Dear all,

IF-UNIKOM-UP-RPL2-NOKELOMPOK-2018

DOKUMEN PEMBANGUNAN PERANGKAT LUNAK

KUMATO

Versi: 1.0.0-26/Final

untuk:

Matakuliah Rekayasa Perangkat Lunak 2

Dipersiapkan oleh:

Kelompok 4, 10116260 Ary Sugiarto

10116263 Farhan Arif N

10116264 Feryanto Abi F

10116236 Anisa Dewi Umi K

10116242 Ahmad Tri Utama

KELAS IF-6

Jurusan Teknik Informatika – Universitas Komputer Indonesia

Jl. Dipati Ukur Nomor 112-114, 40132

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| unikom | **Program Studi**  **Teknik Informatika UNIKOM** | Nomor Dokumen | | Halaman |
| *IF-UNIKOM-UP-RPL2-2018* | | *1/40* |
| Revisi | *1* | *Tgl: 1 November 2018* |

# DAFTAR ISI

[DAFTAR ISI 2](#_Toc536849624)

[1 PENDAHULUAN 4](#_Toc536849625)

[1.1 Latar Belakang Masalah 4](#_Toc536849626)

[1.2 Identifikasi Masalah 5](#_Toc536849627)

[1.3 Maksud dan Tujuan 5](#_Toc536849628)

[1.4 Batasan Masalah 5](#_Toc536849629)

[1.5 Metodologi Penelitian 5](#_Toc536849630)

[1.5.1 Metode Pengumpulan Data 6](#_Toc536849631)

[1.5.2 Metode Pembangunan Perangkat Lunak 10](#_Toc536849632)

[2 TINJAUAN PUSTAKA 13](#_Toc536849633)

[2.1 Validasi Ide 13](#_Toc536849634)

[2.2 Javelin Board 13](#_Toc536849635)

[2.3 Google Design Sprint 14](#_Toc536849636)

[2.4 OOAD dengan UML 15](#_Toc536849637)

[2.4.1 UML 15](#_Toc536849638)

[2.4.2 Usecase 17](#_Toc536849639)

[2.4.3 Usecase scenario 18](#_Toc536849640)

[2.4.4 Activity diagram 18](#_Toc536849641)

[2.4.5 Class diagram 19](#_Toc536849642)

[2.4.6 Sequence diagram 20](#_Toc536849643)

[2.4.7 Deployment Diagram 21](#_Toc536849644)

[2.5 Mobile Development (Native Android) 22](#_Toc536849645)

[2.6 Business Model Canvas 23](#_Toc536849646)

[2.7 Pengujian Black Box 23](#_Toc536849647)

[2.8 Pengujian Acceptance 24](#_Toc536849648)

[3 KEBUTUHAN PERANGKAT LUNAK 24](#_Toc536849649)

[3.1 Analisis Sistem Berjalan 25](#_Toc536849650)

[3.2 Analisis Arsitektur Sistem 25](#_Toc536849651)

[3.3 Spesifikasi Kebutuhan Perangkat Lunak 26](#_Toc536849652)

[3.3.1 Kebutuhan Fungsional 26](#_Toc536849653)

[3.3.2 Kebutuhan Non Fungsional 27](#_Toc536849654)

[3.4 Analisis Kebutuhan Nonfungsional 27](#_Toc536849655)

[3.4.1 Analisis Kebutuhan Perangkat Lunak 27](#_Toc536849656)

[3.4.2 Analisis Kebutuhan Perangkat Keras 28](#_Toc536849657)

[3.4.3 Analisis Kebutuhan Perangkat Pikir 29](#_Toc536849658)

[3.4.4 Analisis Kebutuhan Jaringan 30](#_Toc536849659)

[3.5 Analisis Kebutuhan Data 31](#_Toc536849660)

[3.6 Model Use Case 31](#_Toc536849661)

[*3.6.1* Diagram Use Case 32](#_Toc536849662)

[3.6.2 Definisi Actor 32](#_Toc536849663)

[3.6.3 Definisi Use Case 33](#_Toc536849664)

[3.6.4 Skenario Use Case 33](#_Toc536849665)

[3.7 Model Activity Diagram 37](#_Toc536849666)

[3.8 Analisis Kebutuhan Fungsional 41](#_Toc536849667)

[4 MODEL ANALISIS 41](#_Toc536849668)

[4.1 Diagram Class Analisis 41](#_Toc536849669)

[4.2 Identifikasi Kelas Analisis 41](#_Toc536849670)

[4.3 Realisasi Kelas Terhadap Use Case 42](#_Toc536849671)

[4.3.1 Identifikasi Kelas Analisis tiap Paket 42](#_Toc536849672)

[5 Model Perancangan 43](#_Toc536849673)

[5.1 Realisasi Use Case Tahap Perancangan 43](#_Toc536849674)

[5.1.1 Use Case Tambah Kegiatan 43](#_Toc536849675)

[5.1.2 Use Case Ubah Jadwal Kegiatan 53](#_Toc536849676)

[5.1.3 Use Case Hapus Jadwal Kegiatan 63](#_Toc536849677)

[5.1.4 Use Case Timer Pomodoro 73](#_Toc536849678)

[5.1.5 Use Case Request Data 76](#_Toc536849679)

[5.1.6 Use Case Notifikasi Timer Pomodoro 86](#_Toc536849680)

[5.2 Perancangan Detil Elemen Logical View 89](#_Toc536849681)

[5.2.1 Kelas NoteAdapter 89](#_Toc536849682)

[5.2.2 Kelas DatabaseContract 89](#_Toc536849683)

[5.2.3 Kelas DatabaseHelper 89](#_Toc536849684)

[5.2.4 Kelas NoteHelper 90](#_Toc536849685)

[5.2.5 Kelas Note 90](#_Toc536849686)

[5.2.6 Kelas CustomOnItemClickListener 91](#_Toc536849687)

[5.2.7 Kelas FormAddUpdateActivity 91](#_Toc536849688)

[5.2.8 Kelas MainActivity 93](#_Toc536849689)

[5.2.9 Kelas PomodoroActivity 94](#_Toc536849690)

[5.2.1 Kelas SplashScreenActivity 95](#_Toc536849691)

[5.3 Diagram Kelas Keseluruhan 95](#_Toc536849692)

[5.4 Algoritma / Query 95](#_Toc536849693)

[5.5 Perancangan Data 103](#_Toc536849694)

[5.6 Perancangan Arsitektural Menu 104](#_Toc536849695)

[5.7 Perancangan Antarmuka 105](#_Toc536849696)

[5.7.1 Perancangan antarmuka To Do List 106](#_Toc536849697)

[5.7.2 Perancangan antarmuka Pomodoro 111](#_Toc536849698)

[5.7.3 Deployment Diagram 115](#_Toc536849699)

[6 Implementasi 115](#_Toc536849700)

[6.1 Lingkungan Implementasi 115](#_Toc536849701)

[6.2 Implementasi Data 116](#_Toc536849702)

[6.3 Implementasi Kelas 116](#_Toc536849703)

[6.4 Implementasi Antarmuka 117](#_Toc536849704)

[7 Pengujian 117](#_Toc536849705)

[7.1 Rencana Pengujian 117](#_Toc536849706)

[7.2 Skenario Pengujian 118](#_Toc536849707)

[7.2.1 Skenario Pengujian Fungsional 118](#_Toc536849708)

[7.2.2 Skenario Pengujian Acceptance 119](#_Toc536849709)

[7.3 Hasil Pengujian 119](#_Toc536849710)

[7.3.1 Hasil Pengujian Fungsional 119](#_Toc536849711)

[7.3.2 Hasil Pengujian Acceptance 122](#_Toc536849712)

[7.4 Evaluasi Pengujian Acceptance 123](#_Toc536849713)

[8 Kesimpulan dan Saran 124](#_Toc536849714)

[8.1 Kesimpulan 124](#_Toc536849715)

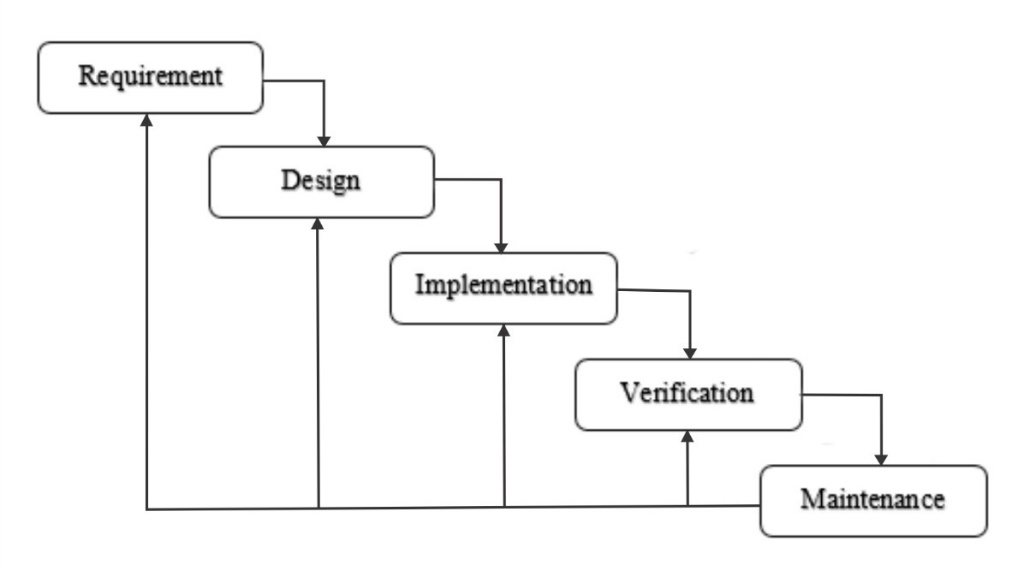
[8.2 Saran 125](#_Toc536849716)

[DAFTAR PUSTAKA 126](#_Toc536849717)

[LAMPIRAN 127](#_Toc536849718)

### Metode Pembangunan Perangkat Lunak

Pada pembangunan perangkat lunak ini kami menggunakan Metode Waterfall atau sering disebut metode air terjun, dimana hal ini menggambarkan pendekatan yang sistematis dan juga berurutan pada pengembangan perangkat lunak, dimulai dengan spesifikasi kebutuhan pengguna lalu berlanjut melalui tahapan-tahapan perencanaan (planning), permodelan (modeling), konstruksi (construction), serta penyerahan sistem ke para pelanggan/pengguna (deployment), yang diakhiri dengan dukungan pada perangkat lunak lengkap yang dihasilkan (Pressman, 2012). Tahapan metode waterfall dapat dilihat pada gambar di bawah ini.



Berikut ini tahapan yang sudah kami lakukan, di antaranya:

**Tahapan Pertama adalah Validasi Ide :**

1. Wawancara
2. Analisa masalah
3. Mengumpulkan informasi data.

**Tahapan Kedua adalah Waterfall :**

1. Requirement Analisis

Tahap ini pengembang sistem diperlukan komunikasi yang bertujuan untuk memahami perangkat lunak yang diharapkan oleh pengguna dan batasan perangkat lunak tersebut. Informasi ini biasanya dapat diperoleh melalui wawancara, diskusi atau survei langsung. Informasi dianalisis untuk mendapatkan data yang dibutuhkan oleh pengguna.

2. System Design

Spesifikasi kebutuhan dari tahap sebelumnya akan dipelajari dalam fase ini dan desain sistem disiapkan. Desain Sistem membantu dalam menentukan perangkat keras(hardware) dan sistem persyaratan dan juga membantu dalam mendefinisikan arsitektur sistem secara keseluruhan.

3. Implementation

Pada tahap ini, sistem pertama kali dikembangkan di program kecil yang disebut unit, yang terintegrasi dalam tahap selanjutnya. Setiap unit dikembangkan dan diuji untuk fungsionalitas yang disebut sebagai unit testing.

4. Integration & Testing

Seluruh unit yang dikembangkan dalam tahap implementasi diintegrasikan ke dalam sistem setelah pengujian yang dilakukan masing-masing unit. Setelah integrasi seluruh sistem diuji untuk mengecek setiap kegagalan maupun kesalahan.

5. Operation & Maintenance

Tahap akhir dalam model waterfall. Perangkat lunak yang sudah jadi, dijalankan serta dilakukan pemeliharaan. Pemeliharaan termasuk dalam memperbaiki kesalahan yang tidak ditemukan pada langkah sebelumnya. Perbaikan implementasi unit sistem dan peningkatan jasa sistem sebagai kebutuhan baru.

## OOAD dengan UML

OOAD adalah metode analisis yang memerikasa requirements dari sudut pandang kelas kelas dan objek yang ditemui dalam ruang lingkup permasalahan yang mengarahkan arsitektur software yang didasarkan pada manipulasi objek-objek system atau subsistem[11]. Sekitar akhir tahun 80-an dan awal tahun 90-an, UML (Unified Modeling Language) menjadi pengganti dari metode analisis berorientasi object dan design berorientasi object(OOAD&D/object oriented analysis and design)[13].

