, termasuk penggunaan berkas.
3. java.util: Peruntukan kelas pelengkap seperti kelas struktur data dan kelas kelas penanggalan.
4. java.net: Peruntukan kelas TCP/IP, yang memungkinkan berkomunikasi dengan komputer lain menggunakan jaringan TCP/IP.

5. java.awt: Kelas dasar untuk aplikasi antarmuka dengan pengguna (GUI).
6. java.applet: Kelas dasar aplikasi antar muka untuk diterapkan pada penjelajah web.

2.4.2.3 Kelebihan Java

Bahasa pemrograman Java memiliki banyak kelebihan di antaranya:

1. *Multiplatform*

Kelebihan utama dari Java ialah dapat dijalankan di beberapa *platform* / sistem operasi komputer, sesuai dengan prinsip tulis sekali, jalankan di mana saja. Dengan kelebihan ini *programer* cukup menulis sebuah program Java dan dikompilasi (diubah, dari bahasa yang dimengerti manusia menjadi bahasa mesin / *bytecode*) sekali lalu hasilnya dapat dijalankan di atas beberapa platform tanpa perubahan. Kelebihan ini memungkinkan sebuah program berbasis java dikerjakan diatas *operating system* Linux tetapi dijalankan dengan baik di atas Microsoft Windows. *Platform* yang didukung sampai saat ini adalah Microsoft Windows, Linux, Mac OS dan Sun Solaris. Penyebanya adalah setiap sistem operasi menggunakan programnya sendiri-sendiri (yang dapat diunduh dari situs Java) untuk meninterpretasikan *bytecode* tersebut.

2. OOP (*Object Oriented Programing* – Pemrograman Berorientasi Objek)

Merupakan paradigma pemrograman yang berorientasikan kepada objek. Semua data dan fungsi di dalam paradigma ini dibungkus dalam kelas-kelas atau objek-objek. Bandingkan dengan logika pemrograman terstruktur. Setiap objek dapat menerima pesan, memproses data, dan mengirim pesan ke objek lainnya.

Model data berorientasi objek dikatakan dapat memberi fleksibilitas yang lebih, kemudahan mengubah program, dan digunakan luas dalam teknik piranti lunak skala besar. Lebih jauh lagi, pendukung OOP mengklaim bahwa OOP lebih mudah dipelajari bagi pemula dibanding dengan pendekatan sebelumnya, dan pendekatan OOP lebih mudah dikembangkan dan dirawat.

3. Perpustakaan Kelas Yang Lengkap

Kelengkapan *library* /perpustakaan (kumpulan program-program yang disertakan dalam pemrograman java) sangat memudahkan *programer* dalam membangun aplikasinya. Kelengkapan perpustakaan ini ditambah dengan keberadaan komunitas Java yang besar yang terus menerus membuat perpustakaan-perpustakaan baru untuk melingkupi seluruh kebutuhan pembangunan aplikasi.

4. Pengumpulan Sampah Otomatis

Java memiliki fasilitas pengaturan penggunaan memori sehingga para *programer* tidak perlu melakukan pengaturan memori secara langsung.

2.4.3 Android SDK (*Software Development Kit*)

Android SDK adalah tools API (*Application Programming Interface*) yang diperlukan untuk mulai mengembangkan aplikasi pada *platform* Android menggunakan bahasa pemrograman Java. Android merupakan subset perangkat lunak untuk ponsel yang meliputi sistem operasi, *middleware* dan aplikasi kunci yang di *release* oleh Google. Saat ini disediakan Android SDK (*software Development kit*) sebagai alat bantu dan API untuk mulai mengembangkan aplikasi pada *platform* Android menggunakan bahasa pemrograman Java. Sebagai

platform aplikasi-netral, Android memberi Anda kesempatan untuk membuat Aplikasi yang kita butuhkan yang bukan merupakan aplikasi bawaan *Handphone/Smartphone*. Beberapa fitur-fitur Android yang paling penting adalah:

1. *Framework* Aplikasi yang mendukung penggantian komponen dan *reusable*.
2. Mesin virtual Dalvik dioptimalkan untuk perangkat *mobile*.
3. *Integrated browser* berdasarkan *engine open source WebKit*.
4. Grafis yang dioptimalkan dan didukung oleh perpustakaan grafis 2D, grafis 3D berdasarkan spesifikasi opengl ES 1,0 (Opsional akselerasi hardware).
5. SQLite untuk penyimpanan data (*database*).
6. Media Support yang mendukung audio, video, dan gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF)GSM Telephony (tergantung *hardware*).
7. *Bluetooth*, EDGE, 3G, dan WiFi (tergantung *hardware*).
8. Kamera, GPS, kompas, dan *accelerometer* (*hardware* tergantung).
9. Lingkungan Development yang lengkap dan kaya termasuk perangkat *emulator*, *tools* untuk *debugging*, profil dan kinerja memori, dan plugin untuk IDE Eclipse.

Untuk source SDK Android ini dapat dilihat dan didownload langsung di situs resmi pengembang SDK Android di *http://www.developer.android.com* . SDK Android terdapat 2 versi baik versi windows maupun versi linux, karena SDK Android adalah *software* yang *free* dan bebas didistribusikan oleh semua kalangan.

2.4.4 Inkscape

Inkscape adalah sebuah perangkat lunak *editor* gambar vektor yang bersifat perangkat lunak bebas dibawah lisensi GNU GPL. Tujuan utama dari Inkscape adalah menjadi perangkat grafik mutakhir yang memenuhi standar XML, SVG, dan CSS.

Inkscape bersifat *cross-platform* dan dapat dijalankan pada Mac OS X (biasanya dibawah aplikasi X11, walaupun *toolkit* GTK+ yang digunakan dapat dikompilasikan untuk beroperasi secara langsung dibawah Quartz), sistem operasi berbasis Unix, dan Microsoft Windows. Implementasi SVG dan CSS di Inkscape belum sempurna, misalnya Inkscape tidak mendukung animasi SVG, dan font SVG, walaupun dukungan dasar untuk pembuatan font SVG telah diimplementasikan pada versi 0.47. Inkscape bersifat multibahasa, terutama untuk antarmuka dan script rumit, sesuatu yang sering terlewatkan pada sebagian besar editor grafik vektor komersil.

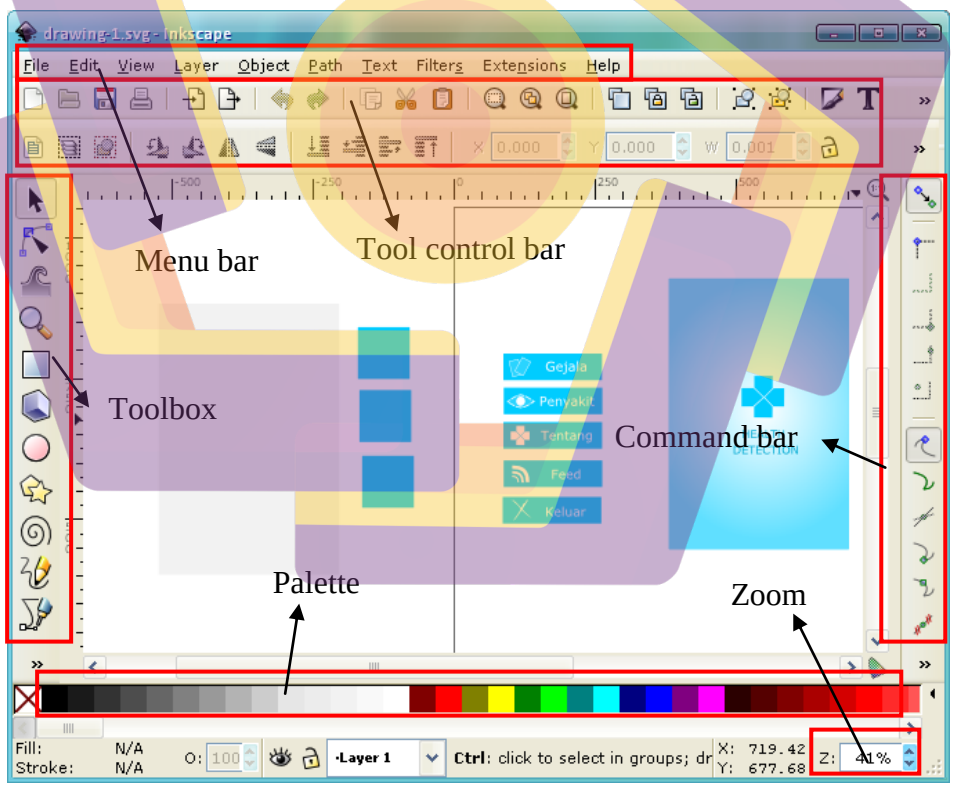
2.4.4.1 Sejarah Inkscape

Inkscape dirintis pada 2003 sebagai sebuah fork dari proyek Sodipodi. Sodipodi sendiri, yang dikembangkan sejak 1999, didasarkan pada Gill (*Gnome Illustration Application*), buah karya dari Raph Levien.

Fork itu dipandu oleh sebuah tim berjumlah empat orang yang beranggotakan mantan pengembang Sodipodi (Ted Gould, Bryce Harrington, Nathan Hurst, dan MenTaLguY) yang mengenali perbedaan-perbedaan dari tujuan-tujuan projek itu, keterbukaan bagi kontribusi pihak ketiga, dan ketidaksetujuan teknis sebagai alasan mereka melakukan forking. Dengan

Inkscape, mereka berpendapat bahwa mereka dapat memfokuskan pengembangan pada penerapan standar SVG secara lengkap, tidak seperti pengembangan Sodipodi yang menekankan pada pembuatan sebuah editor grafik vektor yang umum, dengan mengorbankan implementasi SVG.

Sejak dikembangkan dalam fork itu, Inkscape berganti dari menggunakan bahasa pemrograman C ke C++; berubah ke GTK+ toolkit C++ bindings (gtkmm); merancang ulang antarmuka pengguna dan menambahkan sejumlah fitur baru. Penerapannya terhadap standar SVG telah menunjukkan perbaikan yang signifikan, meski belum lengkap.



Gambar 2.4 Tampilan Inkscape

2.4.5 SQLite

SQLite merupakan mesin database SQL *embedded*. Tidak seperti kebanyakan database SQL lainnya, SQLite tidak memiliki proses server yang terpisah. SQLite membaca dan menulis secara langsung ke *disk*. Database SQL lengkap dengan *multiple* tabel, *indices*, *triggers*, dan *views*, semua terdapat dalam sebuah disk file tunggal. Format file database adalah *cross-platform* yaitu kita bebas mengcopy database antara 32-bit dan sistem 64-bit.

Tidak seperti pada paradigma *client-server* umumnya, Inti SQLite bukanlah sebuah sistem yang mandiri yang berkomunikasi dengan sebuah program, melainkan sebagai bagian integral dari sebuah program secara keseluruhan. Sehingga protokol komunikasi utama yang digunakan adalah melalui pemanggilan API secara langsung melalui bahasa pemrograman. Mekanisme seperti ini tentunya membawa keuntungan karena dapat mereduksi *overhead*, *latency times*, dan secara keseluruhan lebih sederhana. Seluruh elemen basisdata (definisi data, tabel, indeks, dan data) disimpan sebagai sebuah file. Kesederhanaan dari sisi disain tersebut bisa diraih dengan cara mengunci keseluruhan file basis data pada saat sebuah transaksi dimulai.

SQLite adalah salah satu *software* yang *embedded* yang sangat popular, kombinasi SQL *interface* dan penggunaan *memory* yang sangat sedikit dengan kecepatan yang sangat cepat[4].

[4] Nasrudin Safaat H., *Pemrograman Aplikasi Mobile Smarthphone dan Tablet PC Berbasis Android* , Informatika, Bandung, 2011, Halaman 177

2.5. Pengertian UML

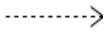
Unified Modeling Language (UML) merupakan sistem arsitektur yang bekerja dalam OOAD (*Object-Oriented Analysis/Design*) dengan satu bahasa yang konsisten untuk menentukan, visualisasi, mengkontruksi, dan mendokumentasikan artifak (sepotong informasi yang digunakan atau dihasilkan dalam suatu proses rekayasa *software*, dapat berupa model, deskripsi, atau *software*) yang terdapat dalam sistem *software*. UML merupakan bahasa pemodelan yang paling sukses dari tiga metode OO yang telah ada sebelumnya, yaitu Booch, OMT (*Object Modeling Technique*), dan OOSE (*Object-Oriented Software Engineering*).

2.5.1 *Use Case Diagram*

Use Case diagram adalah model fungsional sebuah sistem yang menggunakan *actor* dan use case. Setiap *Use Case* adalah suatu urut-urutan (*sequence*) transaksi yang saling berhubungan dan dilakukan oleh sebuah *actor* dan sistem dalam bentuk sebuah dialog. *Use Case Diagram* dibuat untuk memvisualisasikan/ menggambarkan hubungan antara *Actor* dan *Use Case*. *Use Case diagram* mempresentasikan kegunaan atau fungsi-fungsi sistem dari perspektif pengguna.

Tabel 2.2 Simbol *Use Case Diagram*

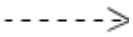
NO	GAMBAR	NAMA	KETERANGAN
1		*Actor*	Menspesifikasikan himpuan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.

2		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*).
3		*Generalization*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
4		*Include*	Menspesifikasikan bahwa *use case* sumber secara *eksplisit*.
5		*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
6		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7		*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8		*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor
9		*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
10		*Note*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi

2.5.2 *Class Diagram*

Class Diagram adalah suatu diagram yang memperlihatkan atau menampilkan struktur dari sebuah sistem, sistem tersebut akan menampilkan sistem kelas,atribut dan hubungan antara kelas ketika suatu sistem telah selesai membuat diagram. *Class diagram* menggambarkan struktur dan deskripsi *class*, *package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain.

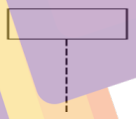
Tabel 2.3 Simbol Class Diagram

NO	GAMBAR	NAMA	KETERANGAN
1		*Generalization*	Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
2		*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
3		*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
4		*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor
5		*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek.
6		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri
7		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya

2.5.3 *Sequence Diagram*

Sequence diagram adalah *visual coding* (perancangan *form* / layar) dan merupakan interaksi objek yang tersusun dalam suatu urutan waktu / kejadian. Diagram ini secara khusus berasosiasi dengan *use case diagram*. *Sequence diagram* juga memperlihatkan tahap demi tahap apa yang seharusnya terjadi untuk menghasilkan sesuatu didalam *use case*. *Sequence diagram* juga dapat merubah atribut atau method pada *class* yang telah dibentuk oleh *class diagram*, bahkan menciptakan sebuah *class* baru. *Sequence diagram* memodelkan aliran logika dalam sebuah system dalam cara yang *visual*.

Tabel 2.4 Simbol *Sequence Diagram*

NO	GAMBAR	NAMA	KETERANGAN
1		*LifeLine*	Objek *entity*, antarmuka yang saling berinteraksi.
2		*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi
3		*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi

2.5.4 *Activity Diagram*

Activity diagram merupakan *state* diagram khusus, di mana sebagian besar state adalah action dan sebagian besar transisi di-trigger oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu *activity diagram* tidak menggambarkan *behaviour internal* sebuah sistem (dan interaksi antar subsistem)

secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.

Activity diagram menggambarkan berbagai alur aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi.

Tabel 2.5 Simbol *Activity Diagram*

NO	GAMBAR	NAMA	KETERANGAN
1		*Actifity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain
2		*Action*	State dari sistem yang mencerminkan eksekusi dari suatu aksi
3		*Initial Node*	Bagaimana objek dibentuk atau diawali.
4		*Actifity Final Node*	Bagaimana objek dibentuk dan dihancurkan
5		*Fork Node*	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran

2.6. ERD

ERD (*Entity Relationship Diagram*) merupakan suatu model untuk menjelaskan hubungan antar data dalam basis data berdasarkan objek-objek dasar data yang mempunyai hubungan antar relasi.

ERD untuk memodelkan struktur data dan hubungan antar data, untuk menggambarkannya digunakan beberapa notasi dan simbol. Pada dasarnya ada tiga simbol yang digunakan, yaitu :

1. Entiti

Entiti merupakan objek yang mewakili sesuatu yang nyata dan dapat dibedakan dari sesuatu yang lain. Simbol dari entiti ini biasanya digambarkan dengan persegi panjang.

2. Atribut

Setiap entitas pasti mempunyai elemen yang disebut atribut yang berfungsi untuk mendeskripsikan karakteristik dari entitas tersebut. Isi dari atribut mempunyai sesuatu yang dapat mengidentifikasikan isi elemen satu dengan yang lain. Gambar atribut diwakili oleh simbol elips.

3. Hubungan / Relasi

Hubungan antara sejumlah entitas yang berasal dari himpunan entitas yang berbeda. Relasi dapat digambarkan sebagai berikut :

a. Satu ke satu (*One to one*)

Hubungan relasi satu ke satu yaitu setiap entitas pada himpunan entitas A berhubungan paling banyak dengan satu entitas pada himpunan entitas B.

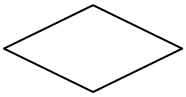
b. Satu ke banyak (*One to many*)

Setiap entitas pada himpunan entitas A dapat berhubungan dengan banyak entitas pada himpunan entitas B, tetapi setiap entitas pada entitas B dapat berhubungan dengan satu entitas pada himpunan entitas A.

c. Banyak ke banyak (*Many to many*)

Setiap entitas pada himpunan entitas A dapat berhubungan dengan banyak entitas pada himpunan entitas B.

Tabel 2.6 Simbol ERD

SIMBOL	NAMA
	Entitas, adalah suatu objek yang dapat diidentifikasi dalam lingkungan pemakai.
	Relasi, menunjukkan adanya hubungan di antara sejumlah entitas yang berbeda.
	Atribut, berfungsi mendeskripsikan karakter entitas (atribut yg berfungsi sebagai key diberi garis bawah)
	Garis, sebagai penghubung antara relasi dengan entitas, relasi dan entitas dengan atribut.

2.7. Database

Database merupakan sekumpulan data yang saling terintegrasi satu sama lain dan terorganisasi berdasarkan sebuah skema atau struktur tertentu dan terimpan pada sebuah *hardware* komputer[5]. Database terdiri dari beberapa tabel (lebih dari satu tabel) yang saling terorganisir. Tabel digunakan untuk menyimpan data dan terdiri dari baris dan kolom. Data tersebut dapat ditampilkan, dimodifikasi, dan dihapus dari tabel. Setiap pemakai (*user*) yang diberi wewenang (otorisasi) saja yang dapat melakukan akses terhadap data tersebut.

Database mempunyai beberapa kriteria penting, yaitu:

1. Bersifat data *oriented* dan bukan *program oriented*.
2. Dapat digunakan oleh beberapa program aplikasi tanpa perlu mengubah basis datanya.
3. Dapat dikembangkan dengan mudah, baik volume maupun strukturnya.

[5] M. Rudyanto Arief, *Pemrograman Basis Data Menggunakan Transact-SQL dengan Microsoft SQL Server 2000* , Andi Offset, Yogyakarta, 2006, Hal 33

4. Dapat memenuhi kebutuhan sistem-sistem baru secara mudah.
5. Dapat digunakan dengan cara-cara yang berbeda.

Prinsip utama Database adalah pengaturan data dengan tujuan utama fleksibelitas dan kecepatan pada saat pengambilan data kembali. Adapun ciri-ciri basis data diantaranya adalah sebagai berikut:

1. Efisiensi meliputi kecepatan, ukuran, dan ketepatan.
2. Data dalam jumlah besar.
3. Berbagi Pakai (dipakai bersama sama/*Sharebility*).
4. Mengurangi bahkan menghilangkan terjadinya duplikasi dan ketidakkonsistenan data.

2.8. Konsep Siklus Hidup Sistem

Dalam pembangunan sistem, setiap *software engineer* harus memiliki Model Proses yang paling tepat sehingga dia mendapatkan kemudahan dalam mengelola dan mengendalikan proses pembangunannya tahap demi tahap. Model proses disebut juga dengan aliran kerja (*workflow*), yakni tata cara bagaimana elemen-elemen proses berhubungan satu dengan lainnya[6].

Aliran kerja ini dapat juga disebut dengan siklus hidup (*life-cycle*) sistem yang dimulai dari sejak sistem diajukan untuk dibangun hingga saat ia ditarik dari peredaran. Pada perkembangannya, proses-proses standar pengembangan sistem, dituangkan dalam suatu metode yang dikenal dengan nama System Development

[6] Hartatik, M.Cs. Model Proses PL. elearning.amikom.ac.id/index.php/download/materi/555158-ST063-2/2012/03/20120312_Proses.ppt [6 Maret 2013]

Life Cycle (SDLC) yang merupakan metodologi umum dalam pengembangan sistem. Proses-proses standar pengembangan sistem sebagai berikut[7] :

2.8.1 Analisis

Tahap analisis adalah tahap di mana sistem yang sedang berjalan dipelajari dan sistem pengganti diusulkan. Dalam tahap ini sistem yang sedang berjalan, masalah, dan kesempatan didefinisikan, dan rekomendasi umum untuk bagaimana memberbaiki, meningkatkan, atau mengganti sistem yang sedang berjalan diusulkan. Tujuan utama dari fase analisis adalah untuk memahami dan mendokumentasikan kebutuhan bisnis (*business need*) dan persyaratan proses dari sistem baru.

2.8.2 Desain

Tahap desain adalah tahapan mengubah kebutuhan yang masih berupa konsep menjadi spesifikasi sistem yang riil. Tahap desain sistem dapat dibagi menjadi 2, yaitu :

2.8.2.1 Desain Logis

Desain logis adalah bagian dari fase desain dalam SDLC di mana semua fitur-fitur fungsional dari sistem dipilih dari tahapan analisis di deskripsikan terpisah dari platform komputer yang nanti digunakan.

2.8.2.2 Desain Fisik

Pada bagisn ini, spesifikasi logis diubah ke dalam detail teknologi di mana pemrograman dan pengembangan sistem bisa diselesaikan.

