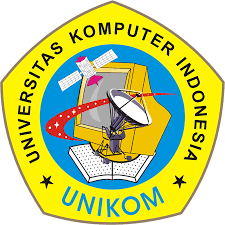
**DOKUMEN PEMBANGUNAN PERANGKAT LUNAK**

**SmartKosan App**

MATAKULIAH   
MANAJEMEN PROYEK PERANGKAT LUNAK



**Disusun Oleh :**

1. 10116239 – Utut Ardiansah
2. 10116254 – Ferri Lasmi Halim
3. 10116243 – Krisna Setiadi
4. 10116260 – Ary Sugiarto
5. 10116263 – Farhan Arif Naufal
6. 10116264 – Feryanto Abi Febrina

**Kelas : IF-6**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK DAN ILMU KOMPUTER**

**UNIVERSITAS KOMPUTER INDONESIA**

# DAFTAR ISI

[DAFTAR ISI 1](#_Toc12742950)

[BAB I PENDAHULUAN 3](#_Toc12742951)

[1.1 Latar Belakang Masalah 3](#_Toc12742952)

[1.2 Identifikasi Masalah 4](#_Toc12742953)

[1.3 Pembatasan Masalah 4](#_Toc12742954)

[1.4 Maksud dan Tujuan 4](#_Toc12742955)

[1.5 Metodologi Penelitian 4](#_Toc12742956)

[1.5.1 Metode Pengumpulan Data 5](#_Toc12742957)

[1.5.2 Metode Pembangunan Perangkat Lunak 5](#_Toc12742958)

[2. TINJAUAN PUSTAKA 8](#_Toc12742959)

[2.1 Validasi Ide 8](#_Toc12742960)

[2.2 Javelin Board 8](#_Toc12742961)

[2.3 Google Design Sprint 9](#_Toc12742962)

[2.4 OOAD dengan UML 10](#_Toc12742963)

[2.4.1 UML 10](#_Toc12742964)

[2.4.2 Usecase 11](#_Toc12742965)

[2.4.3 Usecase scenario 12](#_Toc12742966)

[2.4.4 Activity diagram 13](#_Toc12742967)

[2.4.5 Class diagram 14](#_Toc12742968)

[2.4.6 Sequence diagram 15](#_Toc12742969)

[2.4.7 Deployment Diagram 16](#_Toc12742970)

[2.5 Mobile Development (Native Android) 17](#_Toc12742971)

[2.6 Web Development (CI) 18](#_Toc12742972)

[2.7 Firebase 18](#_Toc12742973)

[2.8 Web Service (Json) 18](#_Toc12742974)

[2.9 Business Model Canvas 18](#_Toc12742975)

[2.10 Pengujian Black Box 19](#_Toc12742976)

[2.11 Pengujian Acceptance 19](#_Toc12742977)

[3. KEBUTUHAN PERANGKAT LUNAK 20](#_Toc12742978)

[3.1 Analisis Sistem Berjalan 20](#_Toc12742979)

[3.2 Analisis Arsitektur Sistem 21](#_Toc12742980)

[3.3 Spesifikasi Kebutuhan Perangkat Lunak 22](#_Toc12742981)

[3.3.1 Kebutuhan Fungsional. 22](#_Toc12742982)

[3.3.2 Kebutuhan Non Fungsional 22](#_Toc12742983)

[3.4 Analisis Kebutuhan Nonfungsional 23](#_Toc12742984)

[3.4.1 Analisis Kebutuhan Perangkat Lunak 23](#_Toc12742985)

[3.4.2 Analisis Kebutuhan Perangkat Keras 24](#_Toc12742986)

[3.4.3 Analisis Kebutuhan Perangkat Pikir 25](#_Toc12742987)

[3.4.4 Analisis Kebutuhan Jaringan 25](#_Toc12742988)

[3.5 Analisis Kebutuhan Data 26](#_Toc12742989)

[3.6 Model Use Case 27](#_Toc12742990)

[3.6.1 Diagram Use Case 27](#_Toc12742991)

[3.6.2 Definisi Actor 27](#_Toc12742992)

[3.6.3 Definisi Use Case 28](#_Toc12742993)

[3.6.4 Skenario Use Case 29](#_Toc12742994)

[3.7 Model Activity Diagram 38](#_Toc12742995)

[4. MODEL ANALISIS 39](#_Toc12742996)

[4.1 Diagram Class Analisis 39](#_Toc12742997)

[4.2 Identifikasi Kelas Analisis 39](#_Toc12742998)

[4.3 Realisasi Kelas Terhadap Use Case 40](#_Toc12742999)

[4.3.1 Identifikasi Kelas Analisis tiap Paket 40](#_Toc12743000)

[5. Model Perancangan 40](#_Toc12743001)

[5.1 Realisasi Use Case Tahap Perancangan 40](#_Toc12743002)

[5.2 Perancangan Detil Elemen Logical View 40](#_Toc12743003)