### UML

Menurut Booch (2005:7) UML adalah Bahasa standar untuk membuat rancangan software. UML biasanya digunakan untuk menggambarkan dan membangun, dokumen artifak dari software –intensive system [3].

Menurut Nugroho (2010:6), UML (Unified Modeling Language) adalah ‘bahasa’ pemodelan untuk sistem atau perangkat lunak yang berparadigma ‘berorientasi objek”. Pemodelan (modeling) sesungguhnya digunakan untuk penyederhanaan permasalahan-permasalahan yang kompleks sedemikian rupa sehingga lebih mudah dipelajari dan dipahami [3].

Menurut Nugroho (2009:4), UML (Unified Modeling Language) adalah Metodologi kolaborasi antara metoda-metoda Booch, OMT (Object Modeling Technique), serta OOSE (Object Oriented Software Enggineering) dan beberapa metoda lainnya, merupakan metodologi yang paling sering digunakan saat ini untuk analisa dan perancangan sistem dengan metodologi berorientasi objek mengadaptasi maraknya penggunaan bahasa “pemrograman berorientasi objek” (OOP) [3].

Menurut Herlawati (2011:10), bahwa beberapa literature menyebutkan bahwa UML menyediakan sembilan jenis diagram, yang lain menyebutkan delapan karena ada beberapa diagram yang digabung, misanya diagram komunikasi, diagram urutan dan diagram pewaktuan digabung menjadi diagram interaksi [3].



Gambar 4 Unified Modeling Language

### Usecase

Diagram Use Case atau *Use Case Diagram* adalah pemodelan untuk menggambarkan *behavior* / kelakuan sistem yang akan dibuat [2]. Selain itu, diagram use case juga digunakan untuk memahami fungsi apa saja yang ada di dalam sebuah sistem dan siapa saja yang dapat menggunakan fungsi-fungsi tersebut.



Gambar 5 Notasi Diagram Usecase

### Usecase scenario

Skenario use case / *use case skenario* adalah alur jalannya proses use case dari sisi aktor dan system [2]. Alur skenario inilah yang nantinya menjadi landasan pembuatan sequence diagram / diagram sekuen.



Gambar 6 Usecase Skenario

### Activity diagram

*Activity diagrams* adalah diagram yang menggambarkan berbagai aliran aktivitas dalam sebuah sistem yang sedang dirancang dan bagaimana masing-masing aliran berawal, keputusan yang mungkin terjadi, dan bagaimana aktivitas tersebut berakhir [3]. Diagram aktivitas ini memiliki beberapa komponen dengan bentuk tertentu yang dihubungan dengan tanda panah. Dimana tanda panah tersebut mengarahkan urutan aktivitas yang terjadi dari awal sampai akhir



Gambar 7 Notasi Activity Diagram

### Class diagram

*Class diagram* adalah diagram yang menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem [4]. Kelas ini memiliki 3 bagian utama yaitu *attribute, operation*, dan *name*. Kelas-kelas yang ada pada struktur sistem harus dapat melakukan fungsi-fungsi sesuai dengan kebutuhan system.



Gambar 8 Bagian-bagian Class Diagram



Gambar 9 Notasi Relasi Class Diagram

### Sequence diagram

*Sequence diagram* ini adalah diagram yang menggambarkan kolaborasi dinamis antara sejumlah object [5]. *Sequence diagram* juga digunakan untuk menunjukkan rangkaian pesan yang dikirim antara object juga interaksi antara object.

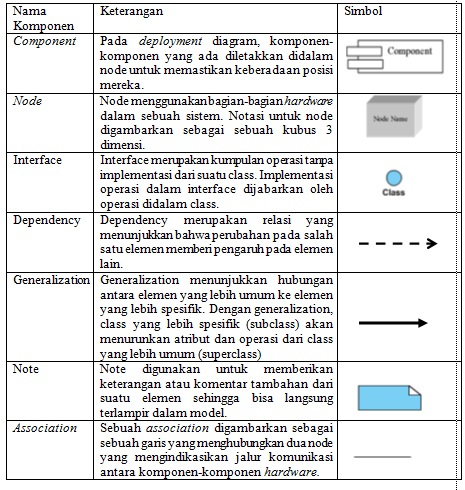


Gambar 10 Notasi Sequence Diagram

### Deployment Diagram

**Deployment diagram** adalah salah satu model diagram dalam UML untuk mengerahkan artifact dalam node. Deployment diagram digunakan untuk memvisualisasikan, menspesifikasikan, dan mendokumentasikan proses yang terjadi pada suatu sistem perangkat lunak berbasis Object Oriented yang akan dibangun. Tujuan atau fungsi dari deployment diagram yaitu untuk menggambarkan/memvisualisasikan secara umum proses yang terjadi pada suatu sistem/software.

Tabel 3 Deployment Program



## Mobile Development (Native Android)

Aplikasi Native adalah aplikasi yang di bangun dengan bahasa pemrograman yg spesifik tertentu platform tertentu, dan di pembangunan perangkat lunak ini berbasis android maka bahasa yg akan digunakan yaitu java.

Java adalah bahasa pemrograman tingkat tinggi yang dikembangkan oleh sun microsystems dan dirilis pertama kali pada tahun 1995 [6]. Pemrograman java ini dapat berjalan di berbagai platform seperti Windows, Mac dan bermacam-macam versi UNIX. Pemrograman java juga semakin banyak peminatnya karena dengan menguasai pemrograman java, berarti akan mudah untuk membuat sebuah aplikasi android.

## Business Model Canvas

Model Bisnis Kanvas adalah merupakan salah satu strategi bisnis yang dapat mendesain, menggambarkan hingga menyimpulkan aspek – aspek bisnis menjadi satu strategi yang utuh. Model Bisnis Kanvas mencakup 9 area atau 9 blok yang akan membantu seseorang dalam merumuskan ide bisnis mereka menjadi suatu model bisnis yang real. [10]



Gambar 12 Bisnis Model Kanvas

## Pengujian Black Box

Metode pengujian secara *black box* menekankan pengujian pada fungsionalitas yang ada dari setiap bagian di dalam sistem yang dibuat tanpa mengetahui bait program yang ada. Pengujian ini dilakukan setelah bait program yang ada selesai untuk dibuat. Tujuan dari pengujian ini yaitu untuk memastikan setiap bagian sudah sesuai dengan alur proses yang ditetapkan dan memastikan semua kesalahan masukan yang dilakukan oleh pengguna dapat ditangani oleh sistem. Pengujian ini akan dilakukan dengan cara yang relatif bertentangan dengan kebutuhan yang ada, tujuannya di sini yaitu untuk memastikan sistem dapat menangani semua masukan yang tidak sesuai dan memastikan pengguna hanya bisa memasukkan data yang benar ke sistem.[14]

## Pengujian Acceptance

Acceptance Testing adalah Pengujian akhir yang berhubungan dengan kebutuhan user aplikasi/sistem, persayaratan dan proses bisnis yang dilakukan untuk menentukan apakah suatu sistem memenuhi kriteria desain awal dan untuk meyakinkan user/pelanggan aplikasi tersebut menentukan apakah sistem dapat diterima dengan baik atau tidak.[15]



Gambar 13 Pengujian Acceptance

# KEBUTUHAN PERANGKAT LUNAK

Dalam pembuatan perangkat lunak pastinya diperlukan kebutuhan yang akan digunakan dala pembangunan perangkat lunak itu sendiri dan disini akan dibahas kebutuhan-kebutuhan yang akan digunakan dalam pembangunan perangkat lunak.

## Analisis Sistem Berjalan

1. Atasan memberi tahu karyawan untuk rapat secara manual;
2. Menggunakan sistem metting arranger lewat SMS gateway untuk memberi tahu rapat ke no masing-masing pegawai;



Gambar 14 Activity Diagram Sistem berjalan

## Analisis Arsitektur Sistem

Berikut ini merupakan gambar analisis arsitektur sitem aplikasi Kumato



Gambar 15 Arsitektur Sistem

Gambaran arsitektur sistem yang akan dibangun hanya memiliki satu buah sistem. Penerima surat dapat menginputkan .

## Spesifikasi Kebutuhan Perangkat Lunak

Spesifikasi kebutuhan perangkat lunak dibagi menjadi 2 kategori kebutuhan, yaitu kebutuhan fungsional dan kebutuhan non fungsional. Berikut ini kebutuhan fungsional dan kebutuhan non fungsional pada Aplikasi Kumato :

### Kebutuhan Fungsional

Berikut ini merupakan tabel kebutuhan fungsional

Tabel 4 Kebutuhan Fungsional

|  |  |
| --- | --- |
| SRS-F-001 | Sistem menyediakan fasilitas bagi pengguna untuk menambahkan list kegiatan |
| SRS-F-002 | Sistem menyediakan fasilitas bagi pengguna untuk mengatur waktu pengingat kegiatan |
| SRS-F-003 | Sistem menyediakan fasilitas bagi pengguna untuk menggunakan teknik pomodoro dengan otomatis |

### Kebutuhan Non Fungsional

Berikut ini merupakan tabel kebutuhan nonfungsional

Tabel 5 Kebutuhan Nonfungsional

|  |  |
| --- | --- |
| SRS-NF-001 | Sistem yang akan dibangun berbasis Mobile Android |
| SRS-NF-002 | Sistem yang akan dibangun akan mulus dijalankan minimal pada sistem operasi android Lolipop |
| SRS-NF-003 | Sistem yang akan dibangun memerlukan media penyimpanan minimal 30 MB untuk dapat menginstallnya |
| SRS-NF-004 | Sistem yang akan dibangun menggunakan bahasa pemrograman Java, Xml, dan Firebase |
| SRS-NF-005 | Sistem yang akan dibangun menggunakan DBMS SQLite |
| SRS-NF-006 | Sistem yang dibangun dengan spesifikasi hardware yang memenuhi standar minimum kebutuhan |
| SRS-NF-007 | Sistem yang akan dibangun akan memberikan notifikasi dalam bentuk suara dan getar |
| SRS-NF-008 | Pengguna yang akan menggunkan sistem ataupun aplikasi ini minimal mengerti mengoperasikan smartphone |
| SRS-NF-009 | Pengguna yang membutuhkan pengingat akan deadline pekerjaan dan tugasnya |
| SRS-NF-010 | Sistem yang akan dibangun hanya memerlukan koneksi dengan database |

## Analisis Kebutuhan Nonfungsional

Kebutuhan nonfungsional dibagi menjadi beberapa bagian, yaitu kebutuhan perangkat lunak, kebutuhan perangkat keras, kebutuhan perangkat pikir, dan kebutuhan jaringan. Berikut ini analisis kebutuhan nonfunsional pada Aplikasi Kumato :

### Analisis Kebutuhan Perangkat Lunak

**Spesifikasi Kebutuhan NonFungsional Perangkat Lunak:**

Berikut ini merupakan tabel kebutuhan nonfungsional perangkat lunak

Tabel 6 kebutuhan Nonfungsional Perangkat lunak

|  |  |
| --- | --- |
| SRS-NF-001 | Sistem yang akan di bangun berbasis Mobile Android |
| SRS-NF-002 | Sistem yang akan di bangun akan mulus dijalankan minimal pada sistem operasi android Lolipop |
| SRS-NF-003 | Sistem yang akan di bangun memerlukan media penyimpanan minimal 30 MB untuk dapat menginstallnya |
| SRS-NF-004 | Sistem yang akan dibangun menggunakan bahasa pemrograman Java, Xml, dan Firebase |
| SRS-NF-005 | Sistem yang akan dibangun menggunakan DBMS SQLite |

**Fakta perangkat Lunak (Yang ada dilingkungan sistem yang berjalan saat ini) :**

Sistem operasi : Android Lolipop.

Dataabase : SQLite

**Kebutuhan Pembangunan Perangkat Lunak :**

Dari sisi *developer*, perangkat lunak yang digunakan dalam pembangunan perangkat lunak ini adalah:

1. Platform : Mobile Android;
2. Sistem operasi : Android Lolipop;
3. Memory : Minimal 30 MB.
4. Bahasa pemrograman : Java, xml dan firebase;
5. DBMS : SQLite;

**Kesimpulan (Hasil Perbandingan fakta dengan kebutuhan) :**

Berdasarkan SRS-NF-001, SRS-NF-002 dan SRS-NF-003 antara kebutuhan perangkat lunak dan fakta yang ada di lingkungan sistem sudah memenuhi syarat kebutuhan.