---

[7] Hanif Al Fattah, Analisis & Perancangan Sistem Informasi, Andi Offset, Yogyakarta, 2007, Hal 25

2.8.3 Implementasi

Output dari tahapan ini adalah *source code* yang *error free*, prosedur pelatihan, dan buku panduan. Pada tahap implementasi, terdapat beberapa hal yang perlu dilakukan, yaitu :

2.8.3.1 Testing

Testing yaitu menguji hasil kode program yang telah dihasilkan dari tahapan desain fisik. Tujuan pengujian ada 2. Dari sisi pengambangan sistem, harus dijamin kode program yang dibuat bebas dari kesalahan sintaks maupun logika. Dari sisi pengguna, program yang dihasilkan harus mampu menyelesaikan masalah yang ada pada klien dan sistem baru harus mudah dijalankan dan dipahami oleh pengguna akhir.

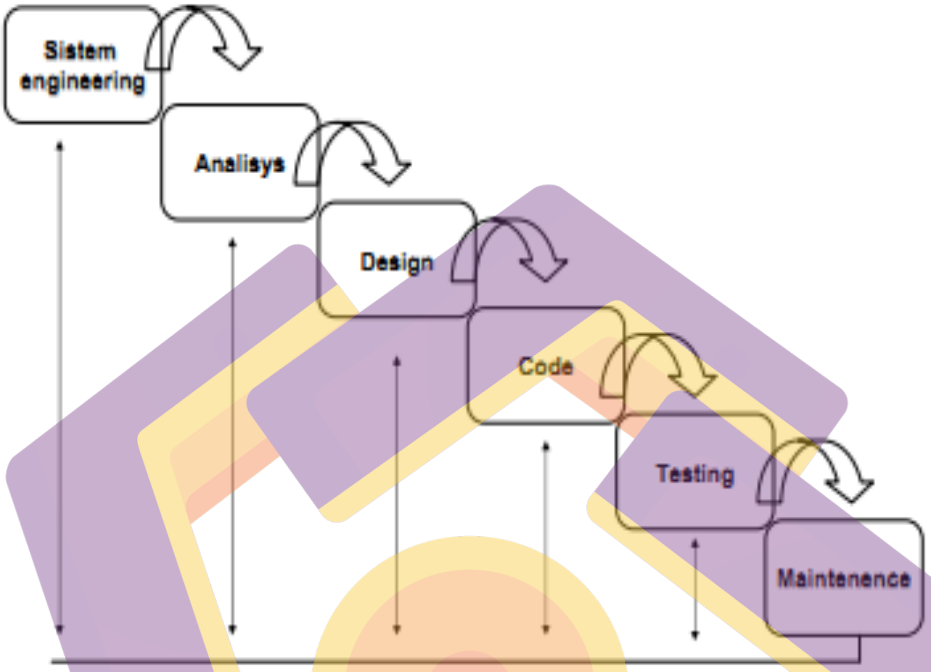
2.8.3.2 Instalasi

Setelah program lulus ujicoba, maka perangkat lunak dan perangkat keras akan diinstal pada organisasi atau perusahaan klien dan secara resmi mulai digunakan untuk menggantikan sistem lama.

2.8.4 Pemeliharaan

Pada tahap ini, sistem secara sistematis diperbaiki dan ditingkatkan. Hasil dari tahap ini adalah versi baru dari perangkat lunak yang telah dibuat. Perbaikan yang dilakukan tingkatannya bisa sangat variatif, mulai dari memperbaiki program yang crash hingga berfungsi kembali samapi pada penambahan modul-modul program yang baru sehingga jawaban atas perubahan kebutuhan pengguna.

2.9 Konsep Waterfall

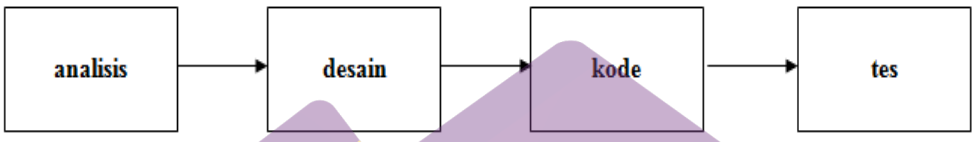


Gambar 2.5 Konsep Waterfall

Model waterfall :

1. Ini adalah model proses yang paling umum digunakan sehingga disebut dengan *classic life-cycle*.
2. Model ini menyarankan pendekatan sekuensial sistematis pembangunan software yang dimulai dari penspesifikasian persyaratan-persyaratan (*requirement*) oleh pengguna yang berlanjut melalui proses perencanaan, pemodelan, konstruksi dan penggelaran serta dukungan berlanjut setelah software telah lengkap sesuai dengan yang dipersyaratkan di awal pembangunannya.

3. Nama setiap tahap dalam model waterfall dapat berbeda pada setiap referensi yang berbeda, namun pada dasarnya esensinya tetap sama yakni urutan yang sekuensial dan sistematis dalam proses pembangunan sistem.



Gambar 2.6 Proses Pembangunan Sistem

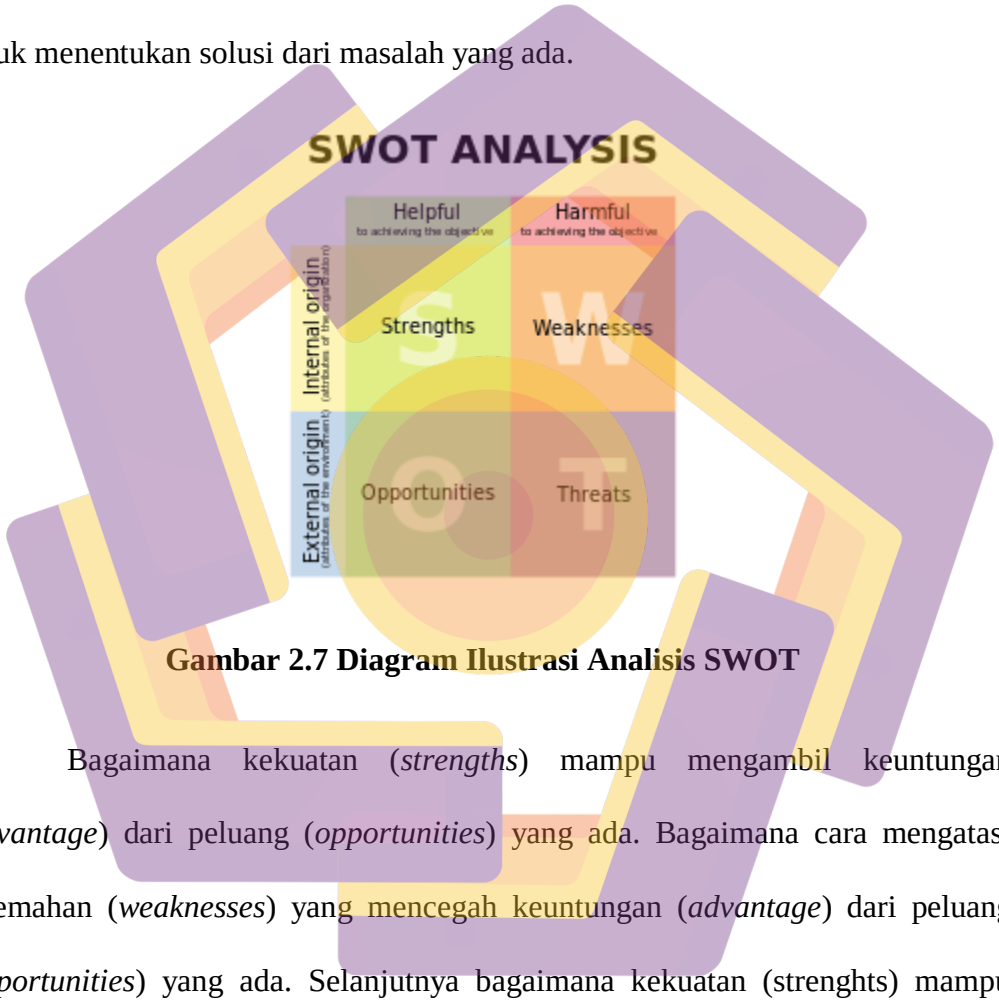
Keterangan[8] :

1. Communication adalah proses analisis kebutuhan perangkat lunak. Pada proses ini dibutuhkan pemahaman tentang domain informasi, tingkah laku, unjuk kerja, dan interface yang diperlukan.
2. Desain yaitu proses desain menerjemahkan syarat tau kebutuhan ke dalam sebuah representasi perangkat lunak. Menentukan dasar-dasar pembentukan dan pemilihan struktur data, struktur program, arsitektur program, pemilihan algoritma, interaksi dengan user.
3. Code adalah penerjemahan desain ke dalam coding program.
4. Pengujian pada logika internal PL untuk memastikan semua pernyataan sudah diuji. Selain itu pada eksternal fungsional , pengujian dilakukan untuk menemukan kesalahan-kesalahn dan memastikan bahwa input akan memberikan hasil actual yang sesuai dengan hasil yang dibutuhkan.

[8] Hartatik, M.Cs. Model Proses PL. elearning.amikom.ac.id/index.php/download/materi/555158-ST063-2/2012/03/20120312_Proses.ppt [6 Maret 2013]

2.10 Analisis SWOT

Analisis SWOT (*Strength*, *Weakness*, *Opportunities*, *Threatness*) yaitu dengan menganalisa kekuatan, kelemahan, peluang dan ancaman dari suatu aplikasi, sehingga akan didapatkan masalah yang akan menjadi objek bahasan untuk menentukan solusi dari masalah yang ada.



Gambar 2.7 Diagram Ilustrasi Analisis SWOT

Bagaimana kekuatan (*strengths*) mampu mengambil keuntungan (*advantage*) dari peluang (*opportunities*) yang ada. Bagaimana cara mengatasi kelemahan (*weaknesses*) yang mencegah keuntungan (*advantage*) dari peluang (*opportunities*) yang ada. Selanjutnya bagaimana kekuatan (strenghts) mampu menghadapi ancaman (threaths) yang ada. Dan terakhir bagaimana cara mengatasi kelemahan (weaknesses) yang mampu membuat ancaman (threats) menjadi nyata atau menciptakan sebuah ancaman baru[9].

[9] "Analisis SWOT." http:id.wikipedia.org/wiki/Analisis_SWOT [27 Januari 2012]

2.11 Pengujian Aplikasi *Mobile*

Aplikasi yang berjalan pada perangkat *mobile* sedang sangat populer, itu menggambarkan bahwa sedang ada revolusi di sektor IT. Dalam tiga tahun terakhir, lebih dari 300.000 telah diciptakan (sumber : *mobithinking.com*). Adalah sangat berbeda teknik pengujian aplikasi *mobile* dengan pengujian sistem informasi. Ada 2 hal yang perlu di uji saat ingin melakukan uji aplikasi *mobile*, yaitu (Valentino Lee, Heather Schneider, Robie Schell : 2004:180) :

2.11.1 *Code Development and Integration*

Project aplikasi *mobile* mungkin membutuhkan pengembangan pada aplikasi *client mobile*. Mereka mungkin juga membutuhkan pengembangan dan integrasi dengan adanya aplikasi *server-side*.

2.11.1.1 Coding

Aplikasi *mobile* membutuhkan kreasi *coding* dan / atau modifikasi paada beberapa tipe software. Diikuti beberapa item yang berhubungan dengan *testing coding*.

1. Aplikasi *mobile client*.
2. Presentasi deretan *code*.
3. Bisnis deretan *code*.
4. Deretan database dan ada tidaknya sistem *code*.
5. Tes *script*.
6. Release *script*.

2.11.1.2 Unit Testing

Terdiri dari *testing* setiap fungsi individu dari *code*. Banyak *tools* pengembangan moderen (sepertin *Microsoft Visual Studio .NET*) yang mengijinkan seseorang untuk menyusuri *code*, baris demi baris, pengujian variabel-variabel, dan item lain yang berhubungan dengan *code*.

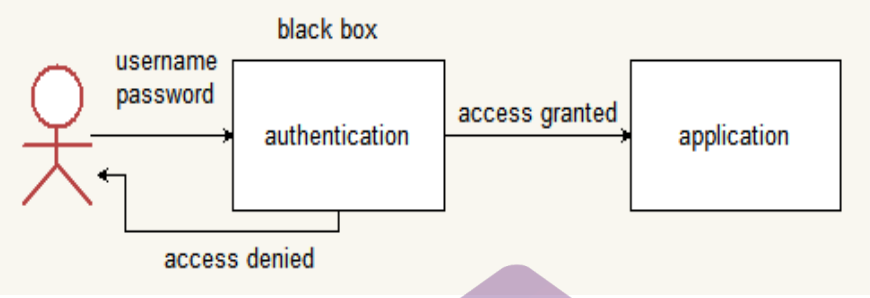
2.11.2 Integration and System Testing

Project pengambangan aplikasi *mobile* harus mengikuti cukup waktu untuk melengkapi integrasi dan sistem *testing* pada aplikasi *mobile* baru, infrastrukturnya, dan banyak perubahan untuk *server-side* aplikasi. Sebagai ciri penggabungan *testing* melibatkan *testing* komponen-komponen *substantial* pada aplikasi dan arsitektur sampai mereka telah menunjukan untuk bekerja bersama.

Itu berguna untuk catatan bahwa itu biasanya membawa waktu yang lama untuk pengetesan sumber dibawah kontrol perusahaan. *Testing* sumber tidak di bawah kontrol perusahaan, secara bersama-sama itu membutuhkan jadwal dan perlengkapan yang lebih lama.

2.11.2.1 Testing Proses

Ada beberapa pendekatan untuk penggabungan *testing* seperti *white-box* atau *black box*. Pada pengujian *black box*, itu lebih umum. *White box testing*, mengetes komponen internal dan lebih detail. Seseorang akan diijinkan untuk melihat output dan input dari komponen tanpa sesungguhnya mengetahui apa yang sedang terjasi di dalam. Pengujian *black box* menunjukan bahwa seseorang tidak usah peduli bagaimana itu bekerja selama apa yang diinputkan memberikan hasil yang benar.



Gambar 2.8 Contoh Testing Black Box

2.11.2.2 Testing Documentation

Tes biasanya didokumentasikan dalam beberapa kasus tunggal atau keseluruhan. Saat pengujian integrasi sedang berlangsung, itu bisa bermanfaat untuk mendokumentasikan, sehingga code dan hasil pengujian bisa dipantau dengan jelas.

2.11.2.3 Testing Considerations

Uji kasus harus dilakukan dari berbagai sudut pandang, sebagai berikut :

1. Fungsi dan kegunaan : menguji fungsional secara keseluruhan adalah pertimbangan utama, seperti menguji antarmuka.
2. Pengujian kinerja : dilakukan untuk mengetahui beban lalu lintas, jumlah pengguna, jumlah halaman yang diakses, server *uptime* dan *downtime*.
3. Pengujian keamanan : dapat dilakukan dengan berbagai tindakan termasuk deteksi virus, kuncian, perlindungan *firewall*.
4. Regresi pengujian : adalah hasil dari uji fungsi, uji kinerja dan uji keamanan.

BAB III

ANALISIS DAN PERANCANGAN SISTEM

3.1. Analisis Sistem

Analisa sistem didefinisikan sebagai bagaimana memahami dan menspesifikasikan dengan detail apa yang harus dilakukan oleh sistem. Menganalisa sistem bertujuan untuk mengurangi konsep kerja sistem yang akan dibuat dan manfaatnya. Pengembangan sistem informasi berbasis komputer menjadi tugas yang membutuhkan sumber daya untuk menyelesaikannya. Siklus hidup suatu sistem (*system life cycle*) suatu metode pengembangan sistem yang dapat memecahkan permasalahan, hambatan yang timbul, sehingga terjadi peningkatan kinerja seluruh elemen organisasi.

3.1.1 Analisis Kelemahan Sistem

Teknisi dalam analisis data menggunakan metode SWOT (*Strength, Weakness, Opportunities* dan *Threats*). SWOT adalah perangkat umum yang didesain dan digunakan sebagai langkah awal dalam proses pembuatan keputusan dan sebagai perancanan strategis dalam berbagai terapan. Analisis SWOT adalah metode perencanaan strategis yang digunakan untuk mengevaluasi kekuatan (*strengths*), kelemahan (*weaknesses*), peluang (*opportunities*), dan ancaman (*threats*) dalam suatu proyek atau suatu spekulasi bisnis.

Tabel 3.1 Analisis SWOT

Internal / **External**	**Strenght (S)**	**Weakness (W)**
	Strategi SO	Strategi WO
Opportunities (O)	1) Memberikan kemudahan navigasi dalam membuat agenda. 2) Mempunyai *alert* pengingat. 3) Menggunakan OS Android yang sedang popular.	1) Untuk mengurangi kelemahan pada sistem dibuatlah layanan berupa alert, yang memberi notifikasi pada member.
	Strategi ST	Strategi WT
Threats (T)	1) Enot dibuat, kedepannya untuk memenuhi kebutuhan informasi secara cepat, yang ditujukkan untuk member, sehingga penjadwalan kegiatan lebih mudah.	1) Meningkatkan kualitas dari aplikasi Agenda Pribadi “enot” sehingga dapat bersaing dengan aplikasi yang lain.

1. Analisis Kekuatan (*Strength*)

Merupakan analisis yang melihat kondisi kekuatan yang ada dalam pembuatan aplikasi Agenda Pribadi Enot. Dimana kekuatan dari analisis ini salah satunya adalah dukungan *sorftware* yang digunakan bersifat *open source*. Dan aplikasi pendukung seperti Inkscape, Photoshop, DiagramDesigner, Visual Paradigm, Microsoft Visio bersifat bebas didistribusikan. Sehingga dari sisi perangkat lunak, ada dukungan untuk analisis kekuatan.

2. Analisis Kelemahan (*Weakness*)

Pada tahap analisis ini, kelemahan yang ada dalam pembuatan aplikasi Agenda Pribadi Enot, sebagai berikut :

a. Aplikasi ini hanya berjalan pada *smartphone* android dengan menggunakan minimal Android OS 3.0.

b. Aplikasi dapat berjalan pada *tablet*, tetapi *icon* kurang mendukung.

3. Analisis Peluang (*Opportunities*)

Merupakan analisis yang melihat kondisi peluang yang ada dalam pembuatan aplikasi Agenda Pribadi Enot. Ponsel yang mendukung Android OS sekarang ini sedang populer, sehingga jumlah pengguna akan semakin meningkat dan akan semakin banyak pengguna aplikasi ini.

4. Analisis Ancaman (Threats)

Merupakan analisis yang melihat kondisi ancaman yang ada dalam pembuatan aplikasi Agenda Pribadi Enot. Analisis ini mencari ancaman dari luar yang bisa mengamcam aplikasi Agenda Pribadi Enot. Ancaman yang nyata adalah

karena Android bersifat *open source* yang *source code*-nya diberikan secara gratis bagi para pengembanguntuk bisa menciptakan aplikasi, maka persaingan dalam pembuatan aplikasi untuk Android sangatlah ketat termasuk pembuatan Aplikasi Agenda Pribadi.

3.1.2 Analisis Kebutuhan Sistem

Analisis kebutuhan sistem digunakan untuk mempermudah dalam menentukan keseluruhan kebutuhan-kebutuhan yang akan digunakan untuk menambah dan membantu jalannya proses pembuatan objek.

3.1.2.1 Kebutuhan Fungsional

Analisis kebutuhan fungsonal adalah bagisn paparan mengenai fitur-fitur yang akan dimasukan kedalam aplikasi yang akan dibuat.

1. Dapat menginputkan gambar yang dipilih sendiri oleh user(member enot).
2. Dapat menginputkan suara user(member enot).
3. Dapat menginputkan kegiatan dengan mengetik.
4. Dapat menampilkan list kegiatan yang sudah tersimpan sesuai jadwal.
5. Dapat menyediakan waktu pengingat saat jam dan tanggal yang di simpan user (member enot).
6. Dapat melihat isi kegiatan.
7. Dapat mengedit isi kegiatan.
8. Dapat menghapus isi kegiatan.
9. Dapat berbagi di beberapa media *sharing*.

3.1.2.2 Kebutuhan Non-Fungsional

Analisis kebutuhan non-fungsional adalah bagian yang akan mendukung jalannya proses pembuatan aplikasi Agenda Pribadi Enot untuk Android.

1. Kebutuhan Perangkat Keras

Spesifikasi perangkat keras yang akan digunakan dalam perancangan sistem sebagai berikut :

Tabel 3.2 Kebutuhan Perangkat Keras

Notebook	: Toshiba Satellite L630;
Processor	: Intel(R) Core(TM) i3 – 350M;
RAM	: 2 GB;
Hardisk	: 300 GB;
VGA	: Intel(R) HD Graphics.

2. Kebutuhan Perangkat Keras Untuk Penerapan

Spesifikasi untuk perangkat keras yang akan digunakan dalam penerapan sistem sebagai berikut :

Tabel 3.3 Kebutuhan Hardware Penerapan

Ponsel	: SONY Xperia sola
CPU	: 1 GHz STE U8500
RAM	: 512 MB
Memory	: 8 GB
Layar	: 3.7 inci
Sistem Operasi	: Ice Cream Sandwich

3. Kebutuhan Perangkat Lunak

Perangkat lunak yang digunakan dalam perancangan sistem ini adalah sebagai berikut :

Tabel 3.4 Kebutuhan Perangkat Lunak

Sistem Operasi	: Windows 7 Ultimate 32-bit;
Program Aplikasi	: Eclips Juno, Android SDK, SQLite Manager, Android ADT;
OS Android	: Versi 3.0;
Bahasa Pemrograman	: Java.

4. Kebutuhan Manusia (Brainware)

Kebutuhan brainware tidak bisa terlepas dari kebutuhan hardware dan kebutuhan software. Karena ketiga aspek ini harus bekerja sama agar aplikasi dapat bekerja dengan sempurna. Sebagai pengguna yang diusulkan adalah pengguna *smartphone* Android dengan user yang menjadi member.