[5.3 Diagram Kelas Keseluruhan 40](#_Toc12743004)

[5.4 Algoritma / Query 40](#_Toc12743005)

[5.5 Perancangan Data 40](#_Toc12743006)

[5.6 Perancangan Arsitektural Menu 42](#_Toc12743007)

[5.7 Perancangan Antarmuka 42](#_Toc12743008)

[6. Implementasi 42](#_Toc12743009)

[6.1 Lingkungan Implementasi 42](#_Toc12743010)

[6.2 Implementasi Data 43](#_Toc12743011)

[6.3 Implementasi Kelas 44](#_Toc12743012)

[6.4 Implementasi Antarmuka 45](#_Toc12743013)

[7. Pengujian 46](#_Toc12743014)

[7.1 Rencana Pengujian 46](#_Toc12743015)

[7.2 Skenario Pengujian 46](#_Toc12743016)

[7.2.1 Skenario Pengujian Fungsional 47](#_Toc12743017)

[7.2.2 Skenario Pengujian Acceptance 47](#_Toc12743018)

[7.3 Hasil Pengujian 47](#_Toc12743019)

[7.3.1 Hasil Pengujian Fungsional 47](#_Toc12743020)

[7.3.2 Hasil Pengujian Acceptance 48](#_Toc12743021)

[7.4 Evaluasi Pengujian Acceptance 50](#_Toc12743022)

[8. Kesimpulan dan Saran 50](#_Toc12743023)

[8.1 Kesimpulan 50](#_Toc12743024)

[8.2 Saran 51](#_Toc12743025)

[DAFTAR PUSTAKA 51](#_Toc12743026)

[LAMPIRAN 51](#_Toc12743027)

# BAB I PENDAHULUAN

## Latar Belakang Masalah

Kos-kosan begitulah istilah yang sangat sering kita dengar, bahkan rasanya sudah tidak asing lagi ditelingan masyarakat. Atau bagi orang-orang yang sedang merantau, bekerja, maupun kuliah. Jasa penyewaan Kos-Kosan ini meberikan benefit yang menjanjikan jika di bangun di area yang strategis.

Dimana para penghuni jasa Kos-Kosan ini akan di berikan keuntungan tersendiri bagi orang tersebut, seperti halnya jarak tempuh kantor atau kampus yang menjadi lebih dekat, sehingga mereka dapat menghemat waktu dan biaya transportasi.

Berdasarkan wawancara yang kami lakukan dengan pemilik jasa penyewaan kosan beliau mengatakan bahwa data yang beliau catat itu sering hilang dan rusak, serta dalam melakukan pencarian data penghuni kosan susah dikarenakan berkas yang dicari kadang hilang ataupun sudah rusak.

Berdasarkan pemaparan fakta dan masalah yang telah dijelaskan sebelumnya, ditemukan suatu teknologi yang diharapkan menjadi solusi yaitu dengan membangun sebuah perangkat lunak Smart Kosan berbasis Mobile (Andorid) dan memiliki server berbasis Website untuk membantu pemilik kosan dalam melakukan penyimpanan ataupun pencarian data penghuni kosan.

## Identifikasi Masalah

Masalah yang sering dihadapi oleh pemilik jasa penyewan kosan diantaranya :

1. Data penghuni kosan sering tidak tercatat dengan rapih dan sering hilang;
2. Kesulitan mencari data penghuni yang sudah disimpan;
3. Telat dan Lupa melakukan pembayaran.

## Pembatasan Masalah

1. Pemilik jasa penyewaan kosan tidak mencatat data penghuni kosannya dengan baik;
2. Penghuni kosan sering lupa melakukan pembayaran;
3. Data penghuni kosan sering hilang atau rusak.

## Maksud dan Tujuan

Maksud dari penelitian ini adalah dengan membangun perangkat lunak SmartKosan App berbasis Mobile (Andorid) dan memiliki server berbasis Website dengan tujuan sebagai berikut :

1. Pemilik jasa penyewaan kosan bisa menyimpan data penghuni kosan dengan baik;
2. Penghuni kosan diingatkan tanggal pembayaran;
3. Data penghuni kosan terjaga dan mudah dipakai kembali.

## Metodologi Penelitian

Metodologi penelitian yang digunakan dalam penelitian ini adalah metode-metode deskriptif dimana metode ini digunakan dalam penelitian awal untuk menghimpun data tentang kondisi yang ada, faktor pendukung dan penghambat pengembangan, serta penggunaan produk dimana produk tersebut akan diterapkan. Metode ini diambil dari sebuah paper penelitian pendidikan karya Surya Dharma MPA., Ph. D dimana metode ini termasuk kedalam Penelitian dan Pengembangan atau *Research and Development* (R&D) yang merupakan rangkaian proses atau langkah-langkah dalam rangka mengembangkan satu produk baru atau menyempurnakan produk yang telah ada agar dapat dipertanggungjawabkan. Penelitian dan pengembangan merupakan metode penghubung atau pemutus kesenjangan antara penelitian dasar dengan penelitian terapan.