### Analisis Kebutuhan Perangkat Keras

**Spesifikasi Kebutuhan NonFungsional Perangkat Keras:**

Tabel 7 kebutuhan nonfungsional Perangkat keras

|  |  |
| --- | --- |
| SRS-NF-006 | Sistem yang dibangun dengan spesifikasi hardware yang memenuhi standar minimum kebutuhan |
| SRS-NF-007 | Sistem yang akan di bangun akan memberikan notifikasi dalam bentuk suara dan getar |

**Fakta perangkat Keras (Yang ada dilingkungan sistem yang berjalan saat ini) :**

Procesor : Snapdragon 435 1.0 GHz

Ram : 1 GB

Memory : 8 GB

Layar : 4.0’’

**Kebutuhan Perangkat Keras :**

Platform : Mobile Android

Procesor : ARM Cortex-A7 1.0 GHz

Ram : 512 Mb

Memory : 4GB

Layar : 3.5’’

**Kesimpulan (Hasil Perbandingan fakta dengan kebutuhan) :**

Berdasarkan SKPL-NF-004 dan SKPL-NF-005 antara fakta dengan kebutuhan sistem sudah memenuhi syarat minimal.

### Analisis Kebutuhan Perangkat Pikir

Analisis ke butuhan Perangkat pikir yang dilakukan pada penelitian ini hanya mengenai costumer segment.

Tabel 8 Kebutuhan perangkat pikir

|  |  |  |  |
| --- | --- | --- | --- |
| **Costumer** | **Umur** | **Behavior** | **Psycologis** |
| Mahasiswa | 19 – 25 Tahun | Butuh adanya pengingat, Biasa menggunakan Smartphone, membutuhkan fokus dan tetap santai. | Kondisinya Normal. |

**Kebutuhan Nonfungsional**

Berikut ini merupakan tabel kebutuhan Nonfungsional perangkat pikir

Tabel 9 Kebutuhan Nonfungsional

|  |  |
| --- | --- |
| SRS-NF-008 | Pengguna yang akan menggunkan sistem ataupun aplikasi ini minimal mengerti mengoperasikan smartphone |
| SRS-NF-009 | Pengguna yang membutuhkan pengingat akan deadline pekerjaan dan tugasnya |

**Fakta Perangkat Pikir**

Berikut ini merupakan tabel fakta perangkat pikir

Tabel 10 Fakta Perangkat Pikir

|  |  |  |  |
| --- | --- | --- | --- |
| **Costumer** | **Umur** | **Behavior** | **Pengalaman menggunakana smartphone** |
| Mahasiswa | 21 Tahun | Sering lupa dalam mengerjakan tugas, butuh di ingatkan | Expert |

**Kesimpulan (Hasil perbandingan Fakta dan kebutuhan)**

Berdasarkan SRS-NF-006 dan SRS-NF-007 fakta yang ada di lapangan sesuai dengan kebutuhan yang diperlukan oleh sistem.

### Analisis Kebutuhan Jaringan

**Kebutuhan Nonfongsional jaringan:**

Berikut ini merupakan tabel kebutuhan nonfungsional jaringan :

Tabel 11 Kebutuhan nonfungsional jaringan

|  |  |
| --- | --- |
| SRS-NF-010 | Sistem yang akan di bangun hanya memerlukan koneksi dengan database |

**Fakta Jaringan (Yang ada di lingkungan sistem):**

Menggunakan SQlite yang berfungsi untuk menampung data yang di inputkan, data yang berhasil di inputkan akan di tampilkan di halaman utama aplikasi dalam bentuk list data.

**Kesimpulan (Hasil Perbandingan fakta dengan kebutuhan) :**

Berdasarkan SKPL-NF-008 antara fakta dengan kebutuhan sistem sudah memenuhi syarat minimal.

## Analisis Kebutuhan Data

Analisis data yang menjadi acuan dalam membangun perangkat lunak ini adalah dokumen-dokumen yang berhubungan dengan donasi serta objek-objek lain yang terkait. Berikut adalah analisis data dari pembangunan perangkat lunak ini:

Tabel 12 Kebutuhan Data

| **ENTITAS** | **ATRIBUT** |
| --- | --- |
| Kegiatan | = {deskripsi\_kegiatan, tanggal\_waktu} |

Dari entitas tersebut, asumsi dalam penelitian ini adalah:

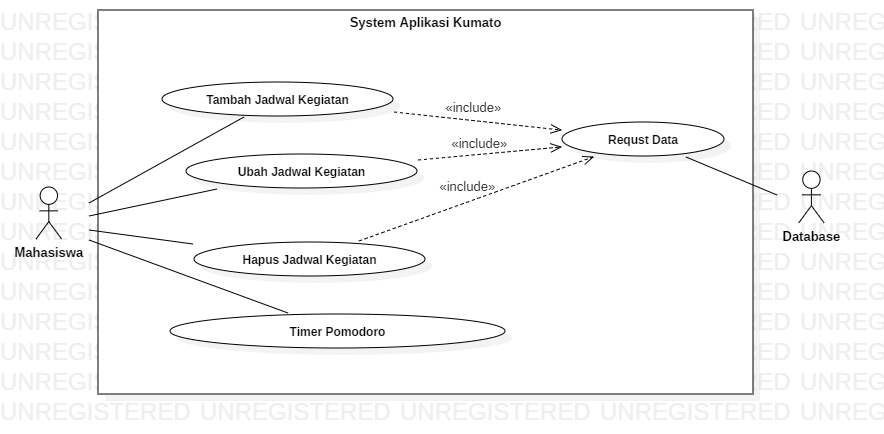
1. Satu kegiatan hanya dapat menggunakan 1 waktu pengingat.

Konsep analisis data yang digunakan adalah dengan menggunakan konsep ORM (*Object Relational Mapping*) dimana ORM ini akan memetakan sebuah objek dengan *database*. Dengan kata lain, ORM ini akan membantu menjadi jembatan antara objek yang didefinisikan dalam kode program dengan *database*. Objek-objek yang akan terlibat dalam ORM ini adalah objek mahasiswa.

## Model Use Case

Pada bagian model *usecase* ini akan dipaparkan mengenai gambar diagram *usecase* keseluruhan, definisi aktor, definisi *usecase* dan skenario *usecase*.

### Diagram Use Case



Gambar 16 Usecase Diagram

### Definisi Actor

Berikut ini merupakan tabel definisi Actor

Tabel 13 Definisi aktor

| **NO** | **AKTOR** | **DESKRIPSI** |
| --- | --- | --- |
| *1* | *User (Mahasiswa)* | *Aktor dengan role ini mempunyai wewenang untuk melakukan penambahan list kegiatan, ubah list kegiatan dan hapus list kegiatan serta mengatur waktu dan tanggal reminder nya. Dan juga aktor ini dapat menggunakan sebuah fitur yang ada pada aplikasi yaitu fitur Teknik Pomodoro.* |
| *2* | *Database* | *Aktor dengan role ini mempunyai wewanang untuk memberikan atau menyimpan data yang direquest atau ditambahkan pada Aplikasi Kumato.* |

### Definisi Use Case

Berikut ini merupakan tabel definisi use case

Tabel 14 Definisi usecase

| **NO** | **USE CASE** | **DESKRIPSI** |
| --- | --- | --- |
| *1* | Tambah Kegiatan | *User menambah sebuah kegiatan baru dan sistem menampilkan daftar list kegiatan* |
| *2* | Ubah Jadwal Kegiatan | *User mengubah jadwal kegiatan dan sistem menampilkan form ubah list kegiatan* |
| *3* | Hapus Jadwal Kegiatan | *User menghapus jadwal kegiatan dan sistem menampilkan form ubah list kegiatan* |
| *4* | Timer Podomoro | *Sistem menjalan timer teknik pomodoro* |
| *5* | Request data | *Sistem menampilkan daftar list kegiatan* |
| *6* | Notifikasi Penggunaan Teknik Pomodoro | *Sistem menampilkan notifikasi bahwa kegiatan telah selesai dikerjakan.* |

### Skenario Use Case

Berikut ini merupakan beberapa tabel skenario use case :

Tabel 15 Usecase Scenario Tambah Data

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Tambah Kegiatan | |
| **Related Requirement** | - | |
| Goal In Context | Menambahkan data kegiatan ke dalam aplikasi | |
| Description | Fungsionalitas ini digunakan oleh user untuk menambahkan data kegiatan ke dalam database aplikasi | |
| Related Use Case | Request data kegiatan | |
| Preconditions | - | |
| Successful End Condition | Aktor berhasil menambahkan data kegiatan | |
| Failed End Condition | Aktor tidak menambahkan data kegiatan | |
| Primary Actors | User (Mahasiswa) | |
| Secondary Actors | - | |
| Trigger | User memilih/menekan tombol berwarna merah dan berbentuk “+” | |
| Main Flow | Step | Action |
|  | 1. | User memilih tombol berwarna merah dan berbentuk “+” |
|  | 2. | Sistem menampilkan halaman penginputan data kegiatan |
|  | 3. | User mengisi data kegiatan berupa nama kegiatan beserta waktu remindernya |
|  | 4.  include:: CRUD data kegiatan | Sistem menyimpan data kegiatan baru ke dalam database |
|  | 5. | Sistem menampilkan halaman home yang berisi list data kegiatan terbaru |
| Extension | Step | Branching Action |
|  | 4.1 | Sistem gagal menyimpan data kegiatan |
|  | 5.1 | Sistem gagal menampilkan halaman home yang berisi list data kegiatan terbaru |

Tabel 16 Usecase Scenario Ubah Kegiatan

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Ubah Kegiatan | |
| **Related Requirement** | - | |
| Goal In Context | Mengubah data kegiatan yang sudah tersimpan | |
| Description | Fungsionalitas ini digunakan oleh user untuk mengubah data kegiatan dalam aplikasi | |
| Related Use Case | Request data kegiatan | |
| Preconditions | - | |
| Successful End Condition | Aktor berhasil mengubah data kegiatan | |
| Failed End Condition | Aktor tidak mengubah data kegiatan | |
| Primary Actors | User (Mahasiswa) | |
| Secondary Actors | - | |
| Trigger | User memilih/menekan kegiatan yang akan diubah | |
| Main Flow | Step | Action |
|  | 1. | User memilih/menekan kegiatan yang akan diubah |
|  | 2.  Include:: Request data kegiatan | Sistem melakukan permintaan detail data kegiatan ke database |
|  | 3. | Sistem menampilkan halaman detail data kegiatan |
|  | 4. | User mengubah detail data kegiatan diantaranya nama kegiatan atau waktu remindernya |
|  | 5.  Include:: CRUD data kegiatam | Sistem akan menyimpan data kegiatan yang sudah diubah ke database |
|  | 6. | Sistem menampilkan halaman home yang berisi list data kegiatan terbaru |
| Extension | Step | Branching Action |
|  | 2.1 | Sistem gagal mengambil detail data kegiatan |
|  | 5.1 | Sistem gagal menyimpan detail data kegiatan |
|  | 6.1 | Sistem gagal menampilkan halaman home yang berisi list data kegiatan terbaru |

Tabel 17 Usecase Scenario Hapus Jadwal Kegiatan

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Hapus Kegiatan | |
| **Related Requirement** | - | |
| Goal In Context | Menghapus data kegiatan yang sudah tersimpan | |
| Description | Fungsionalitas ini digunakan oleh user untuk menghapus data kegiatan dalam aplikasi | |
| Related Use Case | Request data kegiatan | |
|  | - | |
| Successful End Condition | Aktor berhasil menghapus data kegiatan | |
| Failed End Condition | Aktor tidak menghapus data kegiatan | |
| Primary Actors | User (Mahasiswa) | |
| Secondary Actors | - | |
| Trigger | User memilih/menekan tombol berbentuk tong sampah | |
| Main Flow | Step | Action |
|  | 1. | User memilih/menekan kegiatan yang akan dihapus |
|  | 2.  Include:: Request data kegiatan | Sistem melakukan permintaan detail data kegiatan ke database |
|  | 3. | Sistem menampilkan halaman detail data kegiatan |
|  | 4. | User memilih/menekan tombol berbentuk tong sampah yang berada di pojok kanan atas |
|  | 5.  Include:: CRUD data kegiatam | Sistem akan menghapus data kegiatan yang dipilih pada database |
|  | 6. | Sistem menampilkan halaman home yang berisi list data kegiatan terbaru |
| Extension | Step | Branching Action |
|  | 2.1 | Sistem gagal mengambil detail data kegiatan |
|  | **4.1** | **Sistem menampilkan validasi penghapusan** |
|  | 5.1 | Sistem gagal menyimpan detail data kegiatan |
|  | 6.1 | Sistem gagal menampilkan halaman home yang berisi list data kegiatan terbaru |