3.1.3 Analisis Kelayakan Sistem

3.1.3.1 Kelayakan Teknologi

Pada media online vivanews.com Rabu, 22 Februari 2012 memberitakan "pada akhir 2012, jumlah perangkat mobile mulai menyaingi jumlah manusia", dan di Indonesia sendiri diberitakan dari telnologi kompasiana.com 21 Februari 2012 "menurut survey Nielsen (Lembaga Survey Konsumen), jumlah pengguna ponsel per Mei mencapai 125 juta orang dari 238 juta penduduk". Dari detiknet 30 April 2012 memberitakan "49% Ponsel Cerdas di Asean Dikuasai Android". Dari hal tersebut mengindikasikan bahwa penetrasi perangkat mobile dan android

kedalam kehidupan manusia terus meningkat, sangat layak dan memungkinkan sebagai platform dari sistem yang akan dibangun ini (Fajar Rizqi Karsa Perdana, 2012: 60).

3.1.3.2 Kelayakan Hukum

Dari sisi *software*, aplikasi yang digunakan untuk perancangan bersifat *open source*. Dan aplikasi pendukung seperti Inkscape, DiagramDesigner, Visual Paradigm dan lain-lain bersifat gratis dan bebas didistribusikan. Dari sisi data dan informasi, data pengguna disimpan secara pribadi pada *mobile device* masing-masing dan tidak akan disebarluaskan kepada pihak lain oleh sisitem ini. Sehingga jika ditinjau dari Kelayakan Hukum, perancangan sistem ini susuai dengan hukum yang berlaku.

3.1.3.3 Kelayakan Operasional

Agenda Pribadi "enot" yang akan dibangun nanatinya dapat dijalankan pada perangkat mobile berbasis Android minimal versi 3.0. Seperti diketahui Android merupakan *mobile flatform* dengan penetrasi yang besar, penggunanya sangat banyak sampai saat ini dan merupakan *flatform* perangkat *mobile* yang paling diminati di dunia.

3.2 Perancangan Sistem

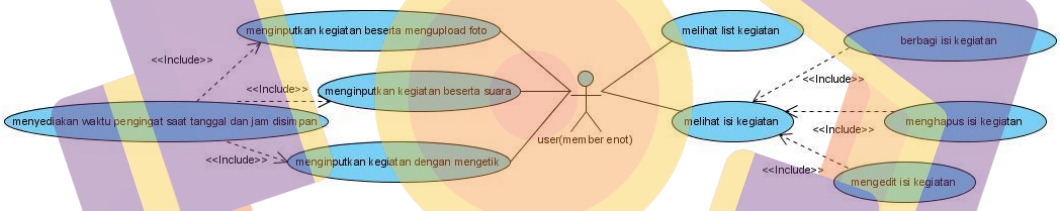
Perancangan sistem merupakan pemodelan secara umum mengenai alur kerja sistem yang akan dibuat. Supaya didapatkan gambaran yang jelas mengenai sistem tersebut.

3.2.1 Perancangan Proses

Perancangan proses dalam penelitian menggunakan *Unifield Modeling Language* (UML). UML adalah bahasa untuk melakukan visualisasi, spesifikasi, konstruksi, memodelkan bisnis, mendokumentasikan komponen-komponen sistem *software* dan sistem *non software*. UML menggunakan notasi yang bersifat standar untuk menjelaskan secara visual mengenai elemen-elemen pemodelan.

3.2.1.1 Use Case Diagram

Berikut ini adalah *usecase* diagram yang digunakan pada perancangan sistem "enot" :



Gambar 3.1 Use Case Diagram Enot

3.2.1.2 Use Case Description

Tabel 3.5 Use Case Description Menginputkan Kegiatan Beserta Mengupload Foto

Nama Use Case	Menginputkan kegiatan beserta mengupload foto
Actor	User (member enot)
Triggering event	Member enot ingin menulis kegiatan dari menu foto
Deskripsi singkat	Member enot ingin memilih menu foto
Relasi Use Case	Menyediakan waktu pengingat saat tanggal dan jam disimpan
Stack holder	-

Preconditions	Aplikasi menampilkan dashboard	
Postconditions	Aplikasi menampilkan dashboard	
Flow of events	Actor	Sistem
	1. Memilih menu foto 2. Menginputkan judul, kegiatan, gambar, dan pengaturan waktu, menyimpan	1. Menampilkan menu foto 2. Menyimpannya ke database `enot_db` di tb_foto
Exception	User (member enot) batal menyimpan	

Tabel 3.6 Use Case Description Menginputkan Kegiatan Dengan Mengetik

Nama Use Case	Menginputkan kegiatan dengan mengetik	
Actor	User (member enot)	
Triggering event	Member enot ingin menulis kegiatan dengan menu ketik	
Deskripsi singkat	Member enot memilih menu ketik	
Relasi use case	Menyediakan waktu pengingat saat tanggal dan jam disimpan	
Stack holder	-	
Preconditions	Apikasi menampilkan dashboard	
Postconditions	Aplikasi menampilkan dashboard	
Flow of event	Actor	Sistem
	1. Memilih menu ketik 2. Menginputkan judul, kegiatan, dan pengaturan tanggal, simpan	1. Menampilkan menu ketik 2. Menyimpannya ke database `enot_db` di tb_ketik
Exception	Member batal menyimpan menu ketik	

Tabel 3.7 Use Case Description Menginputkan Kegiatan Beserta Suara

Nama use case	Menginputkan kegiatan beserta suara	
Actor	User (member enot)	
Triggering event	Member enot ingin meginputkan suara dari menu suara	
Deskripsi singkat	Member enot memilih menu suara	
Relasi use case	Menyediakan waktu pengingat saat tanggal dan jam disimpan	
Stack holder	-	
Preconditions	Aplikasi menampilkan dashboard	
Postconditions	Aplikasi menampilkan dashboard	
Flow of event	Actor	Sistem
	1. Memilih menu suara 2. Menginputkan judul, merekam, melakukan pengaruran dan menyimpan	1. Menampilkan menu suara 2. Menyimpannya ke database `enot_db` di `tb_suara`
Exception	Member enot batal menyimpan menu suara	

Tabel 3.8 Use Case Description Melihat List Kegiatan

Nama Use Case	Melihat list kegiatan
Actor	User (member enot)
Triggering event	Member ingin melihat list kegiatan yang sudah diinputkan dari menu foto, menu suara, menu ketik
Deskripsi singkat	Member enot melihat list kegiatan yang sudah dibuat
Relasi use case	-

Stake holder	-	
Preconditions	Aplikasi menampilkan menu kalender	
Postconditions	Aplikasi menampilkan *single* kegiatan	
Flow of events	Actor	Sistem
	1. Memilih menu kalender 2. Memilih tanggal	1. Menampilkan menu kalender 2. Menampilkan list kegiatan
Exception	User (member enot) hanya melihat menu kalender	

Tabel 3.9 Use Case Description Melihat Isi Kegiatan

Nama Use Case	Melihat isi kegiatan	
Actor	User (member enot)	
Triggering event	Member ingin melihat hasil inputan dari menu foto, menu ketik, menu suara	
Deskripsi singkat	Member melihat isi kegiatan	
Relasi use case	Berbagi isi kegiatan, menghapus isi kegiatan, mengedit isi kegiatan	
Stake holder	-	
Preconditions	Aplikasi menampilkan list kegiatan	
Postcondition	Aplikasi menampilkan *single* kegiatan	
Flow of events	Actor	Sistem
	1. Memilih menu kalender 2. Memilih tanggal	1. Menampilkan menu kalender 2. Menampilkan list kegiatan

	3. Memilih salah satu list kegiatan	3. Menampilkan single kegiatan
Exception	Member enot hanya melihat list kegiatan	

Tabel 3.10 Use case Description Mengedit Isi Kegiatan

Nama Use case	Mengedit isi kegiatan	
Actor	User (member enot)	
Triggering event	Member enot ingin mengedit kegiatan yang pernah diinputkan	
Deskripsi singkat	Member ingin mengedit isi kegiatan	
Relasi use case	Melihat isi kegiatan	
Stake holder	-	
Preconditions	Aplikasi menampilkan list kegiatan	
Postcondition	Aplikasi menampilkan *single* post	
Flow of event	Actor	Sistem
	1. Memilih menu kalender	1. Menampilkan menu kalender
	2. Memilih tanggal	2. Menampilkan list kegiatan
	3. Memilih list kegiatan yang ingin diedit	3. Menampilkan *single* kegiatan
Exception	Member enot hanya melihat list kegiatan	

Tabel 3.11 Use Case Description Berbagi Isi Kegiatan

Nama Use case	Berbagi isi kegiatan
Actor	User (member enot)

<table>
<tr><td>Triggering event</td><td colspan="2">User (member enot) ingin berbagi beberapa kegiatan</td></tr>
<tr><td>Deskripsi singkat</td><td colspan="2">User (member enot) melihat isi kegiatan dan dapat langsung berbagi</td></tr>
<tr><td>Relasi use case</td><td colspan="2">Melihat isi kegiatan</td></tr>
<tr><td>Stake holder</td><td colspan="2">-</td></tr>
<tr><td>Preconditions</td><td colspan="2">Aplikasi menampilkan list kegiatan</td></tr>
<tr><td>Postcondition</td><td colspan="2">Aplikasi menampilkan beberapa media sharing</td></tr>
<tr><td rowspan="2">Flow of events</td><td>Actor</td><td>Sistem</td></tr>
<tr><td>1. Memilih menu kalender
2. Memilih tanggal
3. Melihat list kegiatan yang ingin dibagikan</td><td>1. Menampilkan menu kalender
2. Menampilkan list kegiatan
3. Menampilkan single kegiatan yang ingin dibagikan</td></tr>
<tr><td>Exception</td><td colspan="2">Member enot hanya melihat list kegiatan</td></tr>
</table>

Tabel 3.12 Use Case Description Menghapus Isi Kegiatan

Nama Use Case	Menghapus isi kegiatan
Actor	User (member enot)
Triggering event	Member enot ingin menghapus beberapa list kegiatan
Deskripsi singkat	Member enot manghapus kegiatan
Relasi use case	Melihat isi kegiatan
Stake holder	-
Preconditions	Aplikasi menampilkan list kegiatan
Postcondition	Aplikasi menampilkan *single* kegiatan

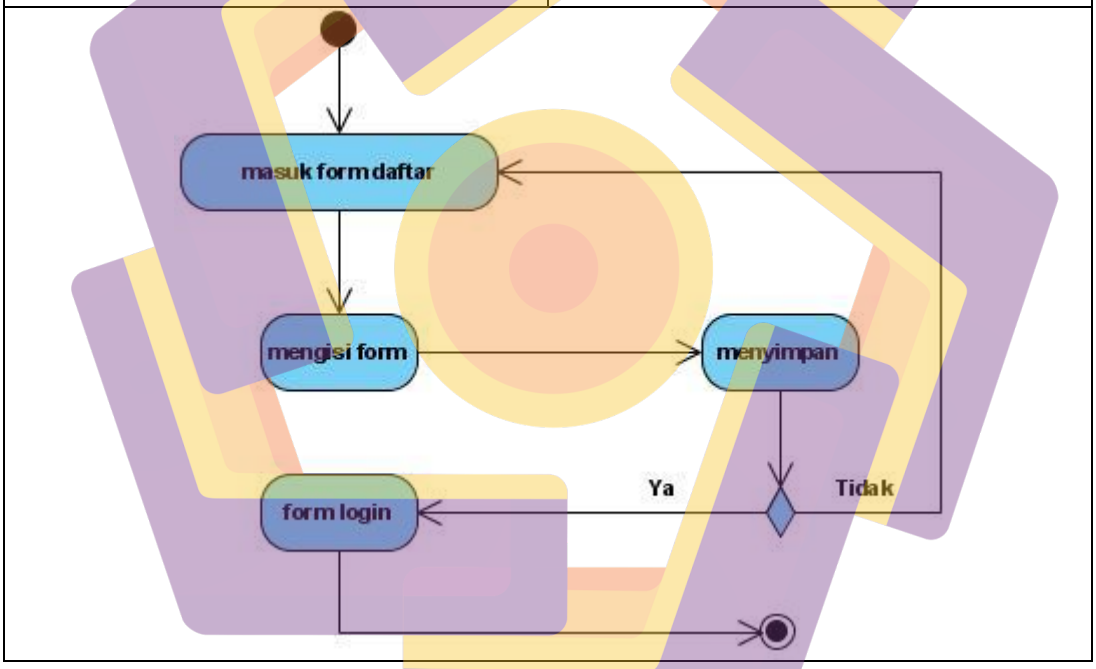
Flow of event	Actor	Sistem
	1. Memilih menu kalender 2. Memilih tanggal 3. Memilih list kegiatan yang ingin dihapus	1. Menampilkan menu kalender 2. Menampilkan list kegiatan 3. Menampilkan *single* kegiatan yang ingin dihapus
Exception	User (member enot) hanya melihat list kegiatan	

Tabel 3.13 Use Case Description Menyediakan Waktu Pengingat

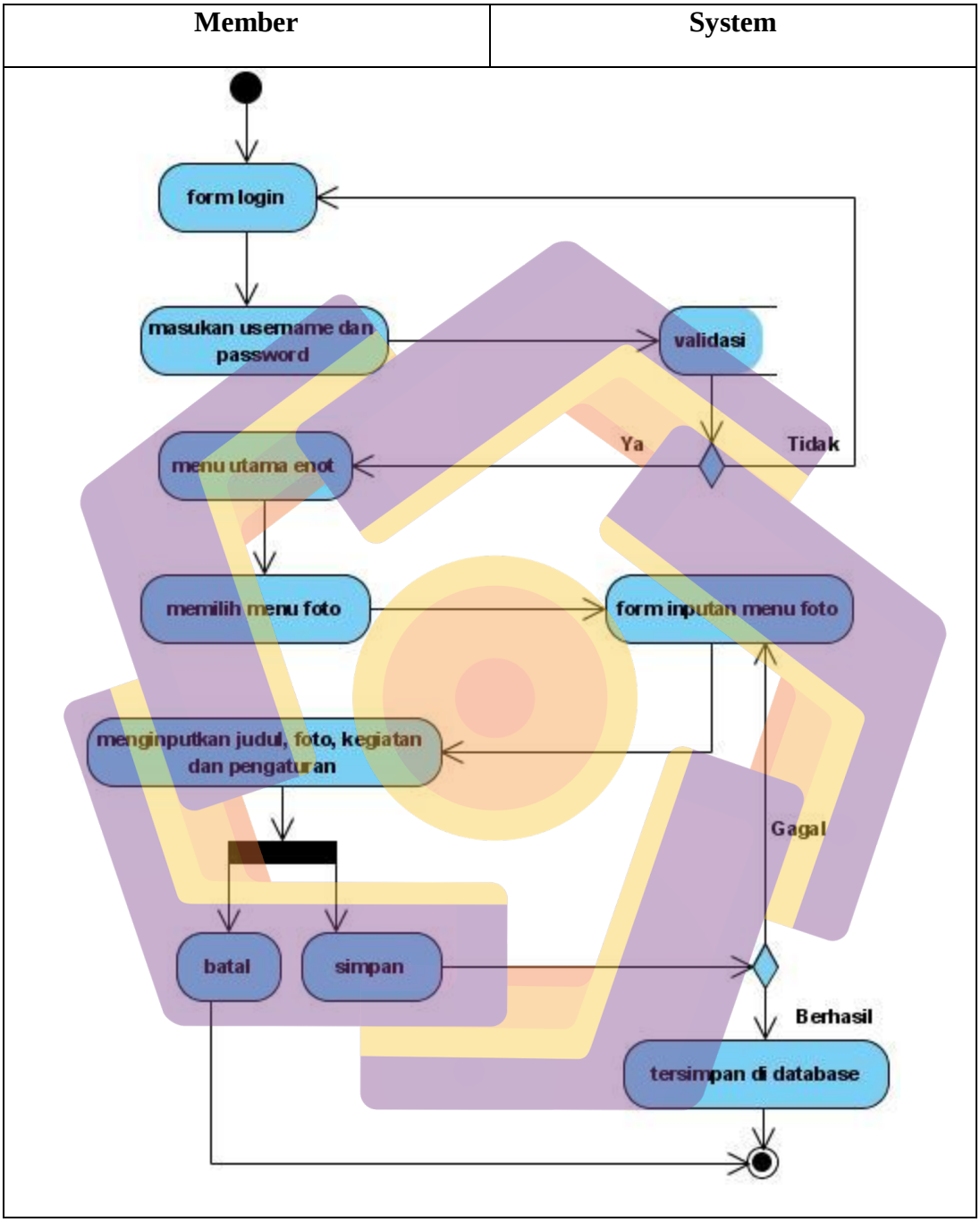
Nama Use Case	Menyediakan waktu pengingat	
Actor	User (member enot)	
Triggering event	Member enot ingin ada *alert* kegiatan	
Deskripsi singkat	Member enot ingin ada *alert* untuk kegiatan yang disimpan	
Relasi use case	Menginputkan kegiatan beserta mengupload foto, menginputkan kegiatan dengan mengetik, menginputkan kegiatan beserta suara	
Stake holder	-	
Preconditions	Aplikasi menampilkan inputan kegiatan	
Postcondition	Aplikasi menampilkan *single* kegiatan	
Flow of event	Actor	Sistem
	1. Memilih menu inputan 2. Melakukan pengaturan, menyimapan	1. Menampilkan menu inputan yang dipilih 2. Menyimpan ke database `enot_db`

	3. Mendapat *alert* kegiatan pada jam yang sudah diinputkan	3. Memberikan *alert*
Exception	User (member enot) tidak melakukan pengaturan kegiatan	

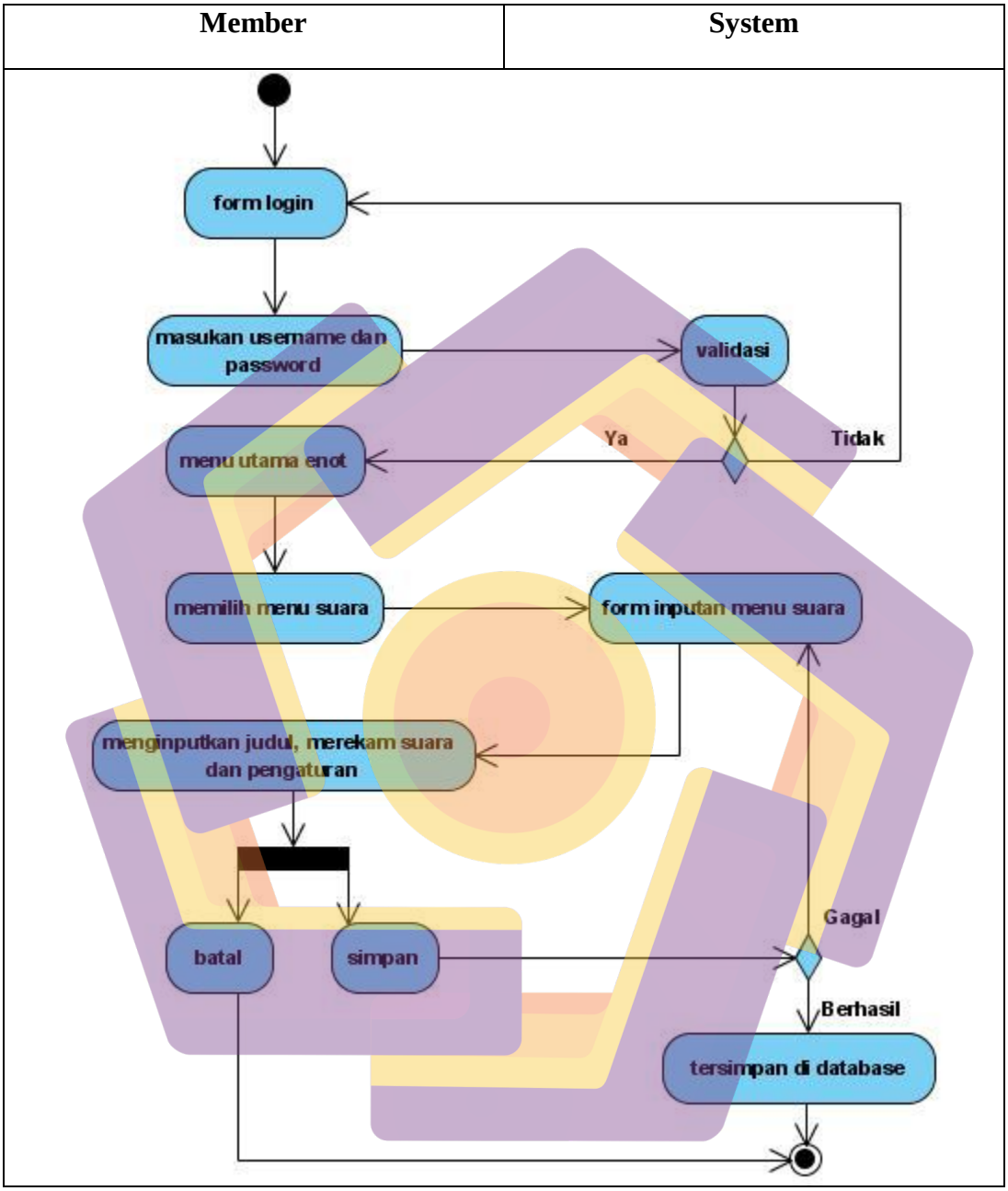
3.2.1.3 Activity Diagram



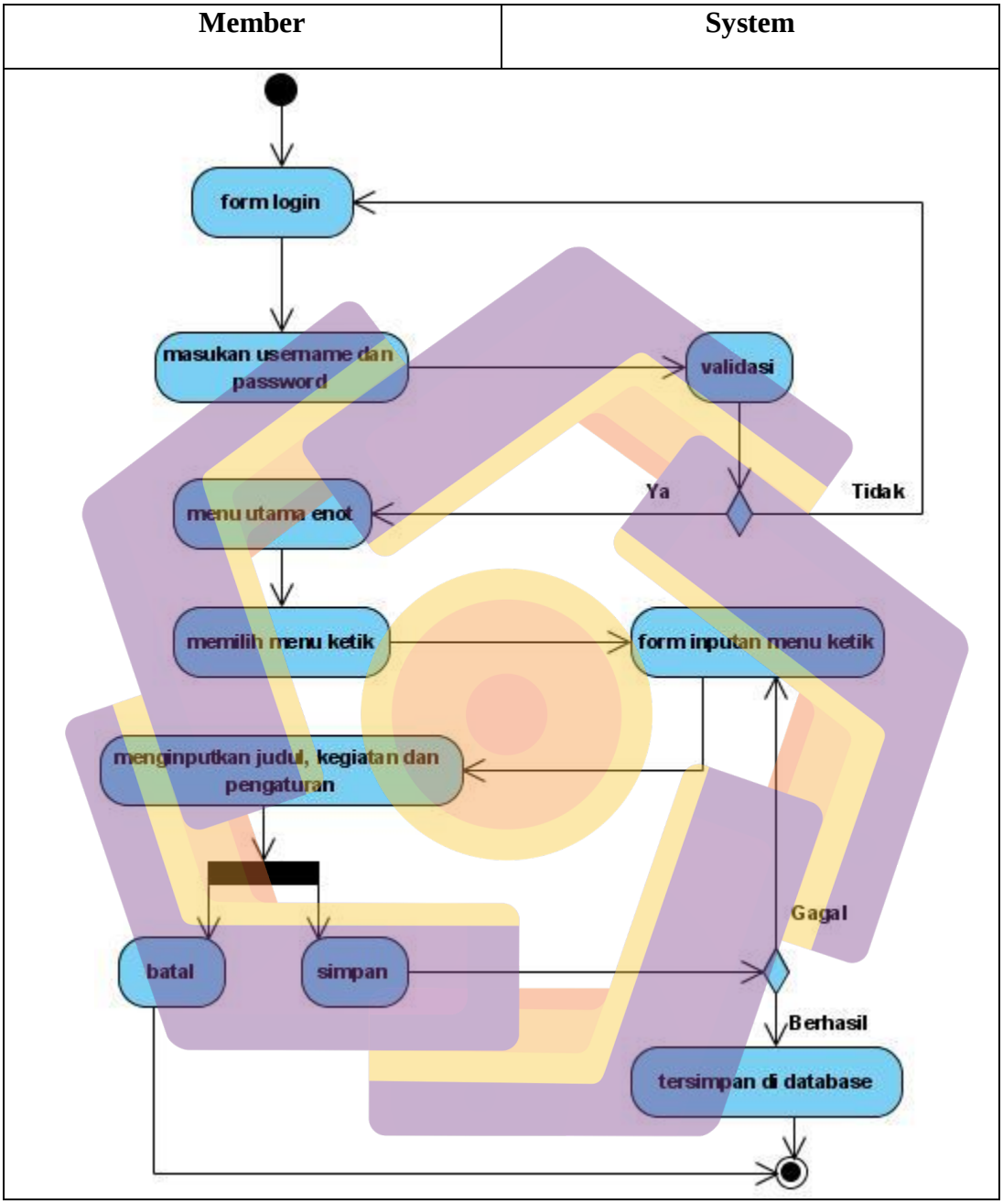
Gambar 3.2 Activity Diagram Registrasi User