Metode penelitian ini memiliki dua metode dalam pelaksanaanya yaitu metode pengumpulan data dan metode pembangunan perangkat lunak. Berikut adalah pemaparannya:

### 1.5.1 Metode Pengumpulan Data

Berikut adalah metode pengumpulan data dalam penelitian ini:

1. Wawancara

Rekaputlasi Interview tahap pertama

Tabel 1 Hasil Validasi Ide

Daftar pertanyaan wawancara yang penting seputar inti masalah:

### Metode Pembangunan Perangkat Lunak

Pada pembangunan perangkat lunak ini kami menggunakan Metode Google Sprint. Metode Google Design Sprint adalah Design sprint adalah suatu metode untuk build product concept dan prototype dalam waktu 5 hari dimana memiliki 5 tahapan yang sangat komunikatif dan interaktif untuk mengeluarkan semua ide, inspirasi, masalah yang ada, solusi yang kemudian diwujudkan dalam prototype yang harus dan di cross check ke calon pengguna[9].

Berikut ini tahapan yang sudah kami lakukan, di antaranya:

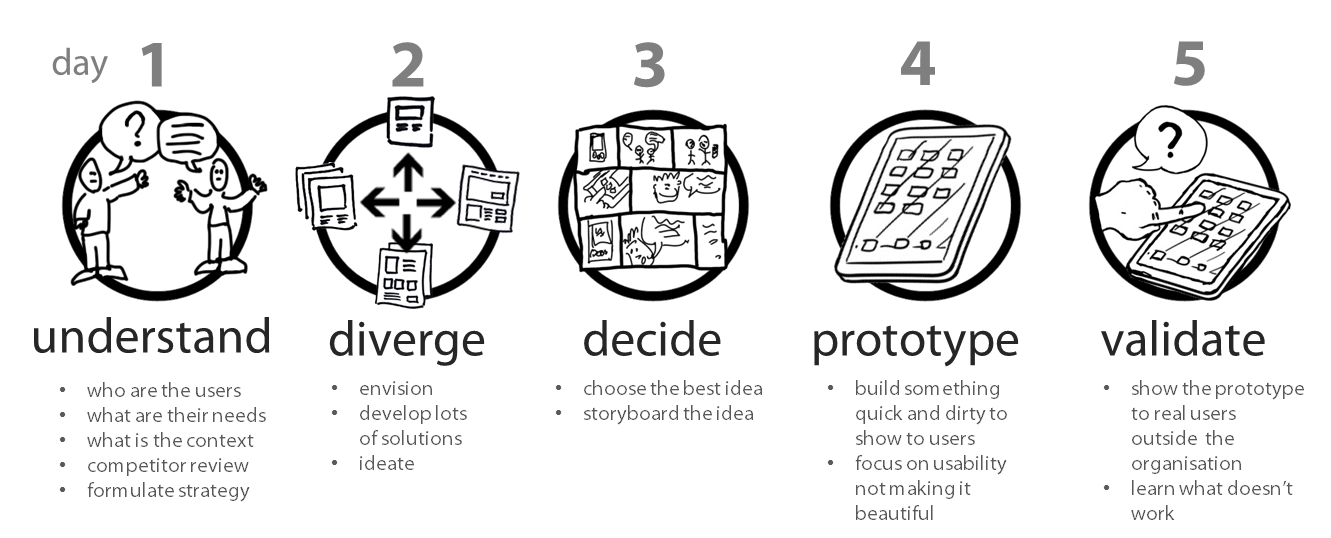
**Tahapan Pertama adalah Validasi Ide :**

1. Validasi Ide
2. Wawancara
3. Mencari masalah
4. Mengumpulkan informasi mengenai keinginan pengguna dalam produk yang akan di bangun.

**Tahapan Kedua adalah Design Sprint :**

Kami mempraktikkan langsung bagaimana menggunakan metode Google Design Sprint dalam membangun produk dengan waktu yang sedikit.

Ada 5 tahapan yaitu sebagai berikut :



Gambar 1 Design Sprint

1. Understand

Pada hari pertama, intinya adalah pengenalan tiap anggota, persiapan yang harus dilakukan dan mendiskusikan bagaimana cara menyelesaikan masalah dan hasil wawancara tersebut menjadi bahan diskusi bersama dengan semua anggota tim. Dari hasil diskusi tersebut kita dapat menentukan rencana jangka panjang dan masalah yang mungkin akan datang, dari hal tersebut kita membuat MAP atau alur aplikasi yang akan dibuat dan disempurnakan dengan mendiskusikan solusi berupa How Might We Notes (HMW).