Tabel 18 Usecase Scenario Penggunaan Teknik Pomodoro

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Penggunaan Teknik Pomodoro | |
| **Related Requirement** | - | |
| Goal In Context | Menerapkan teknik pomodoro pada aplikasi | |
| Description | Fungsionalitas ini digunakan oleh user untuk membantu meningkatkan kefokusan dan produktivitas user melalui aplikasi | |
| Related Use Case | - | |
| Preconditions | - | |
| Successful End Condition | Aktor berhasil menggunakan teknik pomodoro dengan baik | |
| Failed End Condition | Aktor tidak menggunakan teknik pomodoro dengan baik | |
| Primary Actors | User (Mahasiswa) | |
| Secondary Actors | - | |
| Trigger | Aktor memilih/menekan tombol Mulai | |
| Main Flow | Step | Action |
|  | 1. | User memilih/menekan tombol Mulai |
|  | 2. | Timer melakukan waktu mundur selama 25 menit |
|  | 3. | Sistem menampilkan notifikasi untuk beristirahat ketika waktu habis |
|  | 4. | User menekan notifikasi beristirahat |
|  | 5. | Sistem mengubah waktu pada timer menjadi 5 menit |
|  | 6. | User memilih/menekan tombol Mulai |
|  | 7. | Timer melakukan waktu mundur selama 5 menit |
|  | 8. | Sistem menampilkan notifikasi waktu beristirahat sudah habis |
|  | 9. | User menekan notifikasi waktu beristirahat sudah habis |
|  | 10. | Sistem mengembalikan lagi waktu pada timer menjadi 25 menit |
|  | 11. | User mengulangi langkah 1 hingga 10 sampai target tercapai |
| Extension | Step | Branching Action |
|  | 2.1 | Timer tidak berjalan |
|  | 3.1 | Sistem gagal menampilkan notifikasi |
|  | 5.1 | Sistem gagal mengubah waktu pada timer |
|  | 7.1 | Timer tidak berjalan |
|  | 8.1 | Sistem gagal menampilkan notifikasi |
|  | 10.1 | Sistem gagal mengembalikan waktu pada timer |

Tabel 19 Usecase Scenario Timer Teknik Pomodoro

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Penyediaan Informasi Teknik Pomodoro | |
| **Related Requirement** | - | |
| Goal In Context | Memperkenalkan Teknik Pomodoro | |
| Description | Menyampikan informasi tentang Teknik Pomodoro | |
| Related Use Case | - | |
|  | - | |
| Successful End Condition | Pengguna akan mengerti dan mengetahui Teknik Pomodoro | |
| Failed End Condition | Tidak membaca informasi tentang Teknik Pomodoro | |
| Primary Actors | Mahasiswa | |
| Secondary Actors | - | |
| Trigger | Ketika sudah selesai membaca informasi tentang teknik pomodoro makan akan langsung menuju menu utama aplikasi. | |
| Main Flow | Step | Action |
|  | 1. | Buka Aplikasi |
|  | 2. | Tampil informasi mengenai Teknik Pomodoro |
|  | 3. | Silahkan dibaca, dan swip ke kanan untuk menuju halaman selanjutnya |
| Extension | Step | Branching Action |
|  |  | - |
|  |  |  |

## Model Activity Diagram

Berikut ini merupakan model activity diagram

* **Model Activity Diagram Tambah Data List Kegiatan**

****

Gambar 17 Tambah Data Kegiatan

* **Model Activity Diagram Hapus List Kegiatan**

****

Gambar 18 Hapus Data Kegiatan

* **Model Activity Diagram Ubah List Kegiatan**

****

Gambar 19 Hapus Data Kegiatan

## Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional dalam penelitian ini menggunakan pendekatan OOAD (*Object Oriented Analysis and Design*). Pemaparan lebih detail akan dipaparkan pada BAB 4.

# MODEL ANALISIS

## Diagram Class Analisis

Berikut ini adalah diagram kelas analisis yang terdapat pada Aplikasi Kumato :



Gambar 20 Class Diagram Aplikasi Kumato

## Identifikasi Kelas Analisis

Berikut ini adalah daftar seluruh kelas-kelas analisis dalam tabel berikut:

Tabel 20 Kelas Analisis

|  |  |  |
| --- | --- | --- |
| Nama Kelas | Daftar Tanggung-Jawab (Method) | Daftar Atribut |
| To Do List | inputListKegiatan() mengeditListKegiatan()  menghapusListKegiatan()  menampilkanListKegiatan() | JudulKegiatan  DeskripsiKegiatan  Tanggal  Waktu |
| Pomodor Timer | memulaiTimer()  mempauseTimer()  meresetTimer() |  |

## Realisasi Kelas Terhadap Use Case

Bagian ini diisi dengan bentuk realisasi hubungan kelas analisis terhadap Use Case

Berikut ini merupakan tabel realisasi kelas terhadap use case

Tabel 21 Realisasi Kelas Terhadap Use Case

|  |  |  |
| --- | --- | --- |
| No | Use Case | Realisasi Kelas |
| 1 | Tambah Jadwal Kegiatan | Ubah jadwal kegiatan, Hapus jadwal kegiatan, notifikasi kegiatan |
| 2 | Ubah Jadwal Kegiatan | Hapus jadwal kegiatan, notifikasi kegiatan |
| 3 | Hapus Jadwal Kegiatan | Notifikasi kegiatan |
| 4 | Penyediaan Informasi Teknik Pomodoro | Memakai fitur pomodoro |
| 5 | Meningkatkan kefokusan memakai fitur pomodoro | Penyediaan informasi teknik pomodoro, notifikasi kegiatan |
| 6 | Notifikasi Kegiatan | Tambah jadwal kegiatan, ubah jadwal kegiatan, hapus jadwal kegiatan, memakai fitur pomodoro |

### Identifikasi Kelas Analisis tiap Paket

Berikut ini merupakan tabel identifikasi kelas analisis tiap paket

Tabel 22 Identifikasi Kelas Analisis tiap Paket

|  |  |  |  |
| --- | --- | --- | --- |
| No | Nama Paket | Nama Kelas Analisis | Jenis Kelas |
| 1 | adapter | NoteAdapter | Control |
| 2 | db | DatabaseContract DatavaseHelper NoteHelper |  |
| 3 | entity | Note | Entity |
| 4 |  |  |  |

# Model Perancangan

Berikut ini merupakan macam – macam model perancangan :

## Realisasi Use Case Tahap Perancangan

Bagian ini mendetailkan realisasi model perancangan terhadap Use Case dari perangkat lunak yang dibangun.

### Use Case Tambah Kegiatan

Berikut merupakan realisasi usecae <<>> dalam identifikasi kelas diagram

#### Identifikasi Kelas Perancangan

Berikut adalah daftar kelas analaisis :

|  |  |  |
| --- | --- | --- |
| No | Nama Kelas Perancangan | Nama Kelas Analisis Terkait |
| 1 | NoteAdapater | To Do list |
| 2 | DatabaseContract | To Do list |
| 3 | NoteHelper | To Do list |
| 4 | Note | To Do list |
| 5 | CustomOnItemClickListener | To Do list |
| 6 | FormAddUpdateActivity | To Do list |
| 7 | MainActivity | To Do list |

#### Sequence Diagram

1. **Sequence Diagram DatabaseHelper**



1. **Sequence Diagram FormAddUpdate**













1. **Sequence Diagram MainActivity**











1. **Sequence Diagram Note**



1. **Sequence Diagram NoteAdapter**







1. **Sequence Diagram NoteHelper**





1. **Sequence Diagram OnItemClickListener**



1. Sequence Diagram PomodoroActivity









### Use Case Ubah Jadwal Kegiatan

Berikut merupakan realisasi usecae <<>> dalam identifikasi kelas diagram

#### Identifikasi Kelas Perancangan

|  |  |  |
| --- | --- | --- |
| No | Nama Kelas Perancangan | Nama Kelas Analisis Terkait |
| 1 | NoteAdapater | To Do list |
| 2 | DatabaseContract | To Do list |
| 3 | NoteHelper | To Do list |
|  | Note | To Do list |
|  | CustomOnItemClickListener | To Do list |
|  | FormAddUpdateActivity | To Do list |
|  | MainActivity | To Do list |

#### Sequence Diagram

1. **Sequence Diagram DatabaseHelper**



1. **Sequence Diagram FormAddUpdate**













1. **Sequence Diagram MainActivity**











1. **Sequence Diagram Note**



1. **Sequence Diagram NoteAdapter**







1. **Sequence Diagram NoteHelper**





1. **Sequence Diagram OnItemClickListener**



1. Sequence Diagram PomodoroActivity









### Use Case Hapus Jadwal Kegiatan

Berikut merupakan realisasi usecae <<>> dalam identifikasi kelas diagram

#### Identifikasi Kelas Perancangan

|  |  |  |
| --- | --- | --- |
| No | Nama Kelas Perancangan | Nama Kelas Analisis Terkait |
| 1 | NoteAdapater | To Do list |
| 2 | DatabaseContract | To Do list |
| 3 | NoteHelper | To Do list |
|  | Note | To Do list |
|  | CustomOnItemClickListener | To Do list |
|  | FormAddUpdateActivity | To Do list |
|  | MainActivity | To Do list |

#### Sequence Diagram

1. **Sequence Diagram DatabaseHelper**



1. **Sequence Diagram FormAddUpdate**













1. **Sequence Diagram MainActivity**











1. **Sequence Diagram Note**



1. **Sequence Diagram NoteAdapter**







1. **Sequence Diagram NoteHelper**





1. **Sequence Diagram OnItemClickListener**



1. Sequence Diagram PomodoroActivity









### Use Case Timer Pomodoro

Berikut merupakan realisasi usecae <<>> dalam identifikasi kelas diagram

#### Identifikasi Kelas Perancangan

|  |  |  |
| --- | --- | --- |
| No | Nama Kelas Perancangan | Nama Kelas Analisis Terkait |
| 1 | PomodoroActivity | Pomodoro |

#### Sequence Diagram

1. Sequence Diagram PomodoroActivity









### Use Case Request Data

Berikut merupakan realisasi usecae <<>> dalam identifikasi kelas diagram

#### Identifikasi Kelas Perancangan

|  |  |  |
| --- | --- | --- |
| No | Nama Kelas Perancangan | Nama Kelas Analisis Terkait |
| 1 | NoteAdapater | To Do list |
| 2 | DatabaseContract | To Do list |
| 3 | NoteHelper | To Do list |
|  | Note | To Do list |
|  | CustomOnItemClickListener | To Do list |
|  | FormAddUpdateActivity | To Do list |
|  | MainActivity | To Do list |

#### Sequence Diagram

1. **Sequence Diagram DatabaseHelper**



1. **Sequence Diagram FormAddUpdate**













1. **Sequence Diagram MainActivity**











1. **Sequence Diagram Note**



1. **Sequence Diagram NoteAdapter**







1. **Sequence Diagram NoteHelper**





1. **Sequence Diagram OnItemClickListener**



1. Sequence Diagram PomodoroActivity









### Use Case Notifikasi Timer Pomodoro

Berikut merupakan realisasi usecae <<>> dalam identifikasi kelas diagram

#### Identifikasi Kelas Perancangan

|  |  |  |
| --- | --- | --- |
| No | Nama Kelas Perancangan | Nama Kelas Analisis Terkait |
| 1 | PomodoroActivity | Pomodoro |

#### Sequence Diagram

Sequence Diagram PomodoroActivity









## Perancangan Detil Elemen Logical View

### Kelas NoteAdapter

|  |  |  |
| --- | --- | --- |
| **Nama Operasi** | **Visibility (private, public)** | **Keterangan** |
| onCreateViewHolder | **p**ublic | Dipanggil ketiga activity akan diload |
| onBindViewHolder | Public | Memanggil data |
| getItemCount | public | Mendapatkan jumlah item |
| **Nama Atribut** | **Visibility (private, public)** | **Tipe** |
| listNotes | private | ArrayList |
| activity | **p**rivate | Activity |
| tvTitle | Package | string |
| tvTime | Package | String |
| tvDescription | Package | String |
| cvNote | package | String |
| tv\_item\_title | Package | String |
| tv\_item\_description | Package | String |
| tv\_item\_time | Package | String |
| Cv\_item\_note | Package | String |

### Kelas DatabaseContract

|  |  |  |
| --- | --- | --- |
| **Nama Operasi** | **Visibility (private, public)** | **Keterangan** |
| DatabaseContract | public | Membuat class static |
| **Nama Atribut** | **Visibility (private, public)** | **Tipe** |
| Table\_name | public | String |
| Title | Public | String |
| Description | Public | String |
| date | public | String |
| onDate | Public | String |
| onTime | public | String |

### Kelas DatabaseHelper

|  |  |  |
| --- | --- | --- |
| **Nama Operasi** | **Visibility (private, public)** | **Keterangan** |
| DatabaseHelper | public | Membuat keterangan Database |
| onCreate | public | Membuat Database |
| onUpgrade | public | Mengupdate Database |
| **Nama Atribut** | **Visibility (private, public)** | **Tipe** |
| Title | Private | String |
| Description | Private | String |
| Date | Private | String |
| OnDate | Private | String |
| OnTime | Private | String |
| Sql\_Create\_Table\_Note | Private | String |
| Database\_name | Public | String |
| Database\_version | Private | Int |
| Table\_Name | Private | string |

### Kelas NoteHelper

|  |  |  |
| --- | --- | --- |
| **Nama Operasi** | **Visibility (private, public)** | **Keterangan** |
| NoteHelper | public | Memanggil database |
| ArrayList<Note> | public | Mengubah Database |
| insert | public | Memasukan values kedalam database |
| delete | public | Menghapu values dalam database |
| **Nama Atribut** | **Visibility (private, public)** | **Tipe** |
| dataBaseHelper | Private | Class |
| context | private | Context |
| Database | Private | SQLiteDatabase |
| Database\_table | Private | String |
| Title | Public | String |
| Description | Public | String |
| Date | Public | String |
| OnDate | Public | String |
| OnTime | Public | String |
| Table\_name | Public | String |