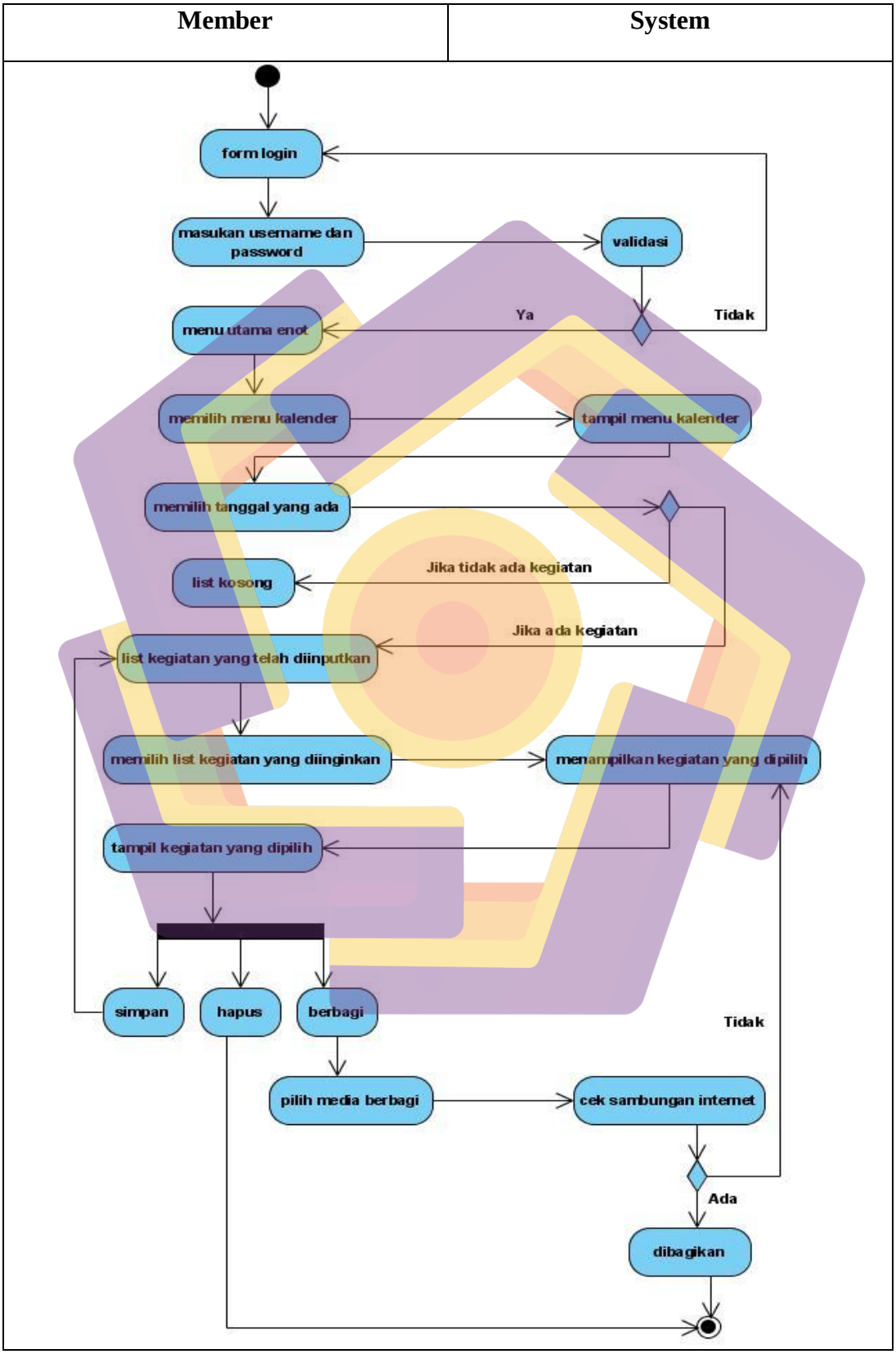
Gambar 3.3 Activity Diagram Tampil Menu Foto



Gambar 3.4 Activity Diargam Tampil Menu Suara

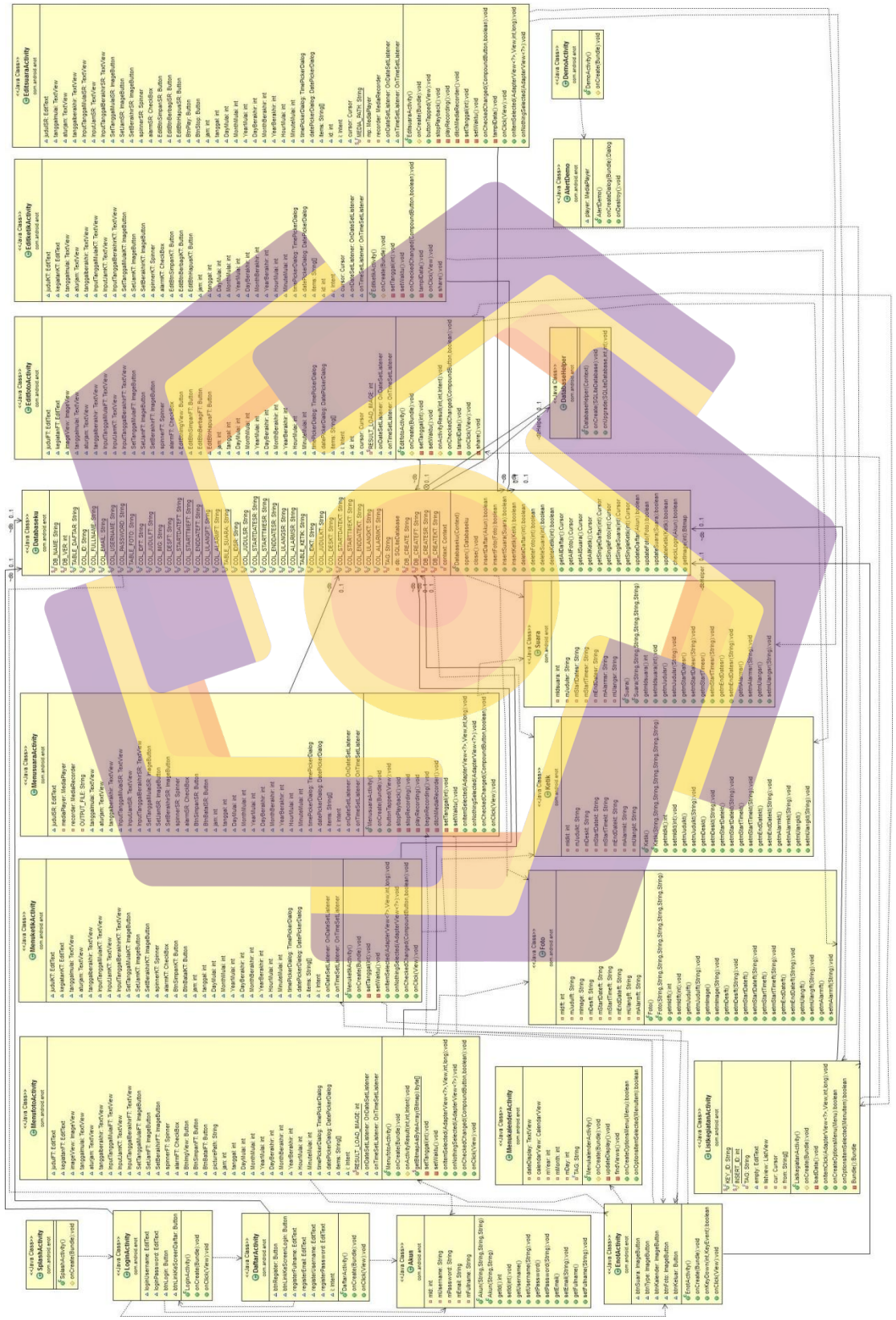


Gambar 3.5 Activity Diagram Tampil Menu Ketik



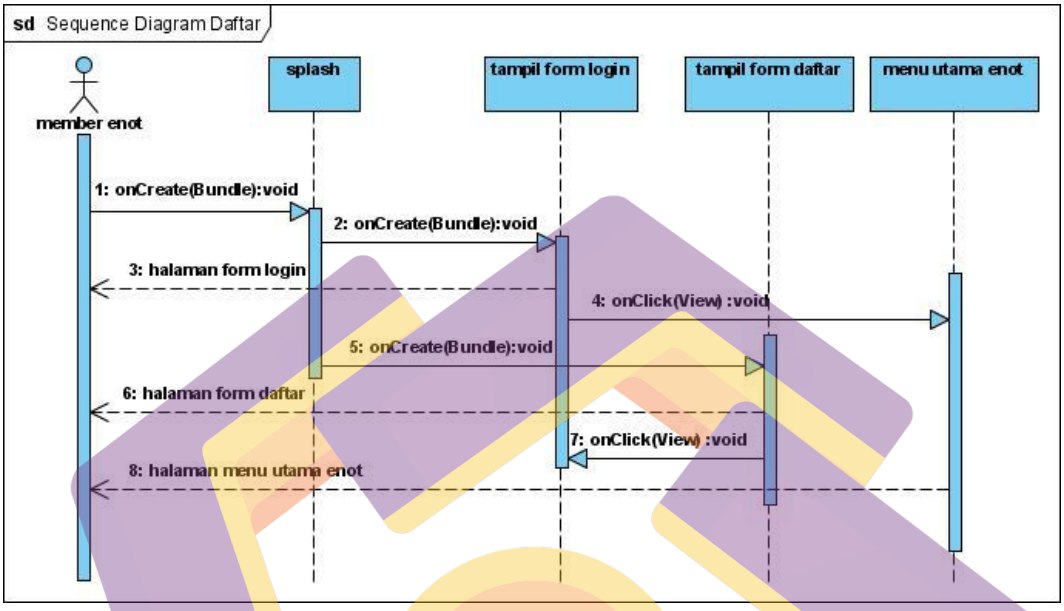
Gambar 3.6 Activity Diagram Tampil Menu Kalender

3.2.1.4 Class Diagram

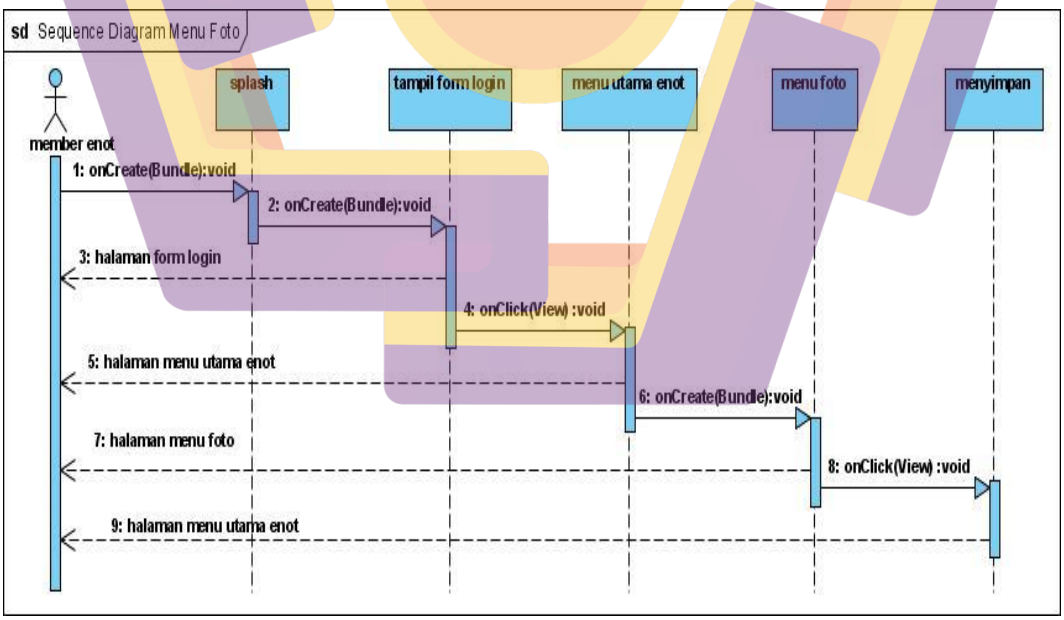


Gambar 3.7 Class Diagram Enot

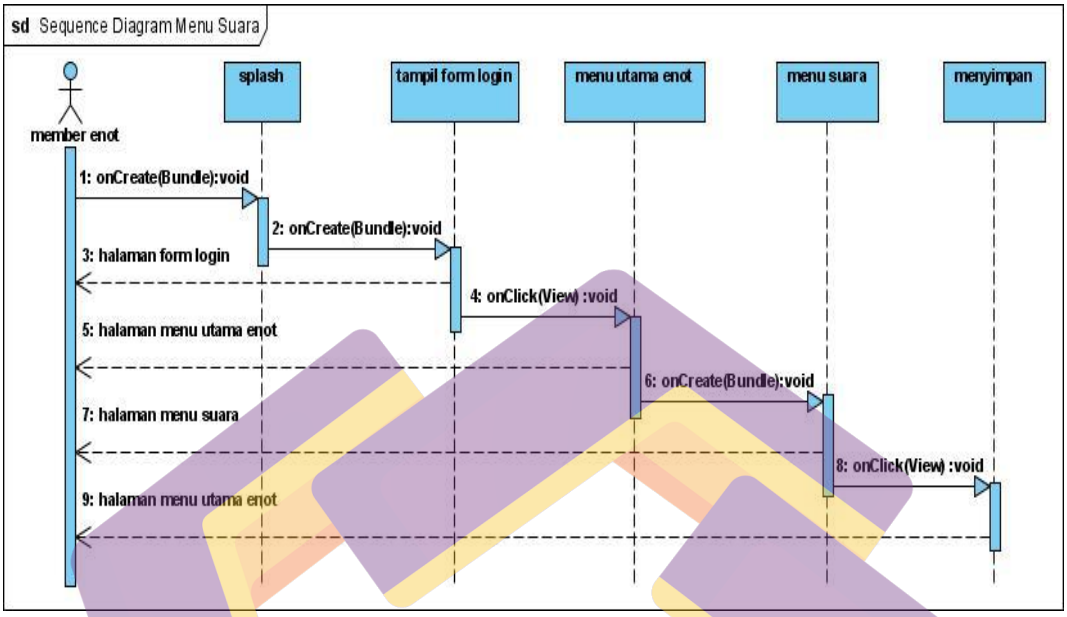
3.2.1.5 Sequence Diagram



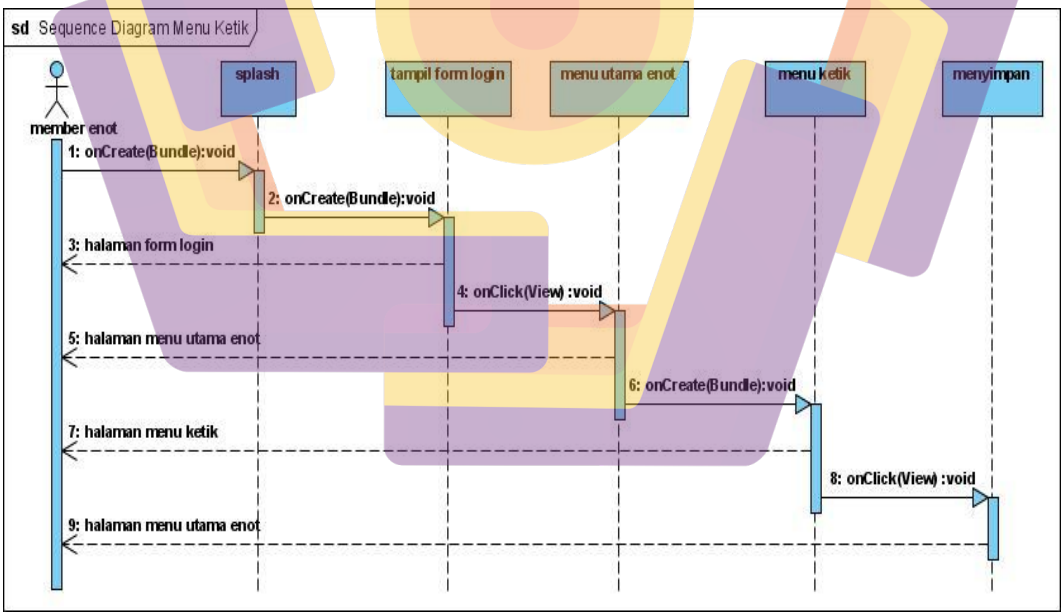
Gambar 3.8 Sequence Diagram Daftar



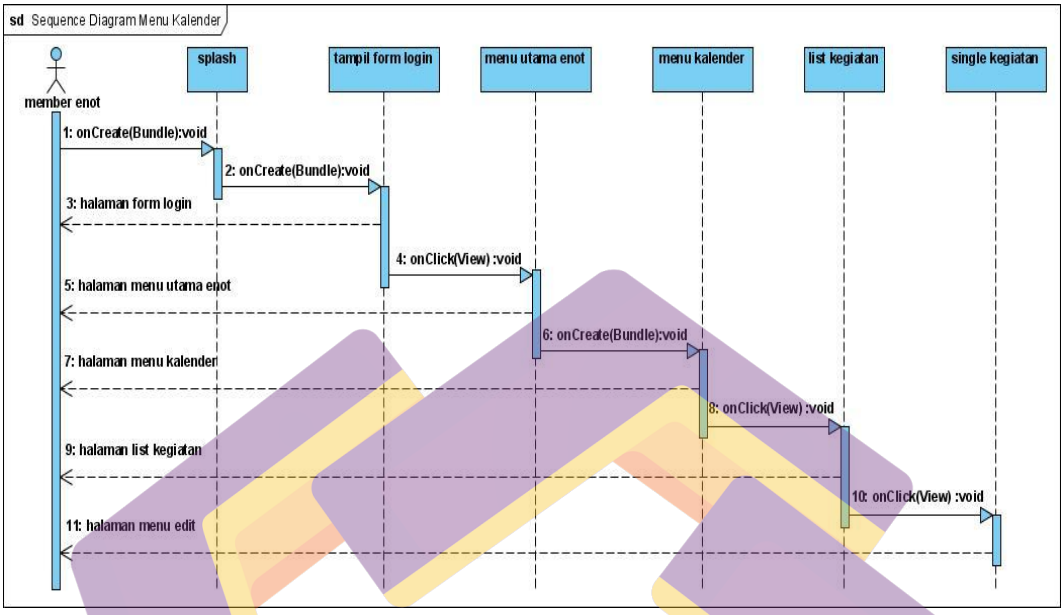
Gambar 3.9 Sequence Diagram Menu Foto



Gambar 3.10 Sequence Diagram Menu Suara



Gambar 3.11 Sequence Diagram Menu Ketik

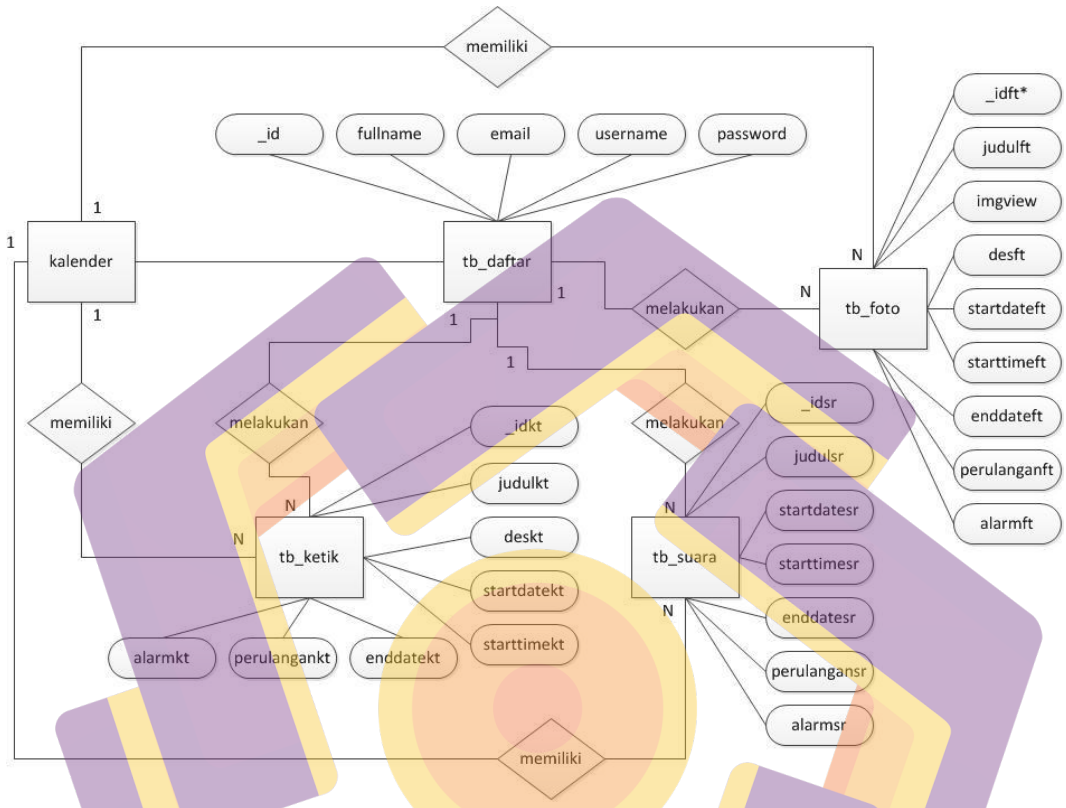


Gambar 3.12 Sequence Diagram Menu Kalender

3.2.2 Perancangan Database

Perancangan basis data merupakan tahapan desain data yang meliputi perancangan tabel yang berfungsi untuk melakukan penyimpanan data. Agar sebuah database efektif dan efesien, maka tabel-tabel yang ada harus memiliki struktur yang baik dan benar.