1. Diverge

Pada hari kedua, Membuat sketsa aplikasi. Tiap anggota membuat sketsa sesuai ide masing – masing yang dilakukan setelah adanya penggambaran demo aplikasi

1. Decide

Masih dihari kedua. Setelah membuat sketsa, tiap anggota menjelaskan design sketsa yang telah dibuat . Tim berkumpul untuk melakukan voting, dimana ide terbaik diputuskan oleh suara terbanyak akan tetapi produk ownerlah yang mempunyai keputusan mutlak. Setelah mendapatkan hasil yang relevan, maka dibuatlah storyboard untuk tahap prototype.

1. Prototype

Pada hari ketiga, designer akan membuat prototipe dari design yang sudah dibuat dan disepakati pada hari sebelumya. Lalu tim akan mencoba maupun menganalisis prototype yang sudah dibuat sebelum dicoba oleh pengguna. Prototype ini hanya sekedar menampilkan fitur utama dari aplikasi yang akan dibuat, yang akan digunakan oleh pengguna kedepannya.

1. Validate

Prototype yang sudah jadi pada hari sebelumnya, pada hari keempat ini kita akan terjun kelapangan untuk meminta pendapat dari calon pengguna. Apa saja kekurangan dan kelebihan dari hasil prototype dalam sudut pandang pengguna.

# TINJAUAN PUSTAKA

Berikut merupakan beberapa bagian yang dimasukan dalam tinjauan pusaka

## 2.1 Validasi Ide

Validasi ide adalah proses awal yang harus dilakukan sebelum meluncurkan nama bisnis, slogan, produk, layanan, atau aplikasi baik yang berbasis website ataupun Mobile. Ini seperti proses yang digunakan perusahaan besar untuk melakukan penelitian dan pengembangan untuk menguji ide produk sebelum dirilis ke masyarakat umum. Penelitian dapat mencakup survei, wawancara, riset online, dan sebagainya untuk mendapatkan hasil terbaik [2].

## 2.2 Javelin Board

Javelin Board atau Papan Validasi merupakan salah satu cara untuk validasi ide startup menguraikan menggunakan Experiment Board. Dimana validasi ide merupakan salah satu tahap krusial yang menentukan. Dari analisis menggunakan Javelin Board kita dapat mengetahui titik kritis, identifikasi masalah, asumsi, melakukan metode pengujian dan juga penentuan kriteria sukses. Board ini terbagi menjadi dua bagian, yaitu bagian kiri atau disebut dengan area Brainstorming. Dan bagian kanan yang disebut dengan Area Execute [1].

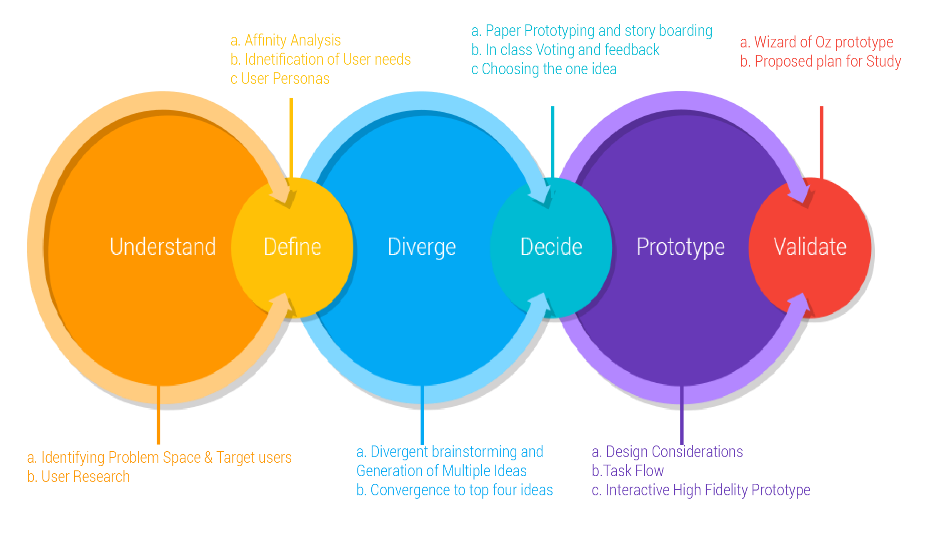


Gambar 2 Javelin Board

Langkah pertama yang harus di ketahui adalah, menentukan calon konsumen untuk produk yang dibuat. Kumpulkan beberapa calon potensial, kemudian pilih salah satu yang akan menjadi fokusnya. Untuk tahap selanjutnya di javelin Board sudah terdapat instruksi secara singkat supaya mempermudah kita dalam pengisian [1].

## 2.3 Google Design Sprint

Design sprint adalah suatu metode untuk build product concept dan prototype dalam waktu 5 hari dimana memiliki 5 tahapan yang sangat komunikatif dan interaktif untuk mengeluarkan semua ide, inspirasi, masalah yang ada, solusi yang kemudian diwujudkan dalam prototype yang harus dan di cross check ke calon pengguna[9]. Design Sprint ini dibuat oleh Jake Knapp dari Google Venture pada tahun 2010[12]. Sehingga disebut dengan Google Design Sprint.