### Kelas Note

|  |  |  |
| --- | --- | --- |
| **Nama Operasi** | **Visibility (private, public)** | **Keterangan** |
| getId | public | Mendapatkan Id |
| setId | public | Menset Id |
| getTitle | public | Mendapatkan Judul |
| setTitle | public | Menset Judul |
| getDescription | public | Mendapatkan Deskripsi |
| setDescription | public | Menset Deskripsi |
| getOndate | public | Mendapatkan Tanggal |
| setOndate | public | Menset Tanggal |
| getOntime | public | Mendapatkan Waktu |
| setOntime | public | Menset Waktu |
| getDate | public | Mendapatkan Tanggal |
| setDate | public | Menset Tanggal |
| describeContents | public |  |
| writeToParcel | public | Membuat Parcel |
| Note | public | Memanggil sumber source |
| **Nama Atribut** | **Visibility (private, public)** | **Tipe** |
| Id | private | Int |
| title | private | String |
| description | private | String |
| ondate | private | String |
| ontime | private | String |
| date | private | String |

### Kelas CustomOnItemClickListener

|  |  |  |
| --- | --- | --- |
| **Nama Operasi** | **Visibility (private, public)** | **Keterangan** |
| CostumOnItemClickListener | public | Dipanggil ketika user menekan button |
| onClick | public | Dipanggil ketika user menekan button |
| onItemClickedCallback | public | Dipanggil ketika user menekan button |
| **Nama Atribut** | **Visibility (private, public)** | **Tipe** |
| position | Private | Int |
| onItemClickCallback | private | Class |

### Kelas FormAddUpdateActivity

|  |  |  |
| --- | --- | --- |
| **Nama Operasi** | **Visibility (private, public)** | **Keterangan** |
| onCreate | public | Dipanggil ketika menload FormAddUpdateActivity |
| onDestroy | Prrotected | Menghapus database |
| onClick | Public | Dipanggil ketika tombol ditekan |
| onCreateOptionsMenu | Public | Dipanggil ketika membuka menu |
| onOptionsItemSelected | Public | Dipanggil ketika membuka drawer |
| onBackPressed | Public | Mengembalikan tampilan ke layout sebelumnya |
| showAlertDialog | Private | Memunculkan teks peringatan |
| getCurrentDate | private | Mendapatkan tanggal |
| **Nama Atribut** | **Visibility (private, public)** | **Tipe** |
| Extra\_Note | Public | String |
| Extra\_Position | Public | String |
| isEdit | Private | Boolean |
| Request\_Add | Public | Static int |
| Request\_Update | Public | Static int |
| Result\_Update | Public | Static int |
| Result\_Delete | Public | Static int |
| note | Private | Class |
| position | Private | Int |
| noteHelper | Private | Class |
| tvOneTimeDate | Private | TextView |
| tvOneTimeTime | Private | TextView |
| btnOneTimeDate | Private | Button |
| btnOneTimeTime | Private | Button |
| calOneTimeDate | private | Calendar |
| calOneTimeTime | Private | Calendar |
| Activity\_form\_add\_update | Protected | LinearLayout |
| Toolbar2 | Protected | Toolbar |
| Tv\_one\_time\_alarm\_date | Protected | String |
| Tv\_one\_time\_alarm\_time | Protected | String |
| btn\_one\_time\_alarm\_date | Protected | String |
| btn\_one\_time\_alarm\_time | Protected | String |
| editTitle | Protected | String |
| Edit\_title | Protected | String |
| editDescription | Protected | String |
| Edit\_description | Protected | String |
| BtnSubmit | Protected | String |

### Kelas MainActivity

|  |  |  |
| --- | --- | --- |
| **Nama Operasi** | **Visibility (private, public)** | **Keterangan** |
| onCreate | public | Dipanggil ketika menload MainActivity |
| onClick | public | Diload ketika user menekan tombol |
| onPreExecute | protected | Dilakukan sebelum menexsekusi perintah |
| doInBackground | protected | Perintah menjalakan intruski di Backgorund |
| onPostExecute | protected | Menjalan intruksi |
| onActivityResult | protected | Menjalankan intruski dan menampilkan hasil intruksi |
| onDestroy | protected | Menghapus data |
| showSnackbarMessage | private | Menampilkan pesan |
| onBackPressed | Public | Kembali ke layout sebelumnya |
| onCreateOptionsMenu | Public | Dipanggil ketika membuka menu |
| onOptionsItemSelected | public | Dipanggil ketika membuka drawer |
| onNavigationItemSelected | public | Dipanggil saat menekan salah satu item di dalam drawer |
| **Nama Atribut** | **Visibility (private, public)** | **Tipe** |
| List | Private | ArrayList |
| Adapter | Private | Class |
| NoteHelper | Private | Class |
| mFireBaseAnalytic | private | FirebaseAnalytics |
| Activity\_main | protected | String |
| Toolbar1 | protected | String |
| Drawer\_layout | protected | String |
| Nav\_view | protected | String |
| rvNotes | Public | RecyclerView |
| progressBar | Public | ProgressBar |
| tvdataKosong | Public | ImageView |
| tvTextKosong | public | TextView |
| fabAdd | public | FloatingActionButton |

### Kelas PomodoroActivity

|  |  |  |
| --- | --- | --- |
| **Nama Operasi** | **Visibility (private, public)** | **Keterangan** |
| onCreate | proteced | Diload saat menjalankan PomodoroActivity |
| sendNotification | public | Menjalankan notifikasi countdown timer |
| startTimer | private | Memulai countdown timer |
| onTick | Public | Jumlah waktu countdown |
| onFinish | Public | Aksi jika countdown selesai |
| pauseTimer | private | Mempause countdown timer |
| risetTimer | Private | Mengulang countdown |
| updateCountDownText | Private | Text yang muncul saat countdown timer |
| onBackPressed | public | Kembali ke layout sebelumnya |
| onCreateOptionMenu | Public | Dipanggil ketika membuka menu |
| onOptionsItemSelected | Public | Dipanggil ketika membuka drawer |
| onNavigationItemSelected | public | Dipanggil saat menekan salah satu item di dalam drawer |
| **Nama Atribut** | **Visibility (private, public)** | **Tipe** |
| START\_TIME\_IN\_MILIS | Private | Static |
| countdown | Private | TextView |
| Btnstart\_pause | Private | Button |
| btnreset | Private | Button |
| mCountDownTimer | Private | CountDownTimer |
| mTimerRunning | Private | Boolean |
| mTimeleftnmilis | Private | Long |
| NOTIFICATION\_ID | Public | Static |
| CHANNEL\_ID | Public | Static |
| CHANNEL\_NAME | Public | Static |
| Activity\_pomodoro | protected | Layout |
| toolbar | protected | String |
| Drawer\_layout | protected | String |
| Nav\_view | protected | String |
| waktu | protected | String |
| tombolstart | protected | String |
| tombolreset | protected | String |

### Kelas SplashScreenActivity

|  |  |  |
| --- | --- | --- |
| **Nama Operasi** | **Visibility (private, public)** | **Keterangan** |
| onCreate | Protected | Dipanggil saat menload SplashScreenActivity |
| **Nama Atribut** | **Visibility (private, public)** | **Tipe** |
| FEATURE\_NO\_TITLE | public | static |
| Activity\_splash\_screen | public | LinearLayout |

## Diagram Kelas Keseluruhan

Bagian ini diisi dengan diagram kelas keseluruhan dari fase perancangan.



## Algoritma / Query

Nama Kelas : MainActivity

Nama Operasi : OnCreate dan OnActivity Result

Algoritma :

|  |
| --- |
| @Override protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_main*);  Toolbar toolbar = (Toolbar) findViewById(R.id.*toolbar1*);  setSupportActionBar(toolbar);   DrawerLayout drawer = (DrawerLayout) findViewById(R.id.*drawer\_layout*);  ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(  this, drawer, toolbar,   R.string.*navigation\_drawer\_open*,  R.string.*navigation\_drawer\_close*);  drawer.addDrawerListener(toggle);  toggle.syncState();   NavigationView navigationView = (NavigationView) findViewById(R.id.*nav\_view*);  navigationView.setNavigationItemSelectedListener(this);    getSupportActionBar().setTitle("Kumato Apps");   // Obtain the FirebaseAnalytics instance.  mFirebaseAnalytics = FirebaseAnalytics.*getInstance*(this);   rvNotes = (RecyclerView)findViewById(R.id.*rv\_notes*);  rvNotes.setLayoutManager(new LinearLayoutManager(this));  rvNotes.setHasFixedSize(true);   progressBar = (ProgressBar)findViewById(R.id.*progressbar*);  tvdataKosong = (ImageView) findViewById(R.id.*tvdatakosong*);  tvTextKosong = (TextView) findViewById(R.id.*tvtextkosong*);   fabAdd = (FloatingActionButton)findViewById(R.id.*fab\_add*);  fabAdd.setOnClickListener(this);     noteHelper = new NoteHelper(this);  noteHelper.open();   list = new ArrayList<>();     adapter = new NoteAdapter(this);  adapter.setListNotes(list);  rvNotes.setAdapter(adapter);   new LoadNoteAsync().execute();   } |
| @Override protected void onActivityResult(int requestCode, int resultCode, Intent data) {  super.onActivityResult(requestCode, resultCode, data);  // Akan dipanggil jika request codenya ADD  if (requestCode == FormAddUpdateActivity.*REQUEST\_ADD*){  if (resultCode == FormAddUpdateActivity.*RESULT\_ADD*){  new LoadNoteAsync().execute();  showSnackbarMessage("Satu item berhasil ditambahkan");  rvNotes.getLayoutManager().smoothScrollToPosition(rvNotes, new RecyclerView.State(), 0);  }  }  // Update dan Delete memiliki request code sama akan tetapi result codenya berbeda  else if (requestCode == *REQUEST\_UPDATE*) {  /\*  Akan dipanggil jika result codenya UPDATE  Semua data di load kembali dari awal  \*/  if (resultCode == FormAddUpdateActivity.*RESULT\_UPDATE*) {  new LoadNoteAsync().execute();  showSnackbarMessage("Satu item berhasil diubah");  int position = data.getIntExtra(FormAddUpdateActivity.*EXTRA\_POSITION*, 0);  rvNotes.getLayoutManager().smoothScrollToPosition(rvNotes, new RecyclerView.State(), position);  }  /\*  Akan dipanggil jika result codenya DELETE  Delete akan menghapus data dari list berdasarkan dari position  \*/  else if (resultCode == FormAddUpdateActivity.*RESULT\_DELETE*) {  int position = data.getIntExtra(FormAddUpdateActivity.*EXTRA\_POSITION*, 0);  list.remove(position);  adapter.setListNotes(list);  adapter.notifyDataSetChanged();  showSnackbarMessage("Satu item berhasil dihapus");  }  if (list.size() == 0) {  tvdataKosong.setVisibility(View.*VISIBLE*);  tvTextKosong.setVisibility(View.*VISIBLE*);  showSnackbarMessage("Tidak ada data saat ini");  }else{  tvdataKosong.setVisibility(View.*GONE*);  tvTextKosong.setVisibility(View.*GONE*);  }    }  } |

Query

|  |  |  |
| --- | --- | --- |
| No Query | Query | Keterangan |
| 0001 | OnCreate OnActivityResult |  |
|  |  |  |

Nama Kelas : PomodoroActivity

Nama Operasi : OnCreate dan sendNotification

Algoritma :

|  |
| --- |
| @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_pomodoro*);   Toolbar toolbar = (Toolbar) findViewById(R.id.*toolbar*);  setSupportActionBar(toolbar);   DrawerLayout drawer = (DrawerLayout) findViewById(R.id.*drawer\_layout*);  ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(  this, drawer, toolbar, R.string.*navigation\_drawer\_open*, R.string.*navigation\_drawer\_close*);  drawer.addDrawerListener(toggle);  toggle.syncState();   NavigationView navigationView = (NavigationView) findViewById(R.id.*nav\_view*);  navigationView.bringToFront();  navigationView.setNavigationItemSelectedListener(this);  countdown = findViewById(R.id.*waktu*);  btnstart\_pause= findViewById(R.id.*tombolstart\_pause*);  btnreset = findViewById(R.id.*tombolreset*);   btnstart\_pause.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  if (mTimerRunnig) {  pauseTimer();  } else {  startTimer();  }  }  });  btnreset.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  risetTimer();  }  });  updateCountDownText();  }   public void sendNotification() {  NotificationManager mNotificationManager = (NotificationManager) getSystemService(Context.*NOTIFICATION\_SERVICE*);  Uri alarmSound = RingtoneManager.*getDefaultUri*(RingtoneManager.*TYPE\_NOTIFICATION*);  NotificationCompat.Builder mBuilder = new NotificationCompat.Builder(this, *CHANNEL\_ID*)  .setSmallIcon(R.drawable.*iconnotif*)  .setLargeIcon(BitmapFactory.*decodeResource*(getResources(), R.drawable.*ic\_access\_alarm\_black\_24dp* ))  .setContentTitle("Pomodoro Teknik")  .setContentText("Waktunya Istarahat") // .setSubText(getResources().getString(R.string.subtext))  .setVibrate(new long[]{1000, 1000, 1000, 1000, 1000})  .setSound(alarmSound)  .setAutoCancel(true);   if (Build.VERSION.*SDK\_INT* >= Build.VERSION\_CODES.*O*) {  NotificationChannel channel = new NotificationChannel(*CHANNEL\_ID*, *CHANNEL\_NAME*, NotificationManager.*IMPORTANCE\_DEFAULT*);  mBuilder.setChannelId(*CHANNEL\_ID*);  if (mNotificationManager != null) {  channel.enableLights(true);  channel.setLightColor(Color.*RED*);  channel.enableVibration(true);  channel.setVibrationPattern(new long[]{100, 200, 300, 400, 500, 400, 300, 200, 400});  mNotificationManager.createNotificationChannel(channel);  }  }   Notification notification = mBuilder.build();   if (mNotificationManager != null) {  mNotificationManager.notify(*NOTIFICAITION\_ID*, notification);  }  } |