3.2.2.1 ERD



Gambar 3.13 Rancangan Model ERD

3.2.2.2 Struktur Tabel

Berikut ini struktur basis data dari agenda pribadi "enot" :

1. Tabel tb_daftar

Tabel `tb_daftar` terdiri dari 5 kolom digunakan untuk menampung data member. Kolom username dan password akan digunakan member untuk login.

Tabel 3.14 Struktur Tabel tb_daftar

Field	Type	Keterangan
_id	INTEGER	Autoincrement
fullname	TEXT	Not Null

email	TEXT	Not Null
username	TEXT	Not Null
password	TEXT	Not Null

2. Tabel tb_foto

Tabel `tb_foto` terdiri dari 9 kolom digunakan untuk menampung inputan dari Menu Foto. Pada kolom `judulft`, akan digunakan untuk memanggil judul di `ListActivity`.

Tabel 3.15 Struktur Tabel tb_foto

Field	Type	Keterangan
_idft*	INTEGER	Autoincrement
judulft	TEXT	Not Null
imgview	TEXT	Not Null
desft	TEXT	Not Null
startdateft	DATETIME	Not Null
starttimeft	DATETIME	-
enddateft	DATETIME	-
perulanganft	TEXT	-
alarmft	TEXT	-

3. Tabel tb_suara

Tabel `tb_suara` terdiri dari 7 kolom digunakan untuk menampung inputan dari Menu Suara. Pada kolom `judulsr`, akan digunakan untuk memanggil judul di `ListActivity`.

Tabel 3.16 Struktur Tabel tb_suara

Field	Type	Keterangan
_idsr	INTEGER	Autoincrement
judulsr	TEXT	Not Null
startdatesr	DATETIME	Not Null
starttimesr	DATETIME	-
enddatesr	DATETIME	-
perulangansr	TEXT	-
alarmsr	TEXT	-

4. Tabel tb_ketik

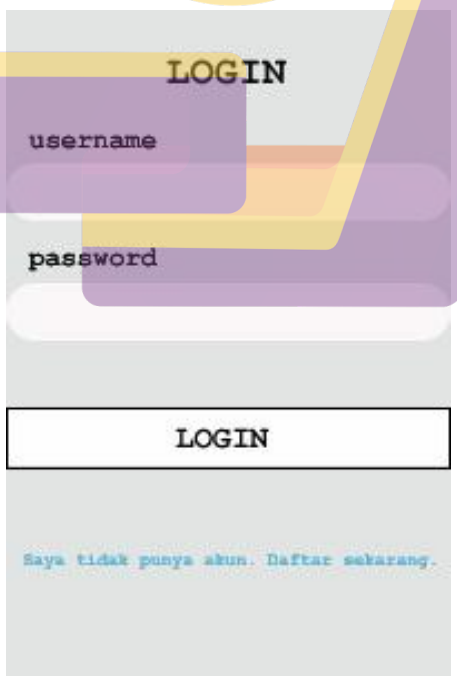
Tabel `tb_ketik` terdiri dari 8 kolom digunakan untuk menampung inputan dari Menu Ketik. Pada kolom `judulkt`, akan digunakan untuk memanggil judul di `ListActivity`.

Tabel 3.17 Struktur Table tb_ketik

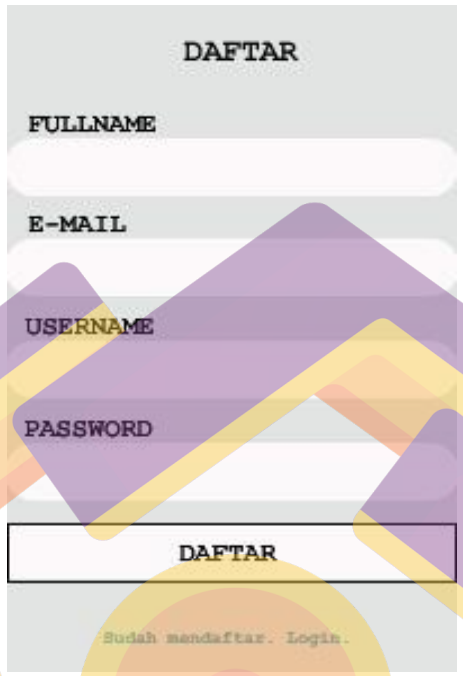
Field	Type	Keterangan
_idkt	INTEGER	Autoincrement
judulkt	TEXT	Not Null
deskt	TEXT	Not Null
startdatekt	DATETIME	Not Null
starttimekt	DATETIME	-
enddatekt	DATETIME	-
peralangankt	TEXT	-
alarmkt	TEXT	-

3.2.3 Perancangan Interface / Antarmuka

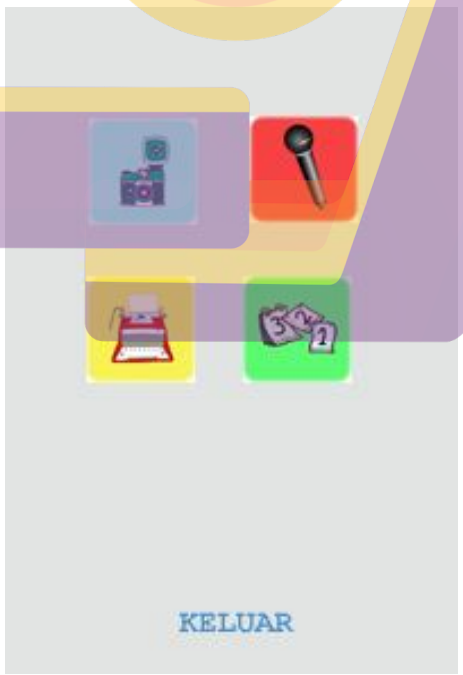
Gambar 3.14 Splash Screen Enot (Resolution : 200 x 300 pixel)



Gambar 3.15 Login Enot (Resolution : 200 x 300 pixel)



Gambar 3.16 Daftar Enot (Resolution : 200 x 300 pixel)

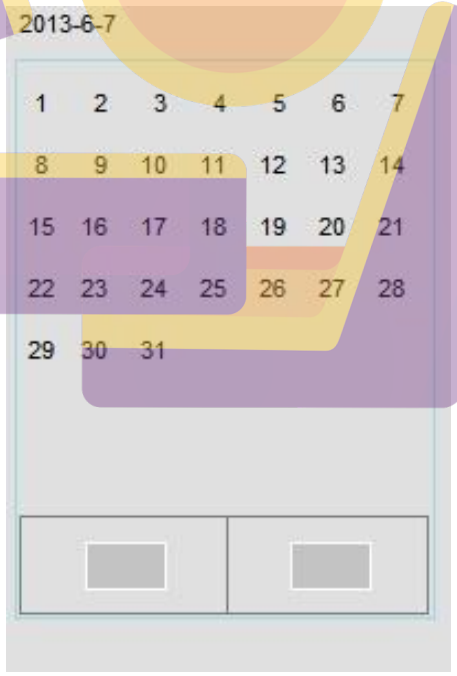


Gambar 3.17 Menu Utama Enot (Resolution : 200 x 300 pixel)

Gambar 3.18 Menu Foto Enot (Resolution : 200 x 300 pixel)

Gambar 3.19 Menu Suara Enot (Resolution : 200 x 300 pixel)

Gambar 3.20 Menu Ketik Enot (Resolution : 200 x 300 pixel)



Gambar 3.21 Menu Kalender Enot (Resolution : 200 x 300 pixel)

Gambar 3.22 List Kegiatan Enot (Resolution : 200 x 300 pixel)

BAB IV

IMPLEMENTASI DAN PEMBAHASAN

4.1. Implementasi

Implementasi merupakan tahap di mana sistem informasi telah digunakan oleh pengguna (Al Fattah, Hanif: 2007:167). Apakah aplikasi yang telah dirancang dapat berjalan dan berfungsi dengan benar sesuai dengan keadaan sebenarnya. Sehingga aplikasi dapat menghasilkan keluaran yang sesuai dengan tujuan yang diinginkan.

Beberapa tahap implementasi yang dilakukan, untuk pembuatan sistem aplikasi Agenda Pribadi Enot :

1. Membuat Project Aplikasi Android dengan Eclipse Juno dengan nama ENOT dan Target SDK min Android 3.0.
2. Mengecek SDK, apakah sudah terinstal atau belum di *Windows* > *Preferences* > Android.
3. Implementasi interface dengan memasukan semua *icon* yang dibutuhkan di *drawable*. Juga memasukan suaraalarm.mp3 di *raw* yang nantinya akan dijadikan *alert*.
4. Membuat database dengan nama `enot_db` yang berisi 4 tabel pada *class* `Databaseku.java` yaitu `tb_daftar`, `tb_foto`, `tb_suara`, dan `tb_ketik`.
5. Mengecek apakah tabel yang dibuat berhasil di *create* atau tidak, menggunakan *SQLite Browser*.

6. Mencoba *insert* di *form* daftar, apakah data yang di diisikan bisa tersimpan atau tidak.
7. Mencoba mengakses *form* login menggunakan data tabel `tb_daftar`.
8. Memberikan *code* pada *Activity* menu foto, menu suara, menu ketik dan menu kalender.
9. Membuat *class* model `Foto.java`, `Suara.java`, `Ketik.java`, `ListKegiatan.java` yang nantinya berfungsi menghubungkan antara database dengan menu foto, menu suara, menu ketik dan list kegiatan.
10. Menampilkan bagaimana semua kegiatan yang tersimpan dapat muncul di menu kalender.
11. Menampilkan bagaimana cara memanggil *single* kegiatan yang akan ditampilkan di l*ist* berupa judul yang disimpan.
12. Membuat *layout* menu edit untuk foto, menu edit suara dan menu edit ketik.
13. Membuat bagaimana saat salah satu *list* judul dipilih, akan menuju ke menu edit.
14. Memberikan 3 *button* pada *page* edit yaitu simpan, berbagi dan hapus.
15. Pengujian program secara keseluruhan, dilakukan pada berbagai *smartphone* (Tab 7" GT-P3100, Sony Xperia Sola, Samsung Galaxy Note, LG E435). Pengujian dilakukan juga untuk memastikan tidak ada kesalahan di *logcat*.
16. Pemeliharaan program dilakukan untuk menghindari kebutuhan sistem yang masih kurang.

4.1.1 Membuat Project Enot

Pertama kali, buat project dengan klik **File** > **New** > **Android Application Project**. Akan muncul jendela seperti gambar. Kemudian pada kota dialog, isikan sebagai berikut :

Application Name : ENOT

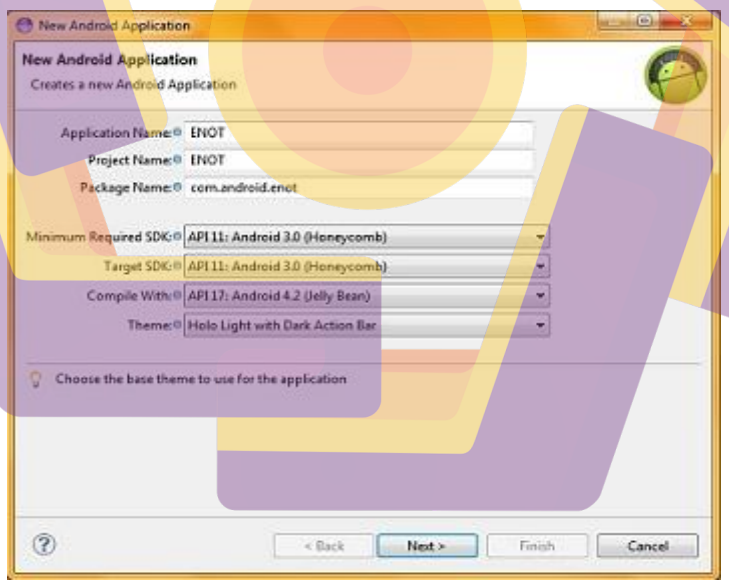
Project Name : ENOT

Build Target : Android 3.0

Package Name : com.android.enot

Min SDK : 11

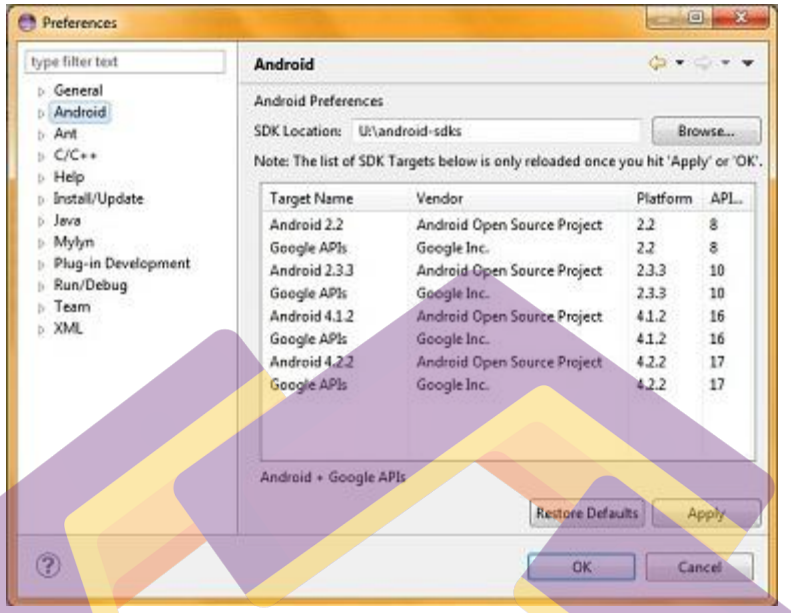
Target SDK : API 11: Android 3.0 (Honeycomb)



Gambar 4.1 Membuat Project ENOT

4.1.2 Cek SDK

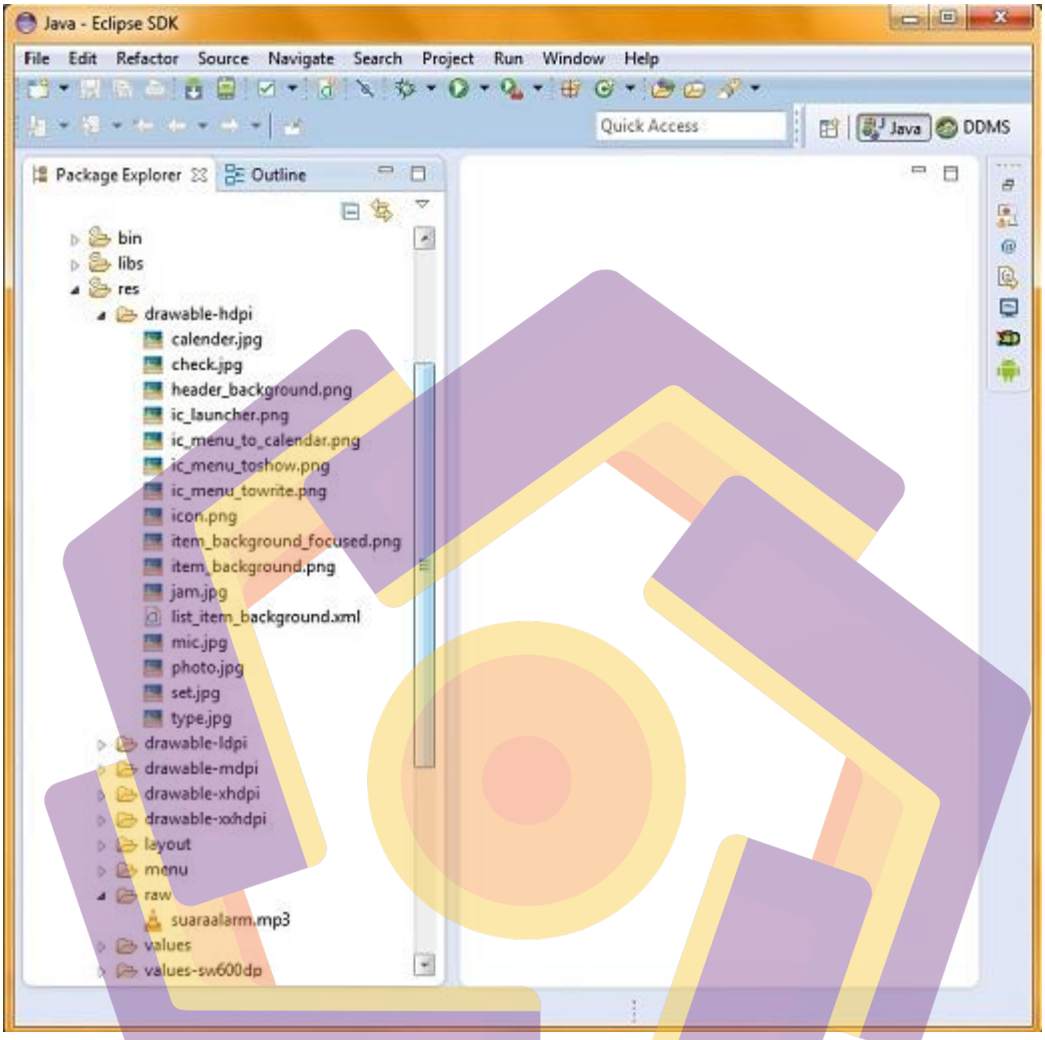
Mengecek apakah SDK sudak terinstal atau belum **Windows** > **Preferences** > **Android**. Jika kosong, maka **Browse** lokasi penyimpanan SDK. Klik **Apply** > **OK**.



Gambar 4.2 Cek SDK

4.1.3 Implementasi Interface

Aplikasi Agenda Probadi Enot berbasis Android ini terdiri dari beberapa halaman *interface*, antara lain : *splash screen*, *dashboard*, menu foto, menu suara, menu ketik dan menu kalender. Cara paling mudah memasukan icon adalah memasukan langsung lewat **folder res**. Sedangkan **folder raw** digunakan untuk menyimpan suara.



Gambar 4.3 Implementasi Icon

4.1.3.1 Splash Screen

Splash screen merupakan halaman *interface* yang pertama muncul ketika aplikasi dijalankan. Tampilan ini kemudian terhubung ke halaman *dashboard*. Berikut merupakan tampilan *interface* dari *splash screen* ketika aplikasi Enot dijalankan.

Gambar 4.4 Tampilan Splash Screen Enot

4.1.3.2 Dashboard

Dashboard merupakan menu utama yang di dalamnya terdapat menu-menu pilihan seperti menu foto, menu suara, menu ketik, menu kalender, dan *button* Keluar aplikasi. Berikut tampilan *dashboard* Enot.

Gambar 4.5 Tampilan Dashboard Enot

4.1.3.3 Menu Foto

Menu foto merupakan menu yang menginputkan judul foto, gambar foto (menggunakan *button* pilih gambar), kegiatan dan beberapa pengaturan. Berikut merupakan tampilan dari menu foto Enot.

Gambar 4.6 Tampilan Menu Foto Enot

4.1.3.4 Menu Suara

Menu suara merupakan menu yang dapat merekam suara member menggunakan *button* merekam dan berhenti. Menu suara juga terdiri dari judul, button putar dan selesai juga beberapa pengaturan. Berikut merupakan tampilan dari menu suara Enot.

Gambar 4.7 Tampilan Menu Suara Enot

4.1.3.5 Menu Ketik

Menu ketik merupakan menu yang menginputkan judul, kegiatan dan beberapa pengaturan. Berikut merupakan tampilan dari menu ketik Enot.

Gambar 4.8 Tampilan Menu Ketik Enot

4.1.3.6 Menu Kalender

Menu kalender merupakan menu untuk melihat kalender, semua kegiatan akan ditampilkan dengan format *list* pada setiap tanggalnya. Berikut merupakan tampilan dari menu kalender Enot.

Gambar 4.9 Tampilan Menu Kalender

4.1.3.7 List Kegiatan

List kegiatan merupakan list yang menampilkan judul dari beberapa kegiatan yang di buat oleh member. Saat satu judul di *tap*, maka akan menuju ke menu edit. Tapi jika pada tanggal tersebut tidak ada inputan kegiatan apapun dari member, berarti list akan menampilkan kosong.