Gambar 3 Google Design sprint

## 2.4 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].

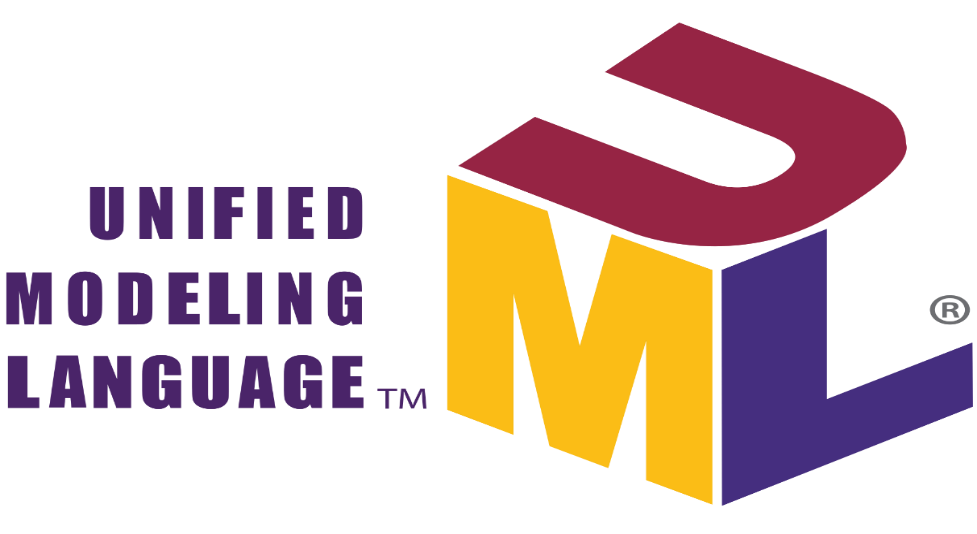
### 2.4.1 UML

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

### 2.4.2 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

### 2.4.3 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

### 2.4.4 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

### 2.4.5 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

### 2.4.6 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.

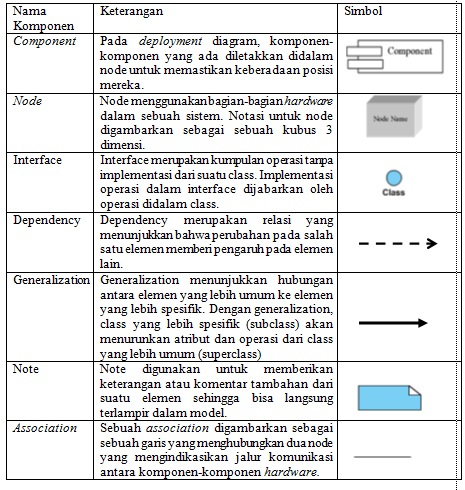


Gambar 10 Notasi Sequence Diagram

### 2.4.7 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



## 2.5 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.

## 2.6 Web Development (CI)

CI atau kepanjangannya Code Igniter adalah “sebuah framework php yang bersifat open source dan menggunakan metode MVC (Model,View,Controller) [12].” Code igniter ini berfungsi memudahkan developer membangun web tidak dari 0 lagi karna semua code dan syntaxnya sudah tergenerate dari sistem itu sendiri.Metode MVC ini maksudnya memisahkan model,view dan controller yang dimana memiliki fungsi masing-masing.Model ini berhubungan dengan pengolahan input/database,View berhubungan dengan user interface sementara controller sendiri berfungsi untuk mengolah intruksi-intruksi.

## 2.7 Firebase

Firebase merupakan platform yang akhir-akhir ini menjadi trend bagi developer dikarenakan banyaknya fitur dan kelebihan yang disediakan firebase sendiri.”Firebase ini memiliki berapa kelebihan seperti cepat dan responsive,tersedia versi free,user friendly,api relative mudah dimengerti,realtime back end dan banyak lagi”. [14]

## 2.8 Web Service (Json)

JSON (dilafalkan "Jason"), singkatan dari JavaScript Object Notation (bahasa Indonesia: notasi objek JavaScript), adalah suatu format ringkas pertukaran data komputer. Formatnya berbasis teks dan terbaca-manusia serta digunakan untuk merepresentasikan struktur data sederhana dan larik asosiatif (disebut objek). [15]