Query

|  |  |  |
| --- | --- | --- |
| No Query | Query | Keterangan |
| 0002 | OnCreate sendNotification |  |
|  |  |  |

Nama Kelas : FormAddUpdateActivity

Nama Operasi : OnCreate

Algoritma :

|  |
| --- |
| @Override protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_form\_add\_update*);  Toolbar toolbar = (Toolbar) findViewById(R.id.*toolbar2*);  setSupportActionBar(toolbar);   tvOneTimeDate = (TextView) findViewById(R.id.*tv\_one\_time\_alarm\_date*);  tvOneTimeTime = (TextView) findViewById(R.id.*tv\_one\_time\_alarm\_time*);  btnOneTimeDate = (Button) findViewById(R.id.*btn\_one\_time\_alarm\_date*);  btnOneTimeTime = (Button) findViewById(R.id.*btn\_one\_time\_alarm\_time*);  btnOneTimeDate.setOnClickListener(this);  btnOneTimeTime.setOnClickListener(this);   edtTitle = (EditText) findViewById(R.id.*edt\_title*);  edtDescription = (EditText) findViewById(R.id.*edt\_description*);  btnSubmit = (Button) findViewById(R.id.*btn\_submit*);  btnSubmit.setOnClickListener(this);    noteHelper = new NoteHelper(this);  noteHelper.open();   note = getIntent().getParcelableExtra(*EXTRA\_NOTE*);   if (note != null) {  position = getIntent().getIntExtra(*EXTRA\_POSITION*, 0);  isEdit = true;  }   String actionBarTitle = null;  String btnTitle = null;   if (isEdit) {  getSupportActionBar().setDisplayOptions(ActionBar.*DISPLAY\_SHOW\_CUSTOM*);  getSupportActionBar().setCustomView(R.layout.*title\_bar\_update*);  btnTitle = "Update";  edtTitle.setText(note.getTitle());  edtDescription.setText(note.getDescription());  } else {  getSupportActionBar().setDisplayOptions(ActionBar.*DISPLAY\_SHOW\_CUSTOM*);  getSupportActionBar().setCustomView(R.layout.*title\_bar*);  btnTitle = "Save";  }   getSupportActionBar().setTitle(actionBarTitle);  getSupportActionBar().setDisplayHomeAsUpEnabled(true);  btnSubmit.setText(btnTitle);   calOneTimeDate = Calendar.*getInstance*();  calOneTimeTime = Calendar.*getInstance*();  } |

Query

|  |  |  |
| --- | --- | --- |
| No Query | Query | Keterangan |
| 0003 | OnCreate |  |
|  |  |  |

Nama Kelas : DatabaseConstract

Nama Operasi : NoteColumns

Algoritma :

|  |
| --- |
| public class DatabaseContract {    static final class NoteColumns implements BaseColumns {  static String *TABLE\_NAME* = "note";   //Note title  static String *TITLE* = "title";  //Note description  static String *DESCRIPTION* = "description";  //Note date  static String *DATE* = "date";  static String *ONDATE* = "date";  static String *ONTIME* = "date";   }  } |

Query

|  |  |  |
| --- | --- | --- |
| No Query | Query | Keterangan |
| 0004 | NoteColumns |  |
|  |  |  |

Nama Kelas : DatabaseHelper

Nama Operasi : Membuat Database

Algoritma :

|  |
| --- |
| public static String *DATABASE\_NAME* = "dbkumato";  private static final int *DATABASE\_VERSION* = 1;  private static final String *SQL\_CREATE\_TABLE\_NOTE* = String.*format*("CREATE TABLE %s"  + " (%s INTEGER PRIMARY KEY AUTOINCREMENT," +  " %s TEXT NOT NULL," +  " %s TEXT NOT NULL," +  " %s TEXT NOT NULL)",  *TABLE\_NAME*,  NoteColumns.*\_ID*,  NoteColumns.*TITLE*,  NoteColumns.*DESCRIPTION*,  NoteColumns.*DATE*,  NoteColumns.*ONDATE*,  NoteColumns.*ONTIME* ); |

Query

|  |  |  |
| --- | --- | --- |
| No Query | Query | Keterangan |
| 0005 | Membuat Database |  |
|  |  |  |

Nama Kelas : NoteHelper

Nama Operasi : insert

Algoritma :

|  |
| --- |
| public long insert(Note note) {  ContentValues initialValues = new ContentValues();  initialValues.put(*TITLE*, note.getTitle());  initialValues.put(*DESCRIPTION*, note.getDescription());  initialValues.put(*DATE*, note.getDate());  initialValues.put(*ONDATE*, note.getOndate());  initialValues.put(*ONTIME*, note.getOntime());  return database.insert(*DATABASE\_TABLE*,  null, initialValues); } |

Query

|  |  |  |
| --- | --- | --- |
| No Query | Query | Keterangan |
| 0006 | Insert |  |
|  |  |  |

Nama Kelas : NoteHelper

Nama Operasi : update

Algoritma :

|  |
| --- |
| public int update(Note note) {  ContentValues args = new ContentValues();  args.put(*TITLE*, note.getTitle());  args.put(*DESCRIPTION*, note.getDescription());  args.put(*DATE*, note.getDate());  args.put(*ONDATE*, note.getOndate());  args.put(*ONTIME*, note.getOntime());  return database.update(*DATABASE\_TABLE*, args,  *\_ID* + "= '" + note.getId() + "'", null); } |

Query

|  |  |  |
| --- | --- | --- |
| No Query | Query | Keterangan |
| 0007 | Update |  |
|  |  |  |

Nama Kelas : NoteHelper

Nama Operasi : delete

Algoritma :

|  |
| --- |
| public int delete(int id) {  return database.delete(*TABLE\_NAME*,  *\_ID* + " = '" + id + "'", null); } |

Query

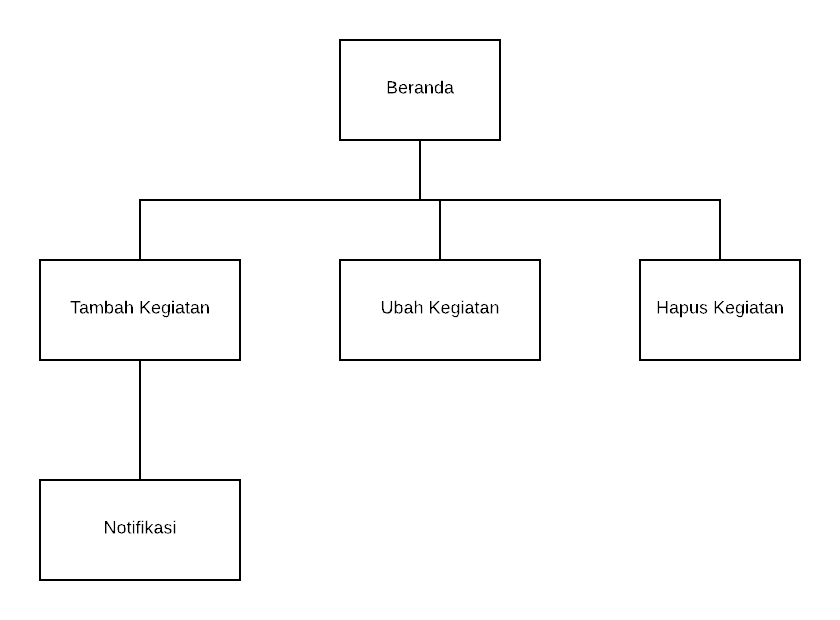
|  |  |  |
| --- | --- | --- |
| No Query | Query | Keterangan |
| 0008 | delete |  |
|  |  |  |

## Perancangan Data

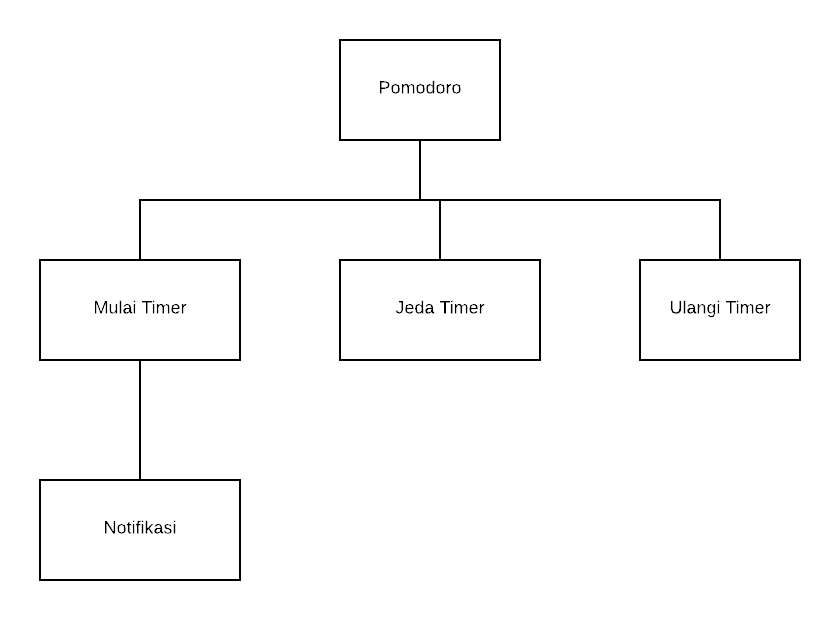
Berikut ini merupakan tabel perancangan data :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nama Field | Tipe Data | Panjang | Kunci | Keterangan |
| Id | integer | 11 | Primary key | Auto\_increment,  not null |
| Title | varchar | 50 |  | not null |
| Description | Text |  |  | not null |
| Date | Datetime |  |  | not null |
| onDate | Date |  |  | not null |
| onTime | Time |  |  | not null |

## Perancangan Arsitektural Menu



Gambar 21 Struktur Menu To Do List



Gambar 22 Struktur Menu Pomodoro

## Perancangan Antarmuka

Paparkan dalam bentuk gambar tentang perancangan antarmuka dalam bentuk seperti

### Perancangan antarmuka To Do List

|  |  |
| --- | --- |
| **K01** | |
|  | * A1 tombol untuk membuka Side Bar (K05) * A2 tombol untuk memunculkan pesan E1, jika memilih pengaturan maka akan menuju setting bahasa, jika memilih *Tentang*  maka akan muncil pesan E2 * A3 tombol untuk membuka Isi List Pekerjaan (K02) |
| Keterangan :   * Memakai Roboto Font ukuran 13 untuk konten * Menu Memakai Roboto Font ukuran 13 * Tulisan Nama Kosan memakai Font Open Sans ukuran 20 Putih * tombol A3 berwarna Merah Hati #AB0103 * warna background putih * sidebar dan head berwarna Merah Hati #AB0103 font roboto putih | |

|  |  |
| --- | --- |
| **K02** | |
|  | * A4 tombol untuk kembali Beranda (K01) * F1 adalah field mengisi *Nama Kegiatan* * F2 adalah field mengisi *Deskripsi Kegiatan* * A5 tombol untuk memilih *Tanggal Kegiatan* * A6 tombol untuk memilih *Waktu Kegiatan* * A7 menyimpan Aktifitas dan menuju ke Beranda (K03) * Jika menekan tombol A7 tanpa mengisi salah satu dari F1,F2,A5,dan A6 maka akan muncul pesan E3 |
| Keterangan :   * Memakai Roboto Font ukuran 13 untuk konten * Menu Memakai Roboto Font ukuran 13 * Tulisan Nama Kosan memakai Font Open Sans ukuran 20 Putih * tombol A3 berwarna Merah Hati #AB0103 * warna background putih * sidebar dan head berwarna Merah Hati #AB0103 font roboto putih | |