Gambar 4.10 Tampilan List Kegiatan Enot

4.1.3.8 Menu Edit

Ada 3 menu edit, yaitu edit menu foto, edit menu suara, dan edit menu ketik. Pada menu edit, member dapat mengupdate kegiatan yang di inputkan, member juda bisa menghapus, dan member juga bisa berbagi.

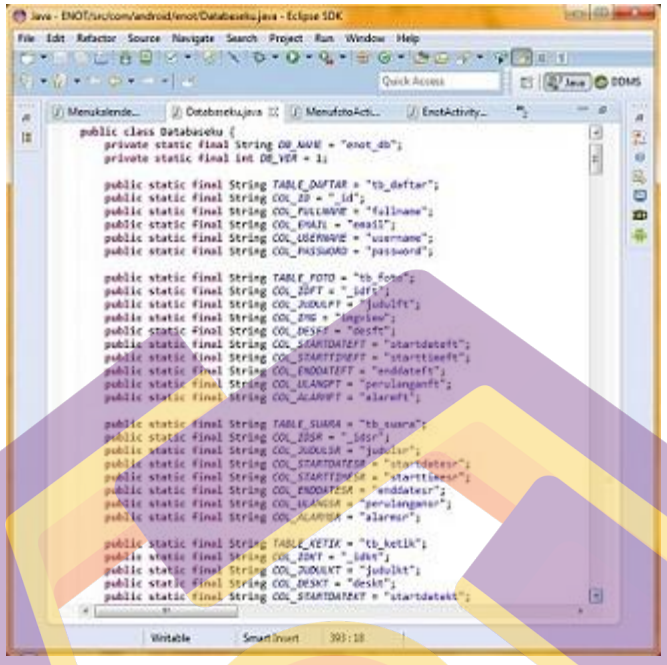
Gambar 4.11 Contoh Tampilan Menu Edit Foto

Gambar 4.12 Tampilan Menu Edit Suara

Gambar 4.13 Tampilan Menu Edit Ketik

4.1.4 Membuat Database ENOT

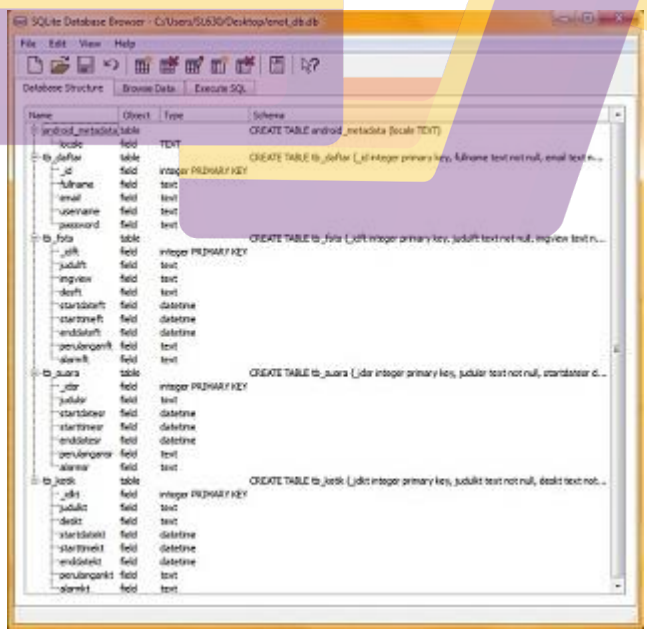
Membuat database dengan nama `enot_db` pada *class* `Database.java`, berisi 4 tabel yaitu `tb_daftar`, `tb_foto`, `tb_suara`, dan `tb_ketik`.



Gambar 4.14 Code Database ENOT

4.1.5 Cek Database Di SQLite Browser

Setelah berhasil di *create*, untuk mengecek apakah database yang dibuat berhasil atau tidak. Cek pada SQLite Browser.

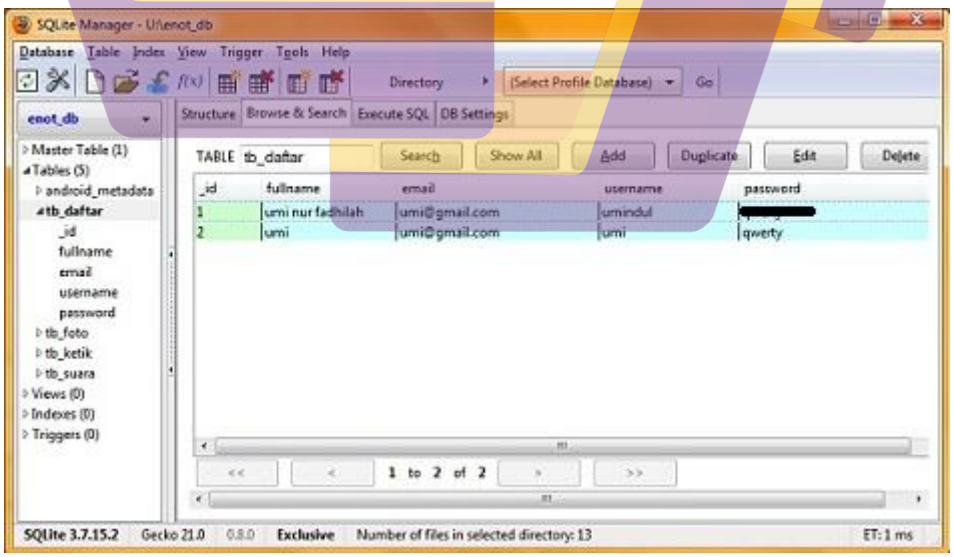


Gambar 4.15 Tampilan Database ENOT pada SQLite Browser

4.1.6 Mencoba Insert di Form Daftar

Mencoba *insert* pada `tb_daftar`, apakah data yang diinputkan bisa tersimpan atau tidak. Setelah itu mencoba mengakses *SQLite Manager* untuk melihat, apakah sudah masuk ke database atau belum.

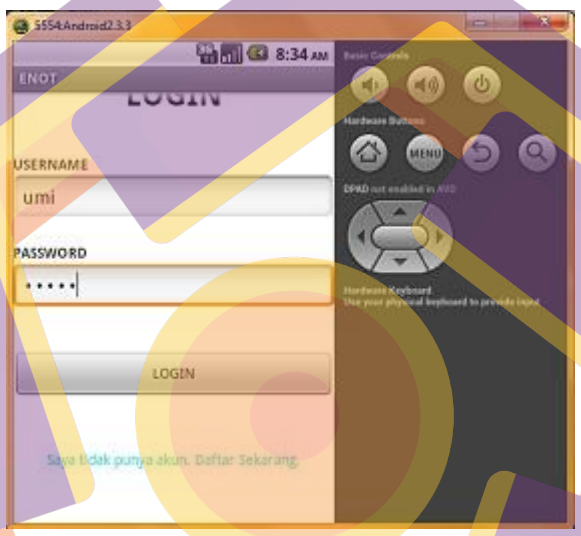
Gambar 4.16 Daftar Menjadi Member ENOT



Gambar 4.17 Data Berhasil Disimpan

4.1.7 Mengakses *Form* Login

Mencoba mengakses *form login* menggunakan data dati `tb_daftar`. *Username* dan *password* yang diinputkan adalah data yang diisikan pada *form* daftar. Jika berhasil login, maka akan langsung menuju ke menu utama / *dashboard*.



Gambar 4.18 Menginputkan Data Pada Form Login

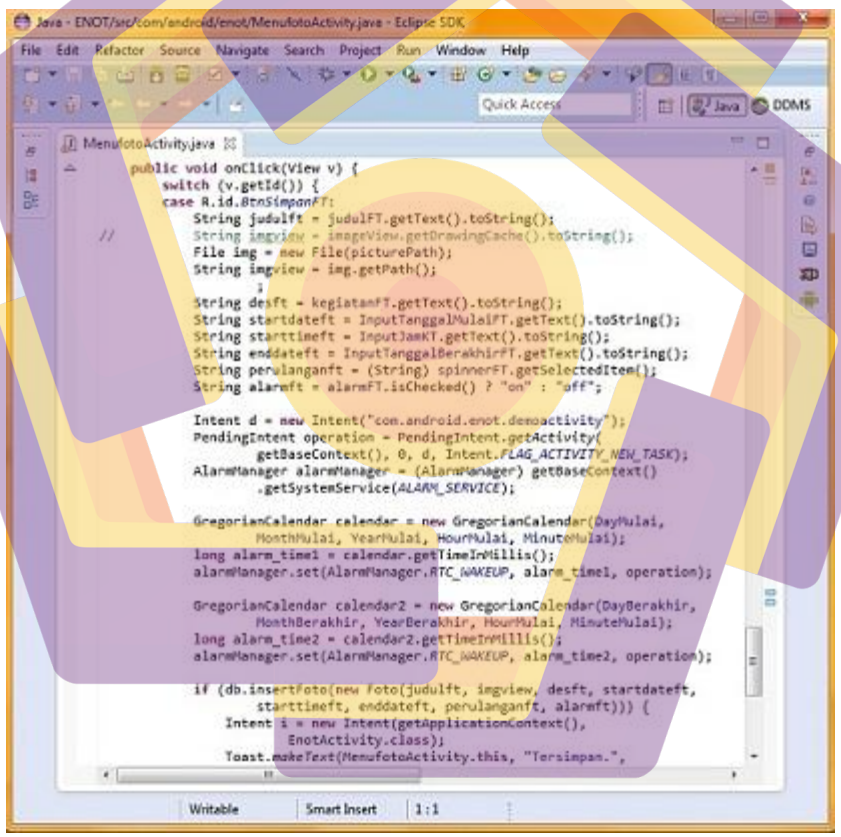
Gambar 4.19 Menu Utama ENOT

4.1.8 Memberikan *Code* Pada Menu Dashboard

Terdapat 4 menu utama pada *dashboard* yaitu menu foto, menu suara dan menu ketik. Juga terdapat *button* keluar, yang berfungsi untuk keluar dari aplikasi.

4.1.8.1 Memberikan *Code* Pada Menu Foto

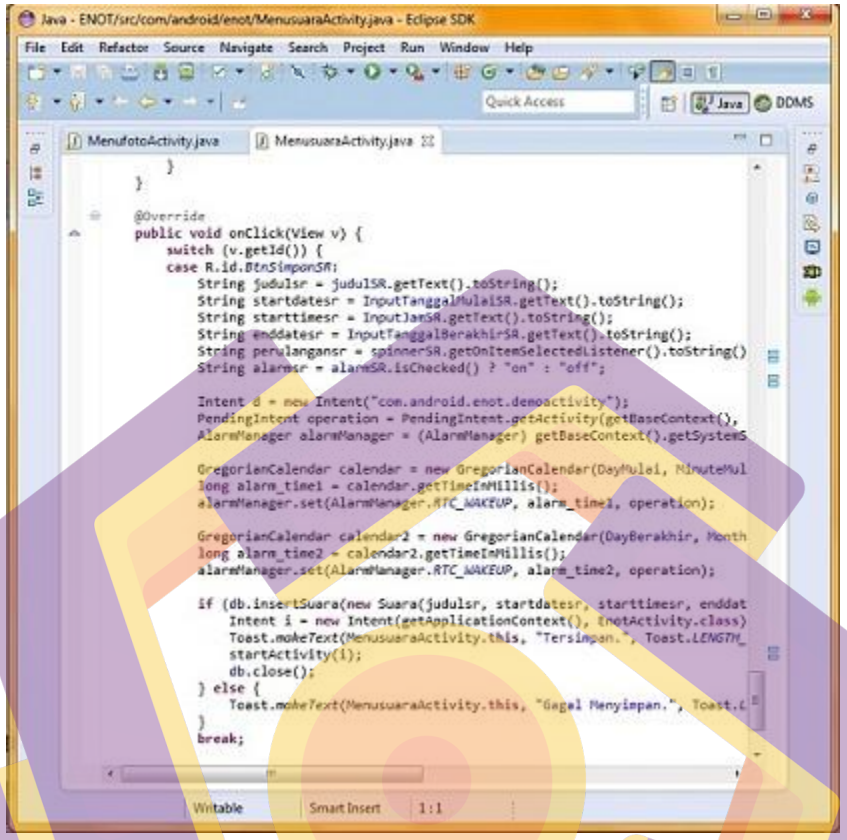
Menu foto adalah menu yang digunakan untuk menginputkan kegiatan beserta foto yang di ambil dari *sdcard*.



Gambar 4.20 *Code* Pada Menu Foto

4.1.8.2 Memberikan *Code* Pada Menu Suara

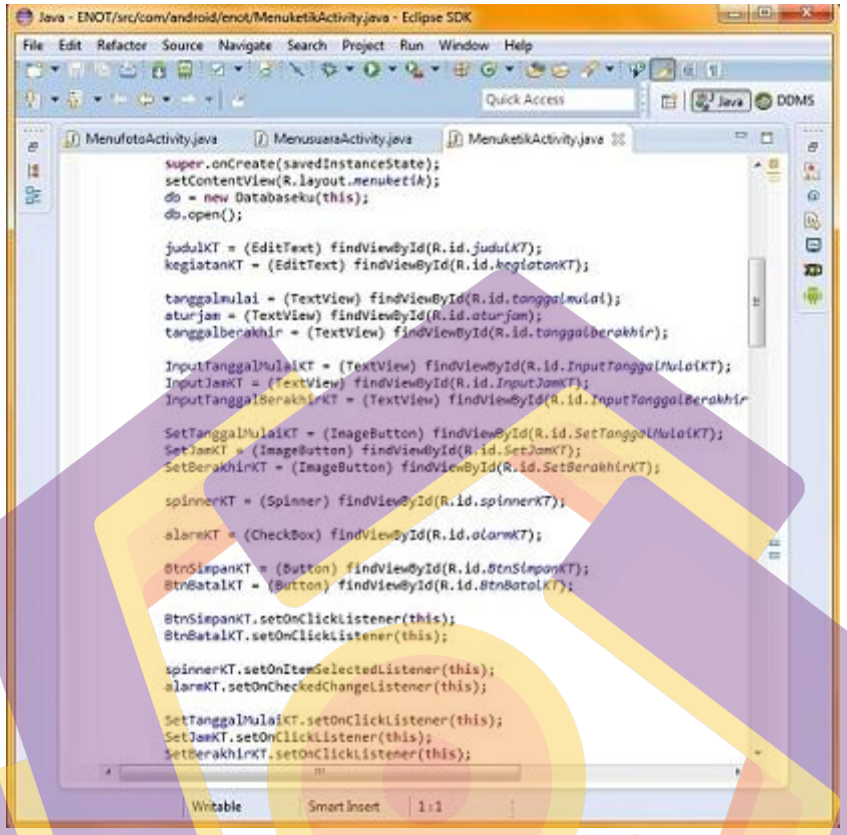
Menu suara adalah menu yang digunakan untuk menginputkan kegiatan beserta suara.



Gambar 4.21 *Code* Pada Menu Suara

4.1.8.3 Memberikan *Code* Pada Menu Ketik

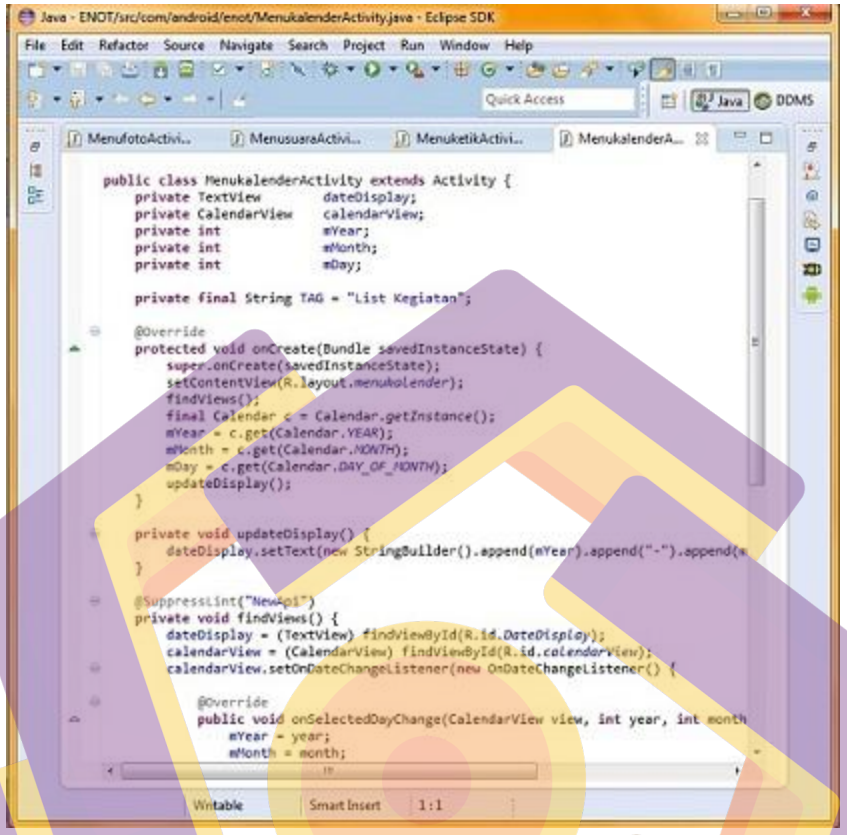
Menu ketik adalah menu yang digunakan untuk menginputkan kegiatan, member menginputkan kegiatan dengan mengetik.



Gambar 4.22 *Code* Pada Menu Suara

4.1.8.4 Memberikan *Code* Pada Menu Kalender

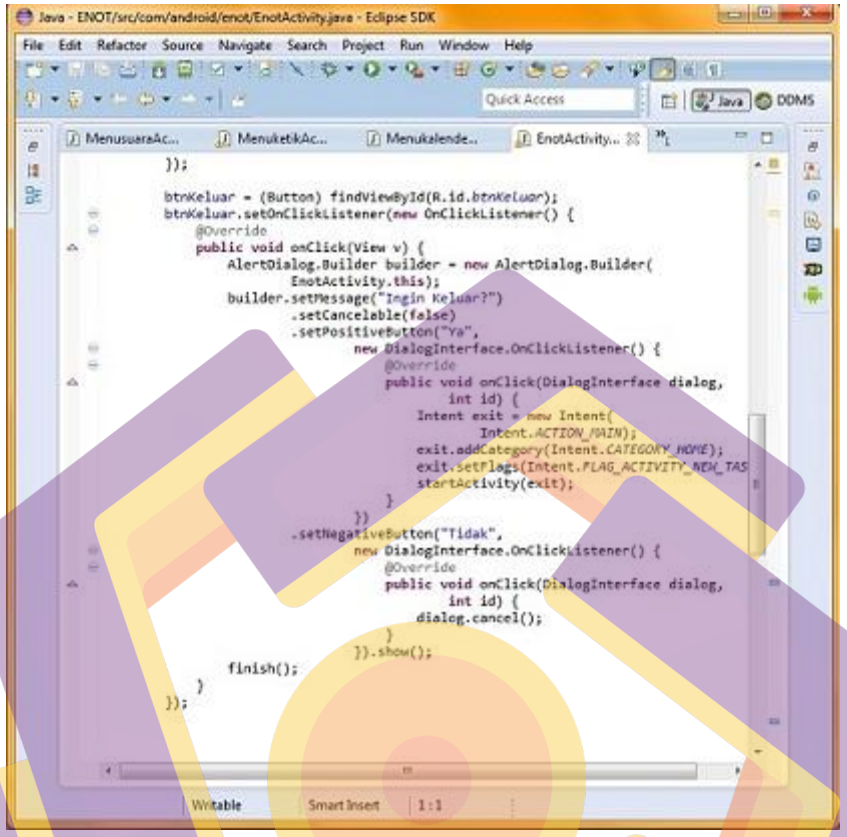
Menu kalender digunakan user untuk melihat kalender kegiatan yang disimpan.



Gambar 4.23 *Code* Pada Menu Kalender

4.1.8.5 Memberikan *Code* Pada *Button* Keluar

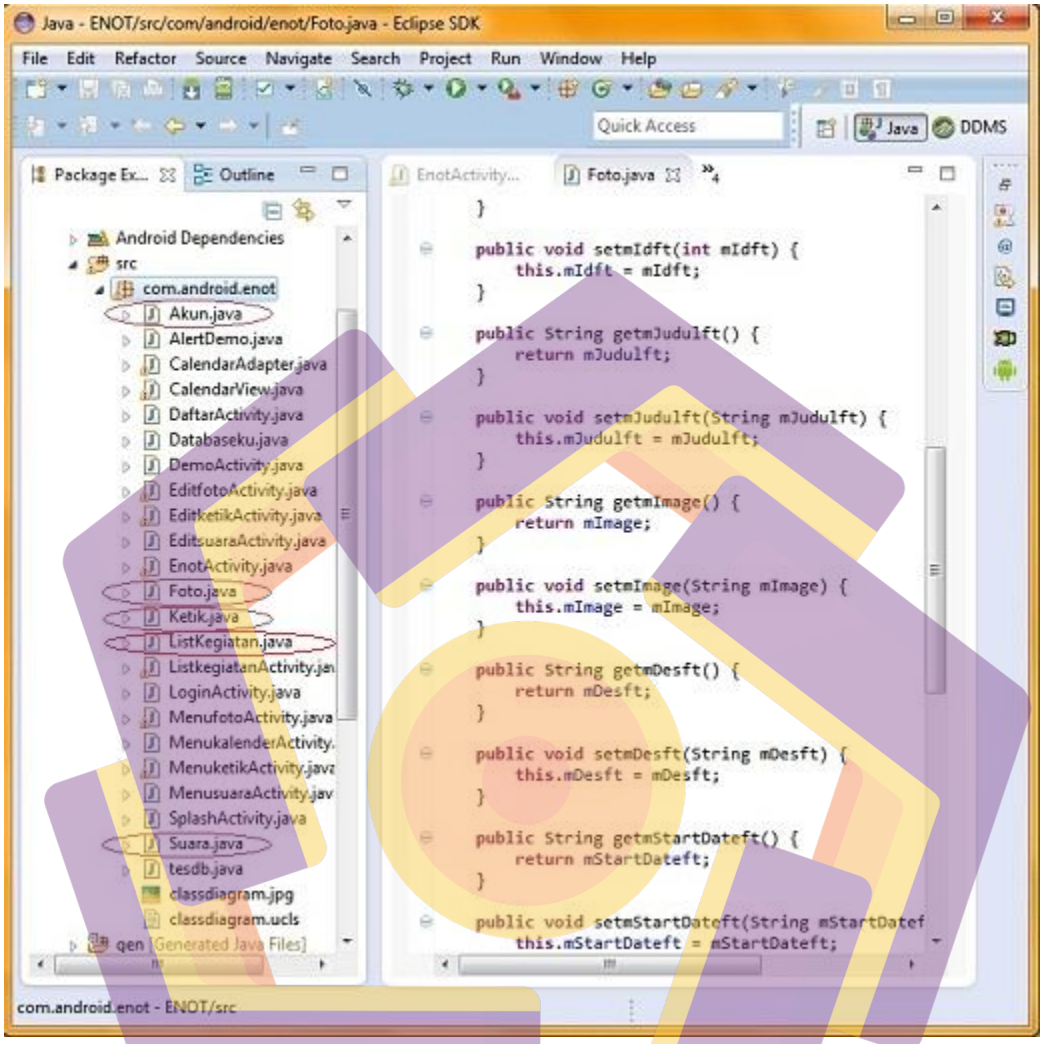
Button keluar dimanfaatkan user untuk keluar dari aplikasi ENOT.