## 2.9 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 11 Bisnis Model Kanvas

## 2.10 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]

## 2.11 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 12 Pengujian Acceptance

# KEBUTUHAN PERANGKAT LUNAK

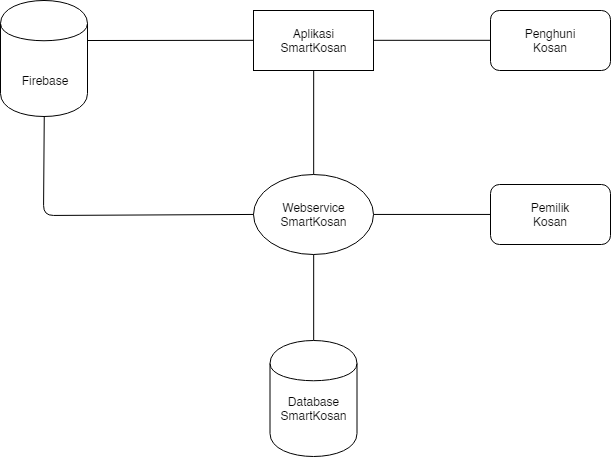
Dalam pembuatan perangkat lunak pastinya diperlukan kebutuhan yang akan digunakan dala pembangunan perangkat lunak itu sendiri dan disini akan dibahas kebutuhan-kebutuhan yang akan digunakan dalam pembangunan perangkat lunak.

## 3.1 Analisis Sistem Berjalan

1. Ketika penghuni kosan ingin melakukan pembayaran kosan baik kamar, internet maupun yang lainnya dan ingin mengetahui informasi mengenai kamar yang kosong;
2. Mereka akan menemui pemilik kosan untuk membayar maupun menanyakan informasi;
3. Pemilik kosan ingin memberi pengumuman mengenai tengat watku pembayaran;
4. Pemilik kosan menemui penghuni kosan untuk memberikan informasi mengani tengat waktu;

## 3.2 Analisis Arsitektur Sistem

Berikut ini merupakan gambar analisis arsitektur SmartKosan App berbasis Mobile (Andorid) dan memiliki server berbasis Website



Gambar 13 Arsitektur Sistem

Gambaran arsitektur sistem yang akan dibangun dimana penghuni kosan mengakses aplikasi Smart Kosan melalui smarthphone yang dimana data tersebut diambil melalu webserviceyang berasal dari database.Disini pemilik kosan berperan sebagai admin untuk mengelola informasi informasi terbaru yang ada yang kemudian di update ke database.Firebase disini digunakan untuk fitur notifikasi saja dimana apabila ada data baru yang diupdate di webservice firebase merespon dan mengirimkan pesan notifikasi ke aplikasi.

## 3.3 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 SmartKosan App :

### Kebutuhan Fungsional.

Berikut ini merupakan tabel kebutuhan nonfungsional :

Tabel 3‑1 Kebutuhan Fungsional

|  |  |
| --- | --- |
| **Nomor Kode** | **Fungsi** |
| SMK-F-001 | Sistem menyediakan fasilitas bagi pengguna untuk melihat informasi terbaru dari pemilik kosan |
| SMK-F-002 | Sistem menyediakan fasilitas bagi pengguna untuk melihat tagihan kosan seperti air, kamar, listrik, keamanan, internet dan kebersihan |
| SMK-F-003 | Sistem menyediakan fasilitas bagi pengguna untuk membayar tagihan kosan seperti air, kamar, listrik, keamanan, internet dan kebersihan |
| SMK-F-004 | Sistem menyediakan fasilitas bagi pengguna untuk melihat histori pembayaran |
| SMK-F-005 | Sistem menyediakan fasilitas bagi pengguna untuk melihat daftar kamar yang kosong |
| SMK-F-006 | Sistem menyediakan fasilitas bagi pengguna untuk melihat informasi pribadi melalui profil |

### Kebutuhan Non Fungsional

Berikut ini merupakan tabel kebutuhan non fungsional :

Tabel 3‑2 Kebutuhan Non Fungsional

|  |  |
| --- | --- |
| **Nomor Kode** | **Fungsi** |
| SMK-NF-001 | User dari sistem yang dibangun sudah terbiasa menggunakan gadget |
| SMK-NF-002 | Aplikasi mobile hanya support di android saja |
| SMK-NF-003 | Web CMS admin sudah support di kebanyakan web browser terkenal seperti(Firefox,Chrome,Edge) |
| SMK-NF-004 | Web CMS admin sudah berbasis web dinamis |
| SMK-NF-005 | Sistem yang dibangun dihosting dengan bandwith yang unlimited |
| SMK-NF-006 | Sistem yang dibangun memerlukan koneksi internet untuk mengaksesnya |
| SMK-NF-007 | Sistem dijalankan dengan menggunakan akses internet unlimited |
| SMK-NF-008 | Dibutuhkan Admin yang berpengalaman mengurus dan mengolah data web |
| SMK-NF-009 | Sistem yang dibangun memerlukan hardware yang memenuhi spesifikasi standar minimum |
| SMK-NF-010 | Sistem yang dibangun menggunakan Code Igniter untuk web development |
| SMK-NF-011 | Sistem yang dibangun menggunakan Android Studio untuk Mobile development |
| SMK-NF-012 | Sistem yang dibangun menggunakan Firebase untuk notifikasi |
| SMK-NF-013 | Sistem Operasi yang digunakan untuk membangun perangkat lunak adalah windows |