|  |  |
| --- | --- |
| **K03** | |
|  | * A1 tombol untuk membuka Side Bar (K05) * A2 tombol untuk memunculkan pesan E1, jika memilih pengaturan maka akan menuju setting bahasa, jika memilih *Tentang*  maka akan muncil pesan E2 * A3 tombol untuk membuka Isi List Pekerjaan (K02) * A8 tombol untuk mengedit kegiatan yang telah ada dan menuju *Edit Kegiatan (*K04) * Jika waktu dan tanggal pada A8 sudah sesuai maka akan muncul notifikasi berupa pesan (E4) |
| Keterangan :   * Memakai Roboto Font ukuran 13 untuk konten * Menu Memakai Roboto Font ukuran 13 * Tulisan Nama Kosan memakai Font Open Sans ukuran 20 Putih * tombol A3 berwarna Merah Hati #AB0103 * warna background putih * sidebar dan head berwarna Merah Hati #AB0103 font roboto putih | |

|  |  |
| --- | --- |
| **K04** | |
|  | * A4 tombol untuk kembali Beranda (K01) * F1 adalah field mengisi *Nama Kegiatan* * F2 adalah field mengisi *Deskripsi Kegiatan* * A5 tombol untuk memilih *Tanggal Kegiatan* * A6 tombol untuk memilih *Waktu Kegiatan* * A7 menyimpan Aktifitas dan menuju ke Beranda (K03) * Jika menekan tombol A7 tanpa mengisi salah satu dari F1,F2,A5,dan A6 maka akan muncul pesan E3 * A9 tombol untuk menghapus List Kegiatan, jika memilih tidak maka akan kembali ke Tampilan (KO4), jika memilih *Ya*  maka akan menuju menu Beranda (K01) |
| Keterangan :   * Memakai Roboto Font ukuran 13 untuk konten * Menu Memakai Roboto Font ukuran 13 * Tulisan Nama Kosan memakai Font Open Sans ukuran 20 Putih * tombol A3 berwarna Merah Hati #AB0103 * warna background putih * sidebar dan head berwarna Merah Hati #AB0103 font roboto putih | |

|  |  |
| --- | --- |
| **K05** | |
|  | * D1 tombol untuk membuka *Beranda* (K01) * D2 tombol untuk membuka *Pomodoro Teknik* (K06) |
| Keterangan :   * Memakai Roboto Font ukuran 13 untuk konten * Menu Memakai Roboto Font ukuran 13 * Tulisan Nama Kosan memakai Font Open Sans ukuran 20 Putih * tombol A3 berwarna Merah Hati #AB0103 * warna background putih * sidebar dan head berwarna Merah Hati #AB0103 font roboto putih | |

Perancangan Pesan Kesalahan

|  |  |  |
| --- | --- | --- |
| No | Kode | Keterangan |
| 1 | E01 | Opsi memilih Pengaturan dan Tentang |
| 2 | E02 | Tampilan Menu About |
| 3 | E03 | Data tidak boleh kosong |
| 4 | E04 | Notifikasi Kegiatan |

### Perancangan antarmuka Pomodoro

|  |  |
| --- | --- |
| **K06** | |
|  | * A1 tombol untuk membuka Side Bar (K05) * A2 tombol untuk memunculkan pesan E1, jika memilih pengaturan maka akan menuju setting bahasa, jika memilih *Tentang*  maka akan muncil pesan E2 * A10 tombol untuk memulai Timer * A11 tombol untuk reset Timer |
| Keterangan :   * Memakai Roboto Font ukuran 13 untuk konten * Menu Memakai Roboto Font ukuran 13 * Tulisan Nama Kosan memakai Font Open Sans ukuran 20 Putih * tombol A3 berwarna Merah Hati #AB0103 * warna background putih * sidebar dan head berwarna Merah Hati #AB0103 font roboto putih | |

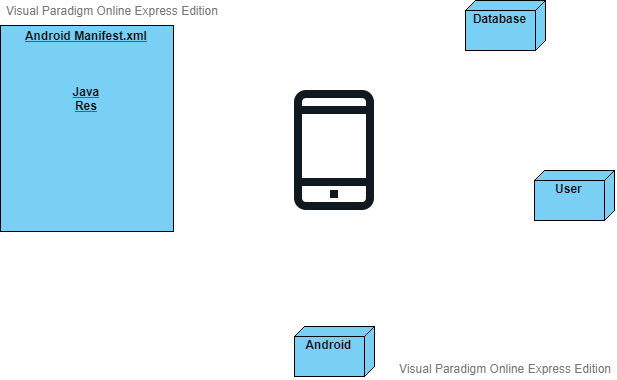
|  |  |
| --- | --- |
| **K07** | |
|  | * A1 tombol untuk membuka Side Bar (K05) * A2 tombol untuk memunculkan pesan E1, jika memilih pengaturan maka akan menuju setting bahasa, jika memilih *Tentang*  maka akan muncil pesan E2 * A12 tombol untuk menjeda Timer * A11 tombol untuk reset Timer * Jika waktu fokus (25 menit) sudah habis maka akan muncul notifikasi E6 * Jika waktu istirahat sudah habis maka akan muncul notifikasi E7 |
| Keterangan :   * Memakai Roboto Font ukuran 13 untuk konten * Menu Memakai Roboto Font ukuran 13 * Tulisan Nama Kosan memakai Font Open Sans ukuran 20 Putih * tombol A3 berwarna Merah Hati #AB0103 * warna background putih * sidebar dan head berwarna Merah Hati #AB0103 font roboto putih | |
| **K05** | |
|  | * D1 tombol untuk membuka *Beranda* (K01) * D2 tombol untuk membuka *Pomodoro Teknik* (K06) |
| Keterangan :   * Memakai Roboto Font ukuran 13 untuk konten * Menu Memakai Roboto Font ukuran 13 * Tulisan Nama Kosan memakai Font Open Sans ukuran 20 Putih * tombol A3 berwarna Merah Hati #AB0103 * warna background putih * sidebar dan head berwarna Merah Hati #AB0103 font roboto putih | |

Perancangan Pesan Kesalahan

|  |  |  |
| --- | --- | --- |
| No | Kode | Keterangan |
| 1 | E06 | Notifikasi waktunya istirahat |
| 2 | E07 | Notifikasi waktunya Fokus |

### Deployment Diagram

Berikut ini merupakan deployment diagram :



# Implementasi

## Lingkungan Implementasi

1. Lingkungan Hardware Pembanguan Perangkat Lunak Android :

|  |  |  |
| --- | --- | --- |
| No | Item | Spesifikasi |
| 1 | Processor | ARM Cortex-A7 1.0 GHz |
| 2 | Layar | 3.5’’ |
| 3 | Memory | 512 Mb |
| 4 | SD Card | 1 GV |

1. Lingkungan Software

Perangkat lunak yang digunakan dalam pembangunan perangkat ini

1. Minimal Android 4.0 Ice Cream Sandwich
2. Sql Lite

## Implementasi Data

1. Pembuatan Database

|  |
| --- |
| CREATE DATABASE dbkumato; |

1. Pembuatan Tabel Anak Asuh

|  |
| --- |
| CREATE TABLE IF NOT EXIST ‘list\_kegiatan’ (  ‘Id’ integer(11) PRIMARY KEY NOT NULL AUTO\_INCREMENT,  ‘Title’ varchar(50) NOT NULL,  ‘Description’ text NOT NULL,  ‘Date’ Datetime NOT NULL,  ‘onDate’ Date NOT NULL,  ‘onTime’ Time NOT NULL); |

## Implementasi Kelas

Berikut ini merupakan tabel implementasi kelas

Tabel 20 Implementasi Kelas

|  |  |  |  |
| --- | --- | --- | --- |
| No | Nama Kelas | Nama File Fisik | Nama File Executable |
| 1 | NoteAdapter | NoteAdapter.java | NoteAdapter.class |
| 2 | DatabaseContract | DatabaseContract.java | DatabaseContract.class |
| 3 | DatabaseHelper | DatabaseHelper.java | DatabaseHelper.class |
| 4 | NoteHelper | NoteHelper.java | NoteHelper.class |
| 5 | CostumOnItemClickListener | CostumOnItemClickListener.java | CostumOnItemClickListener.class |
| 6 | FormAddUpdateActivity | FormAddUpdateActivity.java | FormAddUpdateActivity.class |
| 7 | MainActivity | MainActivity.java | MainActivity.class |
| 8 | PomodoroActivity | PomodoroActivity.java | PomodoroActivity.class |
| 9 | SplashScreenActivity | SplashScreenActivity.java | SplashScreenActivity.class |
| 10 | Note | Note.java | Note.class |

## Implementasi Antarmuka

Berikut ini merupakan tabel implementasi Antarmuka

Tabel 21 Implementasi Antarmuka

|  |  |  |  |
| --- | --- | --- | --- |
| No | Antarmuka | Nama File Fisik | Nama File Executable |
| 1 | Conten\_main | Conten\_main.xml | Conten\_main.xml |
| 2 | Pomodoro\_content | Pomodoro\_content.xml | Pomodoro\_content.xml |
| 3 | Item\_note | Item\_note.xml | Item\_note.xml |
| 4 | Dialog\_about | Dialog\_about.xml | Dialog\_about.xml |
| 5 | App\_bar\_global | App\_bar\_global.xml | App\_bar\_global.xml |
| 6 | App\_bar\_main | App\_bar\_main.xml | App\_bar\_main.xml |
| 7 | Activity\_pomodoro | Activity\_pomodoro.xml | Activity\_pomodoro.xml |
| 8 | Activity\_main | Activity\_main.xml | Activity\_main.xml |
| 9 | activity\_form\_add\_update | Item\_form\_add\_update.xml | Item\_form\_add\_update.xml |
| 10 | Title\_bar | Title\_bar.xml | Title\_bar.xml |
| 11 | Title\_bar\_update | Title\_bar\_update.xml | Title\_bar\_update.xml |
| 12 | Nav\_header\_main | Nav\_header\_main.xml | Nav\_header\_main.xml |
| 13 | Activity\_splash\_screen | Activity\_splash\_screen.xml | Activity\_splash\_screen.xml |

# Pengujian

Pengujian sistem ini terdiri dari 2 tahap yakni pengujian menggunakan quesioner dan wawancara. Metode yang digunakan dalam pengujian sistem ini adalah dengan menggunakan metode pengujian wawancara dimana pengujian ini berfokus pada persyaratan fungsional perangkat lunak.

## Rencana Pengujian

Pada bagian rencana pengujian ini terbagi menjadi dua jenis, yaitu rencana pengujian fungsional dan rencana pengujian acceptance.

1. Rencana pengujian fungsional

Rencana pengujian yang akan dilakukan terdiri dari 1 buah subsistem, yaitu subsistem android dengan jenis pengujian ( ).

Tabel 22 Renca Pengujian Fungsional

|  |  |  |
| --- | --- | --- |
| No | Subsistem yang diuji | Jenis pengujian |
| 1 | Android | Black Box  White Box |

1. Rencana pengujian acceptance

Rencana pengujian acceptance yang dilaukan adalah melakukan wawancara langsung kepada narasumber yakni ( ) untuk melakukan pengujian subsistem android.

## Skenario Pengujian

Pada bagian skenario pengujian ini terbagi menjadi dua jenis, yaitu skenario pengujian fungsional dan skenario pengujian acceptance.

### Skenario Pengujian Fungsional

Skenario pengujian fungsional yang dilakukan adalah dengan menguji 1 buah subsistem, yaitu android.

1. Skenario Pengujian Sistem Android

Tabel 23 Skenario Pengujian Fungsional

|  |  |  |  |
| --- | --- | --- | --- |
|  | Kelas Uji | Poin Pengujian | Jenis Pengujian |
| 1 | Tambah Kegiatan | Isi Date  Isi Time  Isi Judul  Validasi field | *Equivalence Partitioning* |
| 2 | Edit Kegiatan | Isi Date  Isi Time  Isi Judul  Validasi field | *Equivalence Partitioning* |
| 3 | Hapus Kegiatan | Hapus Kegiatan | *Equivalence Partitioning* |
| 4 | Penggunaan teknik Pomodoro | Mulai Timer  Hentikan Timer | *Equivalence Partitioning* |

### Skenario Pengujian Acceptance

Pada bagian ini akan dipaparkan skenario pengujian acceptance berupa skenario wawancara dan kisi-kisi pernyataan kuesioner. Untuk daftar pertanyaan skenario wawancara terdiri dari 5 pertanyaan masing-masing terdapat bentuk negatif dan positifnya.

## Hasil Pengujian

Pada bagian hasil pengujian ini terbagi menjadi dua jenis, yaitu hasil pengujian fungsional dan hasil pengujian acceptance.

### Hasil Pengujian Fungsional

Pada bagian hasil pengujian fungsional ini di bagi menjadi 1 bagian, yaitu hasil pengujian subsistem android.