Gambar 4.24 *Code* Pada *Button* Keluar

4.1.9 Membuat *Class* Model

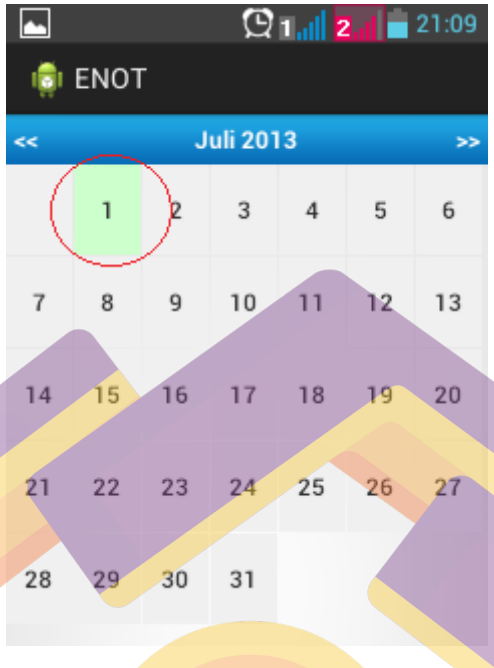
Class model digunakan untuk penghubung antara `Activity` dengan *class* `Databaseku.java`. Terdapat 5 *class* model `Akun.java`, `Foto.java`, `Ketik.java`, `Suara.java`, `ListKegiatan.java`.



Gambar 4.25 *Class* Model Untuk Penghubung

4.1.10 Menampilkan Semua Kegiatan

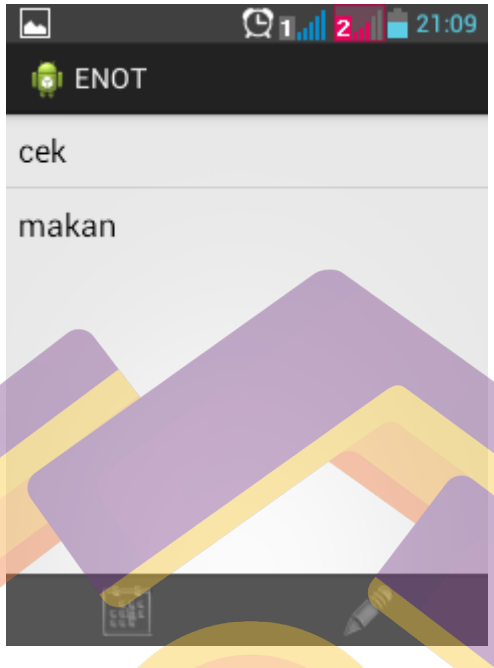
Menampilkan bagaimana semua kegiatan yang sudah tersimpan muncul di menu kalender. Warna kalender akan berubah sedikit cerah, yang menandakan pada tanggal tersebut, ada kegiatan yang diinputkan member enot.



Gambar 4.26 Kegiatan Di Menu Kalender

4.1.11 Menampilkan *Single* Kegiatan

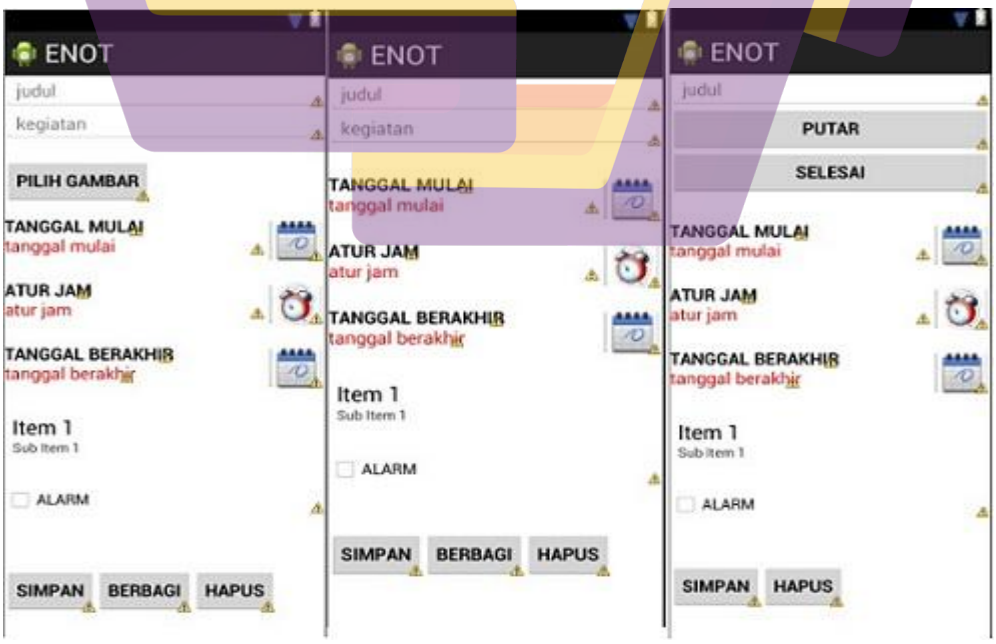
Pada halaman *list*, kegiatan ditampilkan menggunakan *class* model `ListKegiatan.java`. Hasil inputan akan bertambah secara otomatis pada *list* saat member enot menginputkan kegiatan. Pada halaman kalender, warna angka akan berubah sedikit cerah, menandakan pada tanggal tersebut ada kegiatan yang tersimpan. Tetapi jika tidak ada kegiatan yang tersimpan, saat tanggal ditekan, akan menuju ke halaman kososng.



Gambar 2.27 *Single* Kegiatan di List

4.1.2 Membuat Layout Edit

Form edit digunakan saat member ingin mengedit kegiatan yang sudah disimpan.



Gambar 4.28 Layout Menu Edit Foto, Suara, Ketik

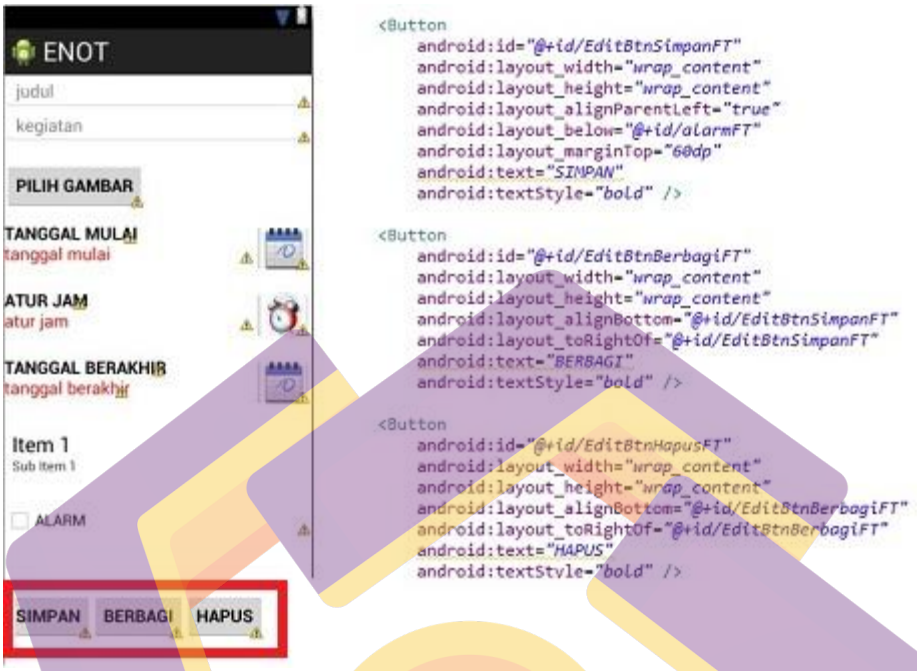
4.1.13 Contoh Saat List Dipilih Menuju Ke Menu Edit

Pada halaman edit ini, adalah single kegiatan dari list kegiatan yang ditekan. Pada halaman ini, member bisa mengedit ulang isi kegiatan, atau mengedit waktu.

Gambar 2.29 Menuju Ke Menu Edit

4.1.14 Memberikan 3 *Button* Pada Halaman Edit

Button yang diberikan adalah simpan, berbagi dan hapus. *Button* simpan digunakan untuk menyimpan kembali kegiatan yang sudah selesai diperbaharui. *Button* berbagi digunakan untuk berbagi kegiatan dibeberapa jejaring sosial, untuk menggunakan *button* ini dibutuhkan koneksi internet. *Button* hapus digunakan untuk menghapus kegiatan yang ada.



Gambar 4.30 *Button* Pada Halaman Edit

4.1.15 Pengujian Program Pada Beberapa *Smartphone*

Pengujian program dilakukan untuk mengecek, apakah program dapat berjalan baik di beberapa perangkat *smartphone* atau tidak.

Tabel 4.1 Uji Program Pada Bebrapa Perangkat Smartphone

No	Smartphone	Keterangan
1.	Tab 7” GT-P3100	Berjalan Baik
2.	Sony Xperia Soal	Berjalan Baik
3.	Samsung Galaxy Note	Berjalan Baik
4.	LG E435	Berjalan Baik

4.1.16 Pemeliharaan Program

Pemeliharaan program dilakukan untuk menghindari kebutuhan sistem ENOT yang masih kurang. Dilakukan dengan :

a. Mengecek jangka pendek, pengecekan akan dilakukan setiap 3 bulan sekali. Untuk melihat apakah sistem berjalan dengan baik atau tidak.

b. Melakukan update jika ada perkembangan atau perubahan dari pengembang.

4.2. Pembahasan

Uji coba program bertujuan untuk menghindari kesalahan pada program yang dibuat. Pengujian program biasanya dilakukan selama proses *coding* dan setelah program aplikasi selesai dibuat.

Ada tidaknya bentuk kesalahan yang memungkinkan ditemukan pada proses pengujian, yaitu sebagai berikut[10] :

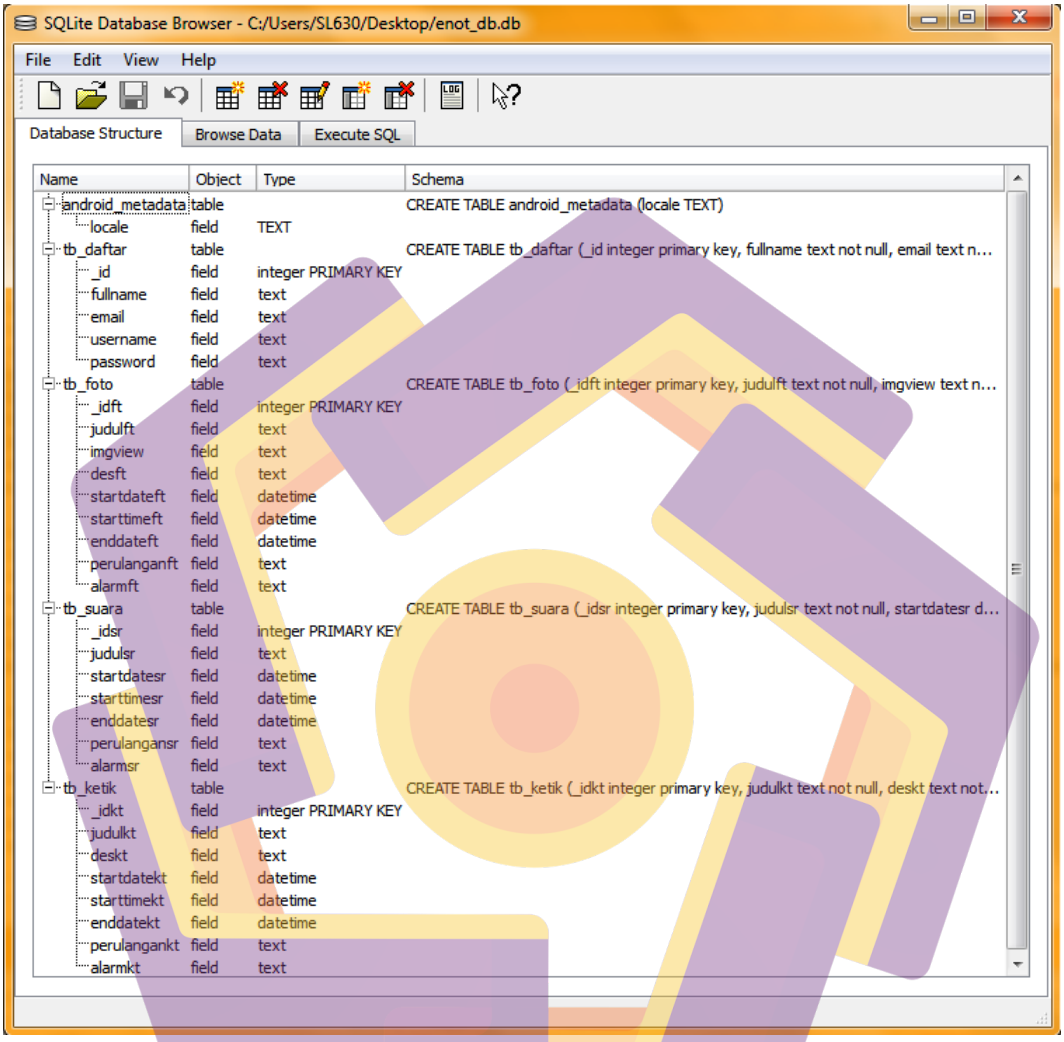
4.2.1 *Code Development and Integration*

Proyek aplikasi *mobile* memerlukan pengembangan *code* dan penyatuan denagn aplikasi yang ada.

1. Coding

Aplikasi *coding* melibatkan penciptaan dan / atau modifikasi beberapa jenis perangkat seperti database, *script test*, *release script*. Berikut adalah hasil dari database yang di *create* pada class `Databaseku.java`.

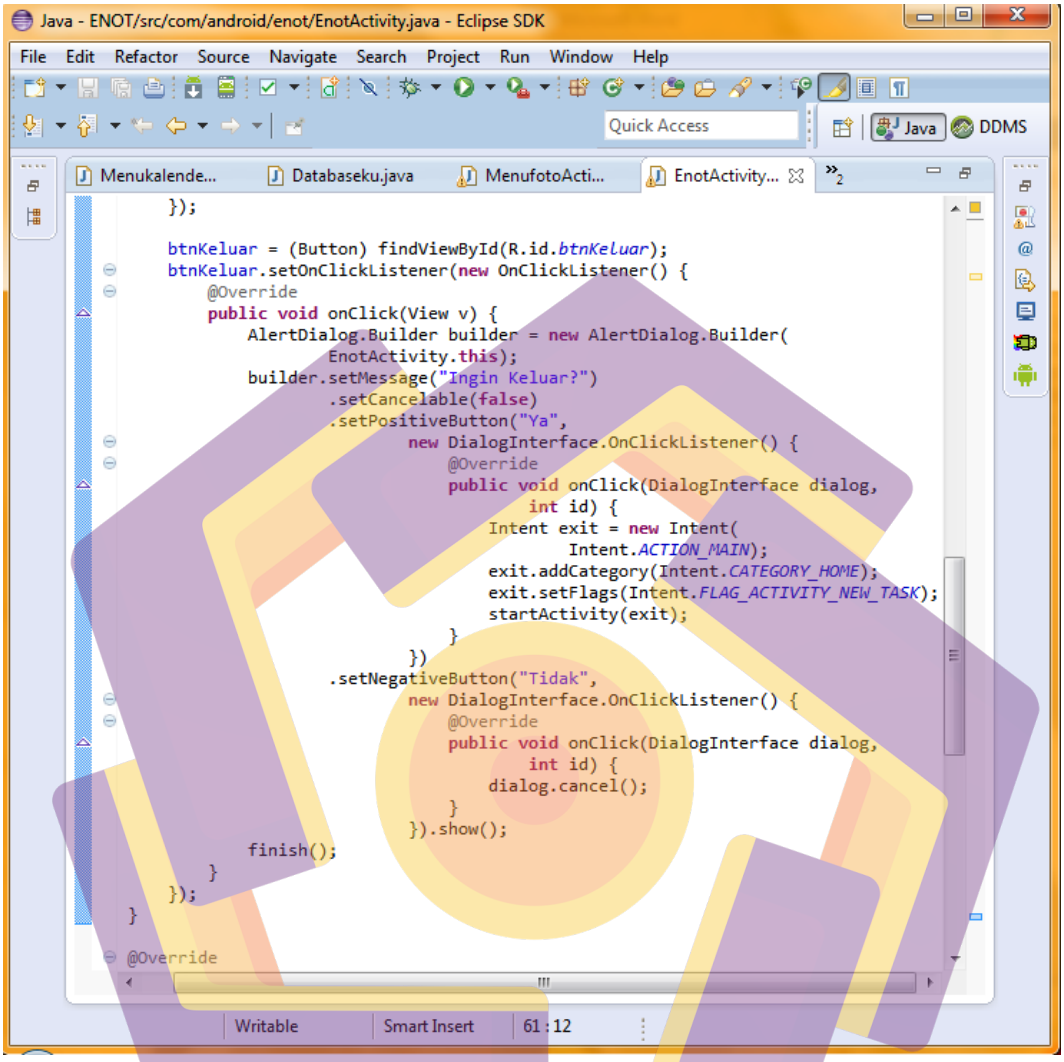
[10] Valentino Lee, Heather Schneider, Robbie Schell. Mobile Applications : Architecture, Design, and Development. Pearson Education. New Jersey. 2004 Hal 180



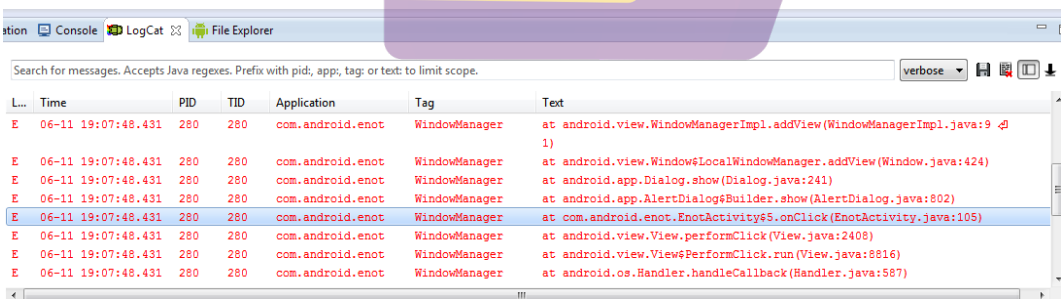
Gambar 4.31 Tampilan Database di SQLite Browser

2. Unit Testing

Terdiri dari pengujian potongan fungsional *code*. Langkah ini memungkinkan untuk malihat *code* di baris demi baris, apakah ada yang *error* atau tidak. Di *workspace* mungkin tidak terjadi kesalahan, tapi *logcat* bisa memberi tahu dimana letak kesalahan *code*. Berikut adalah contoh kesalahan yang tidak terlihat di *workscape*, tapi muncul di *logcat*.



Gambar 4.32 Tampilan *Source Code Button* Keluar



Gambar 4.33 Kesalahan Button Keluar Muncul di Logcat

4.2.2 *Integration and System Testing*

1. Testing Proses

Ada pendekatan untuk pengujian integrasi seperti pengujian *white box* dan *black box*. Pengujian *white box* yaitu menguji komponen internal secara lebih rinci seperti mekanisme otentikasi terhadap database. Sedangkan pengujian black box lebih kepada pendekatan yang lebih umum. Berikut adalah contoh pengujian pada Login yang menginputkan *username* dan *password*.

```
    public boolean checkLogin(Akun akun) {
        Cursor cursor = db.query(TABLE_DAFTAR, new String[] { COL_ID,
                COL_USERNAME, COL_PASSWORD },
                COL_USERNAME + "='" + akun.getUsername() + "' AND "
                        + COL_PASSWORD + "='" + akun.getPassword() + "' ",
                null, null, null, null);
        if (cursor != null)
            cursor.moveToFirst();
        if (cursor.getCount() == 1) {
            return true;
        } else {
            return false;
        }
    }
}
```

Gambar 4.34 Tampilan *Source Code* Otentikasi Login

2. *Testing Documentation*

Tes biasanya didokumentasikan dalam beberapa kasus tunggal atau keseluruhan. Saat pengujian integrasi sedang berlangsung, itu bisa bermanfaat untuk mendokumentasikan, sehingga *code* dan hasil pengujian bisa dipantau dengan jelas. Pada pengujian Aplikasi Enot, tek dokumentai dilakukan tahap demi tahap yaitu saat beberapa *class* bisa selesai diberi *code*. Pengujian akan dipantau dari *logcat*. Jika tidak ada *force close*, berarti aplikasi baik-baik saja. Tetapi jika tiba-tiba Aplikasi Enot berhenti, *logcat* akan menunjukan di mana letak kesalahan.