## 3.4 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 SmartKosan App :

### Analisis Kebutuhan Perangkat Lunak

Berikut ini merupakan tabel kebutuhan perangkat lunak :

|  |  |
| --- | --- |
| **Nomor Kode** | **Fungsi** |
| SMK-NF-003 | Web CMS admin sudah support di kebanyakan web browser terkenal seperti (Firefox,Chrome,Edge) |
| SMK-NF-004 | Web CMS admin sudah berbasis web dinamis |
| SMK-NF-010 | Sistem yang dibangun menggunakan Code Igniter untuk web development |
| SMK-NF-011 | Sistem yang dibangun menggunakan Android Studio untuk Mobile development |
| SMK-NF-012 | Sistem yang dibangun menggunakan Firebase untuk notifikasi |
| SMK-NF-013 | Sistem Operasi yang digunakan untuk membangun perangkat lunak adalah windows |

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

Sistem operasi : Android Lolipop.

Database : SQL

**Kebutuhan Pembangunan Perangkat Lunak :**

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

1. Sistem Operasi Windows 10 64 bit
2. Web Browser yang digunakan Google Chrome,Mozilla dan Microsoft Edge
3. Mobile Development Menggunakan Android Studio
4. Webserver : XAMPP
5. Web Development menggunakan Code Igniter
6. Web Development menggunakan Code Igniter
7. Mobile Development Menggunakan Android Studio Notifikasi menggunakan Firebase

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

Berdasarkan SQU-NF-003, SQU-NF-004, SQU-NF-010, SQU-NF-011, SQU-NF-012, SQU-NF-013 antara kebutuhan perangkat lunak dan fakta yang ada di lingkungan sistem sudah memenuhi syarat kebutuhan.

### Analisis Kebutuhan Perangkat Keras

**Spesifikasi Kebutuhan NonFungsional Perangkat Keras:**

|  |  |
| --- | --- |
| **Nomor Kode** | **Fungsi** |
| SMK-NF-002 | Aplikasi mobile hanya support di android saja |
| SMK-NF-009 | Sistem yang dibangun memerlukan hardware yang memenuhi spesifikasi standar minimum |

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

|  |  |  |
| --- | --- | --- |
| **No** | **Item** | **Spesifikasi** |
|  | Processor |  |
|  | Hardisk |  |
|  | Memory |  |
|  | Mouse |  |
|  | Monitor |  |
|  | Keyboard |  |
|  | LAN |  |
|  | Smartphone(Oppo 1201) |  |

**Kebutuhan Minimal Perangkat Keras :**

1. Processor
2. Hardisk 250MB
3. Memory 4GB
4. Mouse dan Smartphone

**Kesimpulan :**

Berdasarkan SMK-NF-002, SMK-NF-009 antara fakta dan kebutuhan minimal sudah memenuhi kebutuhan.

### Analisis Kebutuhan Perangkat Pikir

### Analisis Kebutuhan Jaringan

Pada bagian analisis jaringan ini akan diidentifikasikan spesifikasi jaringan lokal yang ada maupun yang dibutuhkan dalam pembangunan perangkat lunak.Analisis jaringan hanya dilakukan apabila sistem dibangun dalam jaringan lokal/terbatas.

|  |  |
| --- | --- |
| **Nomor Kode** | **Fungsi** |
| SMK-NF-005 | Sistem yang dibangun dihosting dengan bandwith yang unlimited |
| SMK-NF-006 | Sistem yang dibangun memerlukan koneksi internet untuk mengaksesnya |
| SMK-NF-007 | Sistem dijalankan dengan menggunakan akses internet unlimited |

**Fakta Kebutuhan Jaringan :**

Didalam sistem yang berjalan koneksi internet sudah tersedia yang diperlukan adalah hosting untuk pengoperasian website dan aplikasi.Domain yang digunakan ialah dscunikom.com dengan subdomain sman14bdg.dscunikom.com di IDCloudHost dengan harga Rp45000 dengan bandwidth unlimited, CPU Single Core, dan RAM sebesar 512 MB.

**Kebutuhan Jaringan :**

1. Koneksi Internet ada dan Unlimited
2. Bandwith yang diperlukan Unlimited

**Kesimpulan :**

Berdasarkan SMK-NF-005,SMK-NF-006,SMK-NF-007 fakta kebutuhan jaringan dan Kebutuhan jaringan sudah terpenuhi.

## 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** |
| --- | --- |
|  |  |

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

### Diagram Use Case

A close up of a map

Description automatically generated

### Definisi Actor

Tabel 1.3 *Daftar Aktor*

| **NO** | **AKTOR** | **DESKRIPSI** |
| --- | --- | --- |
| *1* | *Users* | *Aktor ini bisa mengakses aplikasi dengan cara terlebih dahulu di daftarkan oleh admin.* |
| *2* | *Admin* | *Actor ini berperan sebagai admin dikarenakan mereka yang melakukan registrasi bagi users,memberikan informasi terkait konfirmasi pembayaran,report,dan memberikan notifikasi kepada users.* |
| *3* | *Web Service* | *Actor ini berperan untuk mengkomunikasikan mobile apss sendiri dengan cms* |

### Definisi Use Case

Tabel 1.2 *Daftar Use case*

| **NO** | **USE CASE** | **DESKRIPSI** |
| --- | --- | --- |
|  | *Penyajian Info user* | Fungsionalitas ini digunakan admin untuk melihat informasi tentang user |
|  | *Penyajian Info kamar* | Fungsionalitas ini digunakan admin untuk melihat daftar kamar |
|  | *Penyajian Info fasilitas* | Fungsionalitas ini digunakan admin untuk melihat daftar fasilitas yang ada |
|  | *pembayaran* | Fungsionalitas ini digunakan admin untuk melakukan konfirmasi pembayaran |
|  | *notifikasi* | Fungsionalitas ini digunakan admin untuk memberikan notifikasi kepada user |
|  | *Informasi daftar akun* | sistem akan melakukan proses autentifikasi pada data user memvalidasi data tersebut ke web service |
|  | *Autentifikasi data user* | Fungsionalitas ini digunakan user untuk memverifikasi data user |
|  | *History pembayaran* | Fungsionalitas ini digunakan user untuk melihat history pembayaran yang telah di lakukan |
|  | *Lihat daftar kamar* | Fungsionalitas ini digunakan user untuk melihat daftar fasilitas yang tersedia di kamar tersebut |
|  | *Login ke sistem* | Fungsionalitas ini digunakan user untuk masuk kedalam aplikasi |
|  | *Membuat akun* | Fungsionalitas ini digunakan user untuk membuat akun agar bias masuk kedalam aplikasi |
|  | *Lihat fasilitas* | Fungsionalitas ini digunakan user untuk melihat fasilitas yang ada dalam kamar |
|  | *Ubah foto* | Fungsionalitas ini digunakan user untuk mengubah foto profil user |
|  | *Ubah password* | Fungsionalitas ini digunakan user untuk mengubah password |
|  | *Hapus akun* | Fungsionalitas ini digunakan user untuk menonaktifkan akun |
|  | *Tambah user* | Fungsionalitas ini digunakan admin untuk menambahkan user |
|  | *Hapus user* | Fungsionalitas ini digunakan admin untuk menonaktifkan akun user |
|  | *Ubah user* | Fungsionalitas ini digunakan admin untuk mengubah informasi user |
|  | *Hapus kamar* | Fungsionalitas ini digunakan admin untuk menghapus data kamar |
|  | *Edit kamar* | Fungsionalitas ini digunakan admin untuk mengubah data kamar |
|  | *Tambah kamar* | Fungsionalitas ini digunakan admin untuk menambahkan kamar |
|  | *Tambah fasilitas* | Fungsionalitas ini digunakan admin untuk menambahkan fasilitas dalam kamar |
|  | *Ubah fasilitas* | Fungsionalitas ini digunakan admin untuk mengubah fasilitas yang tersedia di kamar |
|  | *Hapus fasilitas* | Fungsionalitas ini digunakan ketika admin ingin menghapus fasilitas yang sudah tersedia |

### Skenario Use Case

Tabel 1.3 *Usecase* Scenario penyajian info user

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Penyajian Info user | |
| **Related Requirement** |  | |
| Goal In Context | Mendapatkan List Info user | |
| Description | Fungsionalitas ini digunakan user untuk melihat daftar informasi yang ada | |
| Related Use Case | Penyajian Info data user | |
| Preconditions | Terhubung ke jaringan internet, Sudah berada di dalam beranda aplikasi | |
| Successful End Condition | user bisa melihat data user | |
| Failed End Condition | user tidak mendapatkan data user | |
| Primary Actors | user | |
| Secondary Actors | Web Service | |
| Trigger | - | |
| Main Flow | Step | Action |
|  | 1. | user memilih menu info user |
|  | 2.  Include::Request data user | Sistem akan melakukan permintaan data ke web service |
|  | 3. | Sistem menampilkan halaman yang berisi list data user |
|  | 4. | user memilih untuk menampilkan data user |
| Extension | Step | Branching Action |
|  | 3.1 | Sistem menampikan toast tidak ada koneksi internet |
|  | 3.2 | Sistem menampilkan halaman data kosong karena tidak ada data |
|  | **4.1** | **Sistem menyajikan info data user** |

Tabel 1.4 *Usecase* Scenario penyajian info kamar

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Penyajian Info kamar | |
| **Related Requirement** |  | |
| Goal In Context | Mendapatkan List Info kamar | |
| Description | Fungsionalitas ini digunakan admin untuk melihat daftar kamar | |
| Related Use Case | Penyajian Info data kamar | |
| Preconditions | Terhubung ke jaringan internet, Sudah berada di dalam beranda aplikasi | |
| Successful End Condition | admin bisa melihat data kamar | |
| Failed End Condition | admin tidak mendapatkan data kamar | |
| Primary Actors | admin | |