1. Hasil Pengujian Sistem Android

Hasil pengujian ini berisi pemaparan dari rencana pengujian yang disusun pada skenario pengujian. Berikut ini adalah pemaparan tiap butir pengujian yang terdapat skenario pengujian:

1. Pengujuan Tambah kegiatan

Tabel 24 Tambah Kegiatan

|  |  |  |  |
| --- | --- | --- | --- |
| Kasus dan Hasil Uji (Data benar) | | | |
| Data Masukan | Hasil yang diharapkan | Pengamatan | Kesimpulan |
| Input : Nama Kegiatan | *Filed Nama Kegiatan* terisi sesuai keinginan user | *Field Nama* terisi | Diterima |
| Input : Deskripsi Kegiatan | *Field Deskripsi* terisi | *Field Deskripsi*  Terisi | Diterima |
| Input : Tanggal Kegiatan | User mengisi Tanggal sesuai dengan kegiatan | *Input Tanggal* terisi | Diterima |
| Input : Waktu Kegiatan | User mengisi waktu sesuai dengan kegiatan | *Input Waktu*  terisi | Diterima |

1. Pengujian Tambah Data Salah

Tabel 25 Tambah Data (Salah)

|  |  |  |  |
| --- | --- | --- | --- |
| Kasus dan Hasil Uji (Data benar) | | | |
| Data Masukan | Hasil yang diharapkan | Pengamatan | Kesimpulan |
| Input : (Kosong) | Ketika menekan tombol *Save* , menampilkan pesan “*Data Tidak Boleh Kosong*” | Muncul Toast dengan pesan “*Data Tidak Boleh Kosong”* | Diterima |

1. Pengujian Edit Data Kegiatan

Tabel 26 Edit Data Kegiatan

|  |  |  |  |
| --- | --- | --- | --- |
| Kasus dan Hasil Uji (Data benar) | | | |
| Data Masukan | Hasil yang diharapkan | Pengamatan | Kesimpulan |
| Input : Nama Kegiatan | *Filed Nama Kegiatan* terisi sesuai keinginan user | *Field Nama* terisi | Diterima |
| Input : Deskripsi Kegiatan | *Field Deskripsi* terisi | *Field Deskripsi*  Terisi | Diterima |
| Input : Tanggal Kegiatan | User mengisi Tanggal sesuai dengan kegiatan | *Input Tanggal* terisi | Diterima |
| Input : Waktu Kegiatan | User mengisi waktu sesuai dengan kegiatan | *Input Waktu*  terisi | Diterima |
| Tombol Hapus | Muncul pesan konfirmasi hapus | Menampilkan pesan konfirmasi kepada pengguna berupa “*Ya”* dan *“Tidak”* | Diterima |

1. Pengujian Edit Data Kegiatan Salah

Tabel 27 Edit Data Kegiatan (Salah)

|  |  |  |  |
| --- | --- | --- | --- |
| Kasus dan Hasil Uji (Data benar) | | | |
| Data Masukan | Hasil yang diharapkan | Pengamatan | Kesimpulan |
| Input : (Kosong) | Ketika menekan tombol *Save* , menampilkan pesan “*Data Tidak Boleh Kosong*” | Muncul Toast dengan pesan “*Data Tidak Boleh Kosong”* | Diterima |

1. Pengujian Fitur Pomodoro

Tabel 28 Pengujian Fitur Pomodoro

|  |  |  |  |
| --- | --- | --- | --- |
| Kasus dan Hasil Uji (Data benar) | | | |
| Data Masukan | Hasil yang diharapkan | Pengamatan | Kesimpulan |
| Menekan Tombol *Mulai* | Ketika menekan tombol *Mulai* , timer hitung mundur akan berjalan | Timer hitung mundur berjalan selama 25 menit | Diterima |
| Menekan Tombol *Jeda* | Ketika menekan tombol *Jeda* , timer akan *Pause* | Timer akan berhenti berjalan | Diterima |
| Menekan Tombol *Ulang* | Ketika menekan tombol *Ulang,* timer akan kembali ke kondisi awal | Timer hitung mundul akan berhenti dan angka akan kembali ke awal | Diterima |

### Hasil Pengujian Acceptance

Hasil pengujian acceptance ini memiliki 1 buah bagian yaitu Quesioner

1. Hasil Pengujian Quesioner

Tabel 29 Hasil Pengujian Kuesioner

|  |  |  |
| --- | --- | --- |
| No | Pertanyaan | Hasil kesimpulan Quesioner |
| 1 | Apakah tampilan aplikasi sudah enak di pandang dan tidak membuat mata anda lelah? (warna) | Sudah Bagus |
| 2 | Apakah tata letak penempatan tombol form sudah memudahkan anda apa masih kesusahan menggunakannya? | Membantu |
| 3 | Apa anda paham aplikasi ini tentang apa? | Paham |
| 4 | Apakah anda paham dengan fitur penambah ke fokusan melalui teknik pomodoro? | Tidak Paham |
| 5 | Apakah setelah pemasangan aplikasi kami di smartphone anda mengalami masalah (Error System)? | Tidak Ada Masalah |
| 6 | Apakah anda akan menungu pembaruan fitur dari kami atau uninstal ? | Menunggu |
| 7 | Saran untuk next fitur yang ingin ada di aplikasi supaya ada masukan untuk rencana pembaruan ke depan? | Penambahan Fitur |

Dari 9 orang yang mengisi quesioner, kita bisa mengambil kesimpulan dalam tabel dibawah ini :

Tabel 30 Kesimpulan Tabel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Pertanyaan | Jumlah Setuju | Jumlah Tidak Setuju | Kesimpulan |
| 1 | Apakah tampilan aplikasi sudah enak di pandang dan tidak membuat mata anda lelah? (warna) | 10 | 0 | Setuju |
| 2 | Apakah tata letak penempatan tombol form sudah memudahkan anda apa masih kesusahan menggunakannya? | 8 | 1 | Setuju |
| 3 | Apa anda paham aplikasi ini tentang apa? | 8 | 1 | Setuju |
| 4 | Apakah anda paham dengan fitur penambah ke fokusan melalui teknik pomodoro? | 3 | 6 | Tidak Setuju |
| 5 | Apakah setelah pemasangan aplikasi kami di smartphone anda mengalami masalah (Error System)? | 7 | 2 | Setuju |
| 6 | Apakah anda akan menungu pembaruan fitur dari kami atau uninstal ? | 7 | 2 | Setuju |

## Evaluasi Pengujian Acceptance

Pada bagian evaluasi pengujian ini terbagi menjadi dua jenis, yaitu evaluasi pengujian fungsional dan evaluasi pengujian acceptance.

1. Evaluasi Pengujian Fungsional

Berdasarkan hasil pengujian fungsional yang dilakukan maka dapat disimpulkan bahwa subsistem android yang dibangun ( ).

1. Evaluasi Pengujian Acceptance

Tabel 31Evaluasi Pengujian Acceptance

|  |  |  |
| --- | --- | --- |
| No | Tujuan | Hasil evaluasi acceptance |
| 1 | Memudahkan pemakai untuk membuat list kegiatan | Membantu |
| 2 | Memudahkan pemakai untuk menambah tingkat kefokusan | Membantu |

# Kesimpulan dan Saran

## Kesimpulan

Berdasarkan hasil pengembangan sistem yang telah dilakukan, berikut adalah suatu kesimpulan yang berhubungan dengan pembahasan yang telah dikemukakan di bab-bab sebelumnya.

1. Aplikasi Kumato berbasis *Android* dapat membantu pengguna untuk mengatur jadwal yang harus dilakukan dengan fitur *To Do List* dan juga membantu pengguna agar tetap fokus dengan fitur *Pomodoro.*
2. Pengguna dapat mengatur waktu dan tanggal kegiatan yang akan dilakukan dalam fitur *To Do List.* Dan akan mendapatkan notifikasi sesuai waktu dan tanggal yang diatur.
3. Pengguna dapat menambah tingkat kefokusan dengan menggunakan fitur *Pomodoro.* Saat menggunakan fitur ini, pengguna akan diberikan watu selama 25 menit untuk fokus dan 5 menit untuk istirahat, terdapat 4 sesi dalam fitur ini dimana 3 sesi pertama pengguna diberikan waktu 25 menit untuk fokus dan 5 menit untuk istirahat. Saat sesi ke 4 pengguna akan diberikan waktu istirahat tambahan sebanyak 30 menit.
4. Pengguna dapat memilih pilihan bahasa aplikasi yang terdiri dari bahasa Indonesia dan Inggris.

## Saran

Tugas besar yang berjudul “Dokumen Pembangunan Perangkat Lunak Kumato” dapat dilakukan pengembangan lebih lanjut agar lebih optimal dan sesuai dengan kebutuhan. Adapun dibawah ini merupakan saran dari pengembang agar pengembangan terhadap aplikasi ini dapat dilakukan pada tahap selanjutnya.

* + - 1. Pada pengembangan aplikasi selanjutnya diharapkan dapat memperbaiki tampilan agar lebih mudah digunakan (*user friendly*).
      2. Pada pengembangan aplikasi selanjutnya diharapkan dapat memperbaiki bug dalam fitur aplikasi.

# DAFTAR PUSTAKA

|  |  |
| --- | --- |
| [1] | E. G. Ramadhan, “Javelin Experiment Board,” Jumat November 2018. [Online]. Available: https://medium.com/codelabs-unikom/javelin-experiment-board-487c65c991f3. [Diakses 02 November 2018]. |
| [2] | A. Rahmani, “Cara Cerdas Melakukan Validasi Ide ke Dalam Sebuah Rencana Bisnis,” 02 November 2018. [Online]. Available: https://www.jurnal.id/id/blog/2018/cara-cerdas-memvalidasi-ide-besar-bisnis-ke-dalam-sebuah-rencana. [Diakses 02 November 2018]. |
| [3] | P. Ritonga, “Pengertian Unified Modeling Language ( UML ) dan Modelnya Menurut Pakar dan Ahli,” 02 November 2018. [Online]. Available: https://bangpahmi.com/pengertian-unified-modeling-language-uml-dan-modelnya-menurut-pakar-dan-ahli/. [Diakses 02 November 2018]. |
| [4] | “Ngoding00,” [Online]. Available: https://www.ngodingoo.com/sekilas-tentang-bahasa-pemrograman-java/. |
| [5] | “Codepolitan,” [Online]. Available: https://www.codepolitan.com/belajar-uml-sequence-diagram-57fdb1a5ba777-17044. |
| [6] | “Dictio,” [Online]. Available: https://www.dictio.id/t/apa-yang-dimaksud-dengan-diagram-aktivitas-atau-activity-diagram/15129. |
| [7] | “UMIALFAH,” [Online]. Available: http://fatimahumi.blogspot.com/2014/03/uml-class-diagram.html. |
| [8] | “Materi Dosen,” [Online]. Available: http://www.materidosen.com/2017/04/use-case-diagram-lengkap-studi-kasus.html. |
| [9] | Iosipratama, “Product Design Sprint 101 — Bahasa Indonesia,” 1 Februari 2018. [Online]. Available: https://medium.com/@iosipratama/product-design-sprint-101-bahasa-indonesia-4b2acde80540. |
| [10] | B. Logistik, “Model Bisnis Kanvas Adalah Cara Memetakan Bisnis Lebih Simpel dan Efisien,” MGT Logistik, 29 Agustus 2018. [Online]. Available: https://mgt-logistik.com/model-bisnis-kanvas-adalah/. [Diakses 20 Desember 2018]. |
| [11] | D. Amelia, “Makalah OOAD Dan UML,” Scribd, 1 Januari 2017. [Online]. Available: https://id.scribd.com/document/341568507/Makalah-OOAD-Dan-UML. [Diakses 20 Desember 2018]. |
| [12] | Yasha, “Memahami Google Design Sprint Demi Kesuksesan Bisnis,” Dewaweb, 9 November 2017. [Online]. Available: https://www.dewaweb.com/blog/google-design-sprint/. [Diakses 20 Desember 2018]. |
| [13] | A. R. Pratama, “Belajar Unified Modeling Language (UML) - Pengenalan,” [Online]. Available: https://www.codepolitan.com/unified-modeling-language-uml. |
| [14] | Mahadisuta, “Pengujian White Box Dan Black Box Wajib Dilakukan Seorang Developer,” [Online]. Available: https://www.mahadisuta.com/artikel/5a1f5eb584545/pengujian-white-box-dan-black-box-wajib-dilakukan-seorang-developer. |

# LAMPIRAN

Bagian lampiran diisi dengan

* Pembagian tugas

Scrum Master : Ary Sugiarto

Product Owner : Farhan Arif N

Team Development : Feryanto Abi F

Anisa Dewi Umi K

Ahmad Tri Utama

* Rencana iterasi pembangunan perangkat lunak

1. Validasi Ide
2. Diskusi Hasil Validasi Ide
3. Analisis Desain tampilan
4. Diskusi Hasil Analisi Desain
5. Proses Development (Koding)

* notulen koordinasi team

1. Feryanto Abi F

* Resume hasil jawaban wawancara per responden
* Foto kegiatan wawancara

-