3. Testing Considerations

Uji kasus harus dilakukan dari berbagai sudut pandang, sebagai berikut :

a. Fungsi dan kegunaan : menguji fungsional secara keseluruhan adalah pertimbangan utama, seperti menguji antarmuka. Untuk menguji kegunaan, enot tampil dengan antar muka yang sederhana. Sehingga member tidak harus menggunakan banyak fitur.

b. Pengujian kinerja : dilakukan untuk mengetahui beban lalu lintas, jumlah pengguna, jumlah halaman yang diakses, server *uptime* dan *downtime*. Kerena Enot merupakan aplikasi *offline*, maka tidak bisa dipantau untuk pengguna yang tidak tersambung dengan PC untuk melihat *logcat*.

c. Pengujian keamanan : dapat dilakukan dengan berbagai tindakan termasuk deteksi virus, kuncian, perlindungan *firewall*. Untuk pengujian keamanan, enot sudah didukung oleh *username* dan *password*, sehingga tidak sembarang orang bisa mengakses kegiatan member.

d. Regresi pengujian : adalah hasil dari uji fungsi, uji kinerja dan uji keamanan. Dan dapat ditarik kesimpulan, seperti pada sub bab selanjutnya.

4.3 PEMBAHASAN

4.3.1 Pembahasan Listing Program

Penulisan program bertujuan untuk mengimplementasikan rancangan yang sudah dibuat dengan menuliskan perintah-perintah atau logika. Langkah ini

merupakan salah satu tahapan dari tahap implementasi sehingga hasil akhir implementasi sesuai dengan rancangan yang telah dibuat.

Dalam pembuatan aplikasi Enot ini, menggunakan Eclipse Juno sebagai editor. Hasil pengkodean diuji coba menggunakan emulator yang satu paket dengan Eclipse Juno.

4.3.1.1 Pembahasan Daftar

Potongan *code* berikut, membawa data yang diisikan pada *form* daftar akan di bawa ke tabel `tb_daftar` menggunakan `db.insertDaftar`. Jika berhasil *insert*, maka akan menuju ke halaman Login. Dibawa dengan `Intent i`. Tetapi jika gagal insert data, maka akan ada `Toast` "Gagal Daftar."

```
@Override
public void onClick(View v)
{
    switch (v.getId())
    {
        case R.id.btnRegister:

            String fullname = registerFullname.getText().toString();
            String email = registerEmail.getText().toString();
            String username = registerUsername.getText().toString();
            String password = registerPassword.getText().toString();

            if (db.insertDaftar(new Akun(username, password, email, fullname))) {
                Intent i = new Intent(getApplicationContext(), LoginActivity.class);
                startActivity(i);
                finish();
                db.close();
            } else {
                Toast.makeText(DaftarActivity.this, "Gagal Daftar",
                        Toast.LENGTH_SHORT).show();
            }
```

Gambar 4.35 Potongan *Listing Code* Daftar

4.3.1.2 Pembahasan Login

Potongan *code* berikut, adalah proses mengecek data pada `tb_daftar` menggunakan model `Akun.java`. Yang diambil pada `tb_daftar` adalah

username dan *password*. Karena pada *form* login, member enot harus menginputkan *username* dan *password*. Proses pengecekan dilakukan menggunakan `cursor`, jika data terdapat pada `tb_daftar` (`==1`), maka login akan berhasil. Jika tidak ada data, maka gagal login.

```
public boolean checkLogin(Akun akun) {
    Cursor cursor = db.query(TABLE_DAFTAR, new String[] { COL_ID,
            COL_USERNAME, COL_PASSWORD },
            COL_USERNAME + "='" + akun.getUsername() + "' AND "
                    + COL_PASSWORD + "='" + akun.getPassword() + "' ",
            null, null, null, null);
    if (cursor != null)
        cursor.moveToFirst();
    if (cursor.getCount() == 1) {
        return true;
    } else {
        return false;
    }
}
```

Gambar 4.36 Potongan *Listing Code* Login

4.3.1.3 Pembahasan Simpan

Potongan *code* berikut menjelaskan proses penyimpanan data yang sudah diinputkan member. Data dibawa menuju tabel `tb_foto` menggunakan *class* model `Foto.java`. Jika data berhasil disimpan, akan muncul `Toast` "Tersimpan." Tetapi jika data member gagal disimpan, akan muncul `Toast` "Gagal Menyimpan." *Button* simpan ada pada menu foto, menu suara, menu ketik. Pada halaman edit juga terdapat *button* simpan yang fungsinya juga membawa data menuju tabel.

```
if (db.insertFoto(new Foto(judulft, imgview, desft, startdateft,
        starttimeft, enddateft, perulanganft, alarmft))) {
    Intent i = new Intent(getApplicationContext(),
            EnotActivity.class);
    Toast.makeText(MenufotoActivity.this, "Tersimpan.",
            Toast.LENGTH_SHORT).show();
    startActivity(i);
    db.close();
} else {
    Toast.makeText(MenufotoActivity.this, "Gagal Menyimpan.",
            Toast.LENGTH_SHORT).show();
}
break;
```

Gambar 4.37 Potongan *Listing Code* Simpan

4.3.1.4 Pembahasan *Code* Memanggil Foto

Potongan *code* berikut menjelaskan proses pengambilan gambar dari *sdcard* (media penyimpanan) dan akan ditampilkan pada `ImageView` yang sudah disediakan pada *layout*.

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (requestCode == RESULT_LOAD_IMAGE && resultCode == RESULT_OK
            && null != data) {
        Uri selectedImage = data.getData();
        String[] filePathColumn = { MediaColumns.DATA };

        Cursor cursor = getContentResolver().query(selectedImage,
                filePathColumn, null, null, null);
        cursor.moveToFirst();

        int columnIndex = cursor.getColumnIndex(filePathColumn[0]);
        picturePath = cursor.getString(columnIndex);
        cursor.close();

        imageView = (ImageView) findViewById(R.id.ImgView);
        imageView.setImageBitmap(BitmapFactory.decodeFile(picturePath));
    }
}
```

Gambar 4.38 Potongan *Listing Code* Memanggil Foto

4.3.1.5 Pembahasan Code Pada Menu Suara

Potongan *code* berikut, adalah *code* untuk merekam suara. Menggunakan `MediaRecorder()`, `MIC`, hasilnya `THREE_GPP` hasilnya akan disimpan di *sdcard* suaranya `AMR_NB`. Saat *button* mulai merekam ditekan, akan muncul `Toast` "mulai" dan proses merekam berlangsung.

```
private void beginRecording() throws Exception {
    ditchMediaRecorder();
    File ouFile = new File(OUTPUT_FILE);

    if (ouFile.exists())
        ouFile.delete();

    recorder = new MediaRecorder();
    recorder.setAudioSource(MediaRecorder.AudioSource.MIC);
    recorder.setOutputFormat(MediaRecorder.OutputFormat.THREE_GPP);
    recorder.setAudioEncoder(MediaRecorder.AudioEncoder.AMR_NB);
    recorder.setOutputFile(OUTPUT_FILE);
    recorder.prepare();
    recorder.start();
    Toast.makeText(getBaseContext(), "mulai", Toast.LENGTH_LONG).show();
}
```

Gambar 4.39 Potongan *Listing Code* Mulai Merekam

Potongan *code* berikut adalah *code* yang menjelaskan berhenti merekam. Dengan `recorder.stop()` dan saat *button* selesai merekam ditekan, akan muncul `Toast` "selesai."

```
private void stopRecording() {
    if (recorder != null)
        recorder.stop();
    Toast.makeText(getBaseContext(), "mulai", Toast.LENGTH_LONG).show();
    Log.d("rec", "selesai");
}
```

Gambar 4.40 Potongan *Listing Code* Selesai Merekam

Potongan *code* berikut menjelaskan proses memutar ulang hasil yang sudah direkam. Dengan `mediaPlayer.setDataSource(OUTPUT_FILE)` berarti mengambil data di tempat penyimpanan yaitu *sdcard*.

```
private void playRecording() throws Exception {
    ditchMediaRecorder();
    mediaPlayer = new MediaPlayer();
    mediaPlayer.setDataSource(OUTPUT_FILE);
    mediaPlayer.prepare();
    mediaPlayer.start();
    Log.d("play", "mulai");
}
```

Gambar 4.41 Potongan *Listing Code* Memutar Hasil Rekaman

Potongan *code* berikut adalah *code* selesai memutar hasil rekaman yang tersimpan.

```
private void stopPlayback() {
    if (mediaPlayer != null)
        mediaPlayer.stop();

    Log.d("play", "stop");

}
```

Gambar 4.42 Potongan *Listing Code* Selesai Memutar Rekaman

4.3.1.6 Pembahasan Set Waktu

Potongan *code* berikut adalah *code* saat member men-set waktu, berupa jam dan menit. Akan muncul `timePickerDialog.show()` akan menampilkan dialog agar member bisa menginputkan waktu.

```
private void setWaktu() {
    Calendar calendar = Calendar.getInstance();

    timePickerDialog = new TimePickerDialog(MenuketikActivity.this, onTimeSetListener,
            calendar.get(Calendar.HOUR_OF_DAY), calendar.get(Calendar.MINUTE), true);
    timePickerDialog.show();
}

OnTimeSetListener onTimeSetListener = new OnTimeSetListener() {

    @Override
    public void onTimeSet(TimePicker view, int hourOfDay, int minute) {
        Calendar calNow = Calendar.getInstance();
        Calendar calSet = (Calendar) calNow.clone();

        calSet.set(Calendar.HOUR_OF_DAY, hourOfDay);
        calSet.set(Calendar.MINUTE, minute);
        calSet.set(Calendar.SECOND, 0);
        calSet.set(Calendar.MILLISECOND, 0);

        InputJamKT.setText(hourOfDay + "-" + minute);
        HourMulai = hourOfDay;
        MinuteMulai = minute;
    }
};
```

Gambar 4.43 Potongan *Listing Code* Set Waktu

4.3.1.7 Pembahasan Set Tanggal

Potongan *code* berikut adalah *code* untuk men-set tanggal mulai dan tanggal berakhir. Untuk tanggal mulai dibahas dengan `jenis 1` (`case 1`), sedangkan untuk tanggal berakhir dibahas dengan `jenis 2` (`case 2`). Dialog tanggal akan muncul dengan fungsi `datePickerDialog.show()` dan dengan fungsi `onDateSetListener` digunakan member untuk men-set tanggal.

```
private void setTanggal(int jenis) {
    Calendar calendar = Calendar.getInstance();
    tanggal = jenis;
    datePickerDialog = new DatePickerDialog(MenuketikActivity.this,
            onDateSetListener, calendar.get(Calendar.YEAR),
            calendar.get(Calendar.MONTH),
            calendar.get(Calendar.DAY_OF_MONTH));
    datePickerDialog.show();
}
OnDateSetListener onDateSetListener = new OnDateSetListener() {
    @Override
    public void onDateSet(DatePicker view, int year, int monthOfYear,
            int dayOfMonth) {
        Calendar calNow = Calendar.getInstance();
        Calendar calSet = (Calendar) calNow.clone();

        calSet.set(Calendar.YEAR, year);
        calSet.set(Calendar.MONTH, monthOfYear);
        calSet.set(Calendar.DAY_OF_MONTH, dayOfMonth);
        switch (tanggal) {
        case 1:
            InputTanggalMulaiKT.setText(dayOfMonth + "-"
                    + (monthOfYear + 1) + "-" + year);
            DayMulai = dayOfMonth;
            MonthMulai = monthOfYear;
            YearMulai = year;
            break;
        case 2:
            InputTanggalBerakhirKT.setText(dayOfMonth + "-"
                    + (monthOfYear + 1) + "-" + year);
            DayBerakhir = dayOfMonth;
            MonthBerakhir = monthOfYear;
            YearBerakhir = year;
            break;
        }
    }
};
```

Gambar 4.44 Potongan *Listing Code* Set Tanggal

4.3.1.8 Pembahasan Menu Kalender

Potongan code berikut adalah potongan yang menampilkan menu kalender dengan `setContentView(R.layout.calendar)`, menjelaskan bahwa *layout* `calendar.xml` ditampilkan. Di menu kalender, database juga dibuka. Kalender tampil, dengan tampilan per 1 bulan penuh. Sedangkan

`CalendarAdapter`, digunakan untuk menghubungkan menu kalender dengan database.

```
super.onCreate(savedInstanceState);
setContentView(R.layout.calendar);
db = new Databaseku(this);
db.open();

month = Calendar.getInstance();
onNewIntent(getIntent());

items = new ArrayList<String>();
adapter = new CalendarAdapter(this, month);
```

Gambar 4.45 Potongan *Listing Code* Menampilkan Kalender

4.3.1.9 Pembahasan Tampil Kegiatan Pada Menu Kalender

Potongan *code* berikut adalah cara menampilkan kegiatan pada menu kalender dipanggil dari bulan dan tahun menggunakan *query* `SELECT * FROM tb_ketik WHERE ltrim(startdatekt, '123456789') = ' " + tgl + " ' "`. *Code* `ltrim` artinya menghapus angka di sebelah kiri.

```
public Cursor getKetikPerbulan(int bulan, int tahun){
    String tgl = "-" + String.valueOf(bulan) + "-" + String.valueOf(tahun);
    String sql = "SELECT * FROM tb_ketik WHERE ltrim(startdatekt,'1234567890') = '" + tgl + "'";
    Log.d("query", sql);
    return db.rawQuery(sql, null);
}
```

Gambar 4.46 Potongan Listing Code Cara Menampilkan Kegiatan

4.3.1.10 Pembahasan Tampil List Kegiatan

Potongan *code* berikut adalah pemanggilan pada list kegiatan berdasarkan tanggal, dan menggunakan *query* `SELECT * FROM tb_ketik WHERE startdatekt = ' " + tanggal + " ' "`.

```
public Cursor getKetikPertanggal(String tanggal){
    String sql = "SELECT * FROM tb_ketik WHERE startdatekt = '" + tanggal + "'";
    return db.rawQuery(sql, null);
}
```

Gambar 4.47 Potongan Listing Code Cara Menampilkan List

4.3.1.11 Pembahasan Button Berbagi

Potongan *code* berikut adalah *code* yang memanfaatkan intent untuk membungkus data, kemudian data tersebut dibawa ke aplikasi lain menggunakan fungsi `ACTION_SEND`. Pengguna aplikasi akan disajikan beberapa pilihan aplikasi lain dengan fungsi `createChooser()`. *Method* `startActivity()` digunakan untuk memanggil aplikasi lain melalui `chooser`.

```
private void share() {
    Intent sharingIntent = new Intent(Intent.ACTION_SEND);
    sharingIntent.setType("text/html");
    sharingIntent.putExtra(android.content.Intent.EXTRA_TEXT,
            Html.fromHtml("<p>Ini akan dibagikan.</p>"));
    startActivity(Intent
            .createChooser(sharingIntent, "Berbagi Menggunakan"));
}
```

Gambar 4.48 Potongan *Listing Code Button* Berbagi

4.3.1.12 Pembahasan *Button* Hapus

Potongan *code* berikut adalah *code* untuk menghapus, mengambil dari database dan dihapus berdasarkan `id`. Code `db.close()` digunakan untuk menutup database setelah selesai menghapus data.

```
case R.id.EditBtnHapusKT:
    db.deleteKetik(id);
    startActivity(new Intent(EditketikActivity.this,
            ListkegiatanActivity.class));
    Toast.makeText(EditketikActivity.this, "Telah Dihapus.",
            Toast.LENGTH_SHORT).show();
    db.close();
    finish();
    break;
```

Gambar 4.49 Potongan *Listing Code Button* Hapus

4.3.2 Kelemahan dan Keunggulan Enot

Dari hasil uji coba di atas, dapat ditarik kesimpulan sebagai berikut :

Tabel 4.1 Kelemahan dan Keunggulan Enot

Keunggulan Enot	**Kelemahan Enot**
1. Bekerja secara *offline*, membuat hemat baterai. 2. Mempunyai form login, jadi mendukung *security*. 3. Dapat memberikan *alert* dialog.	1. Antarmuka sederhana. 2. Lalu lintas data tidak bisa dipantau, karena Enot adalah aplikasi *offline*. 3. Tidak ada perlindungan *firewall*.

BAB V

PENUTUP

5.1 Kesimpulan

Dari hasil penelitian dan pembahasan materi yang telah dilakukan pada bab-bab sebelumnya, dapat diambil kesimpulan sebagai berikut :

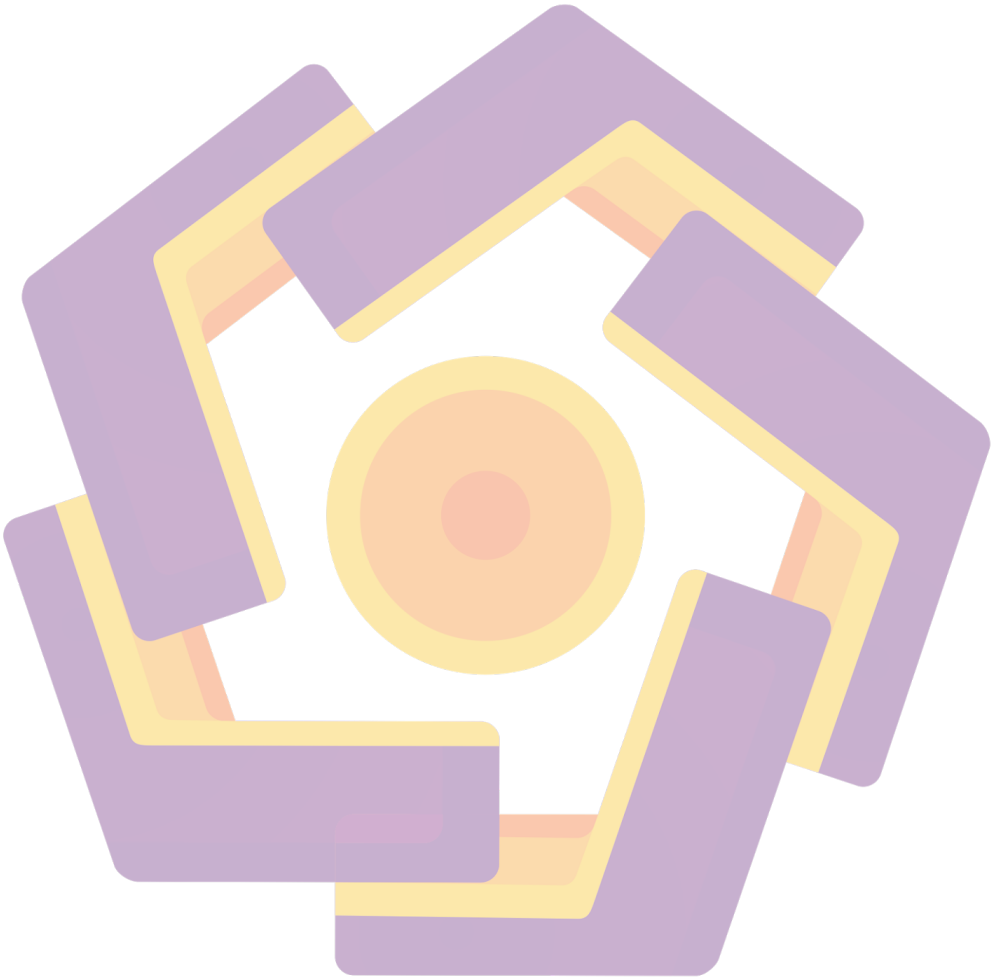
1. Dalam penelitian ini, dibangun sebuah aplikasi Agenga Pribadi Enot Berbasis Android. Aplikasi Enot dibangun menggunakan Eclipse Juno dengan minimal SDK 3.0, karena SDK 3.0 mendukung untuk mengolah kalender dan suara.
2. Ada 4 tabel yang dibuat untuk menyimpan hasil inputan dari form daftar, menu foto, menu suara, menu ketik. Tabel `tb_foto`, `tb_ketik`, `tb_suara` akan ditampilkan pada menu kalender menggunakan *method* yang ada pada *class* database.
3. Untuk menghubungkan beberapa `Activity` dengan *class* database, dibutuhkan *class* model.
4. *Alert dialog* dapat digunakan member untuk pengingat kegiatan yang akan dilakukan. Dengan suara *default* yang sudah di set pada aplikasi.

5.2 Saran

Dari hasil evaluasi terhadap sistem ini, maka didapatkan beberapa saran untuk pengembangan penelitian kedepannya, yaitu :

1. Tampilan interface dapat dirubah agar lebih menarik.
2. Pemberian warning saat OS smartphone berada dibawah ketentuan yaitu 3.0.

3. Menu suara tidak berjalan dengan baik di beberapa *device*.
4. Aplikasi dapat ditambah *capture* foto secara langsung.
5. Aplikasi dapat ditambah dengan *action voice* yaitu *voice to text*.



DAFTAR PUSTAKA

Al Fatta, Hanif. 2007. *Analisis dan Perancangan Sistem Informasi*. Yogyakarta : Penerbit Andi.

Arief, M. Rudyanto. 2006. *Pemrograman Basis Data Menggunakan Transact-SQL dengan Microsoft SQL Server 2000*. Yogyakarta: Andi Offset.

Hakim, Rahmat dan Sutarto. 2009. *Mastering Java : Konsep Pemprograman Java dan Penerapan Untuk Membuat Software Aplikasi*. Jakarta : Elekmedia Komputindo.

Gunadi, Hariman dan Suhendar A. 2002. *Visual Modeling Menggunakan UML dan Rational Rose*. Bandung: Informatika Bandung.

Wibisono, Gunawan dan Gunadi Dwi Hantoro. 2008. *Mobile Broadband*. Bandung: Informatika.

Nugroho, Adi. 2009. *Rekayasa Perangkat Lunak Menggunakan UML dan Java.* Yogyakarta : Penerbit Andi.

Safaat, Nazaruddin. 2011. *Android Pemrograman Aplikasi Mobile Smartphone dan Tablet PC Berbasis Android*. Bandung: Informatika Bandung.

Anonim. 2012. *About SQLite*. http://www.sqlite.org/about.html. Diakses pada : 5 Desember 2012.

Dharwiyanti, Sri dan Romi Satria Wahono. 2003. *Pengantar Unified Modeling Language (UML).* http://setia.staff.gunadarma.ac.id/Downloads/files/6077/Modul_UML.pdf . Diakses pada : 6 Maret 2013

Hartatik. 2012. *Model Proses PL*. elearning.amikom.ac.id/index.php/download/materi/555158-ST063-2/2012/03/20120312_Proses.ppt. Diakses pada : 6 Maret 2013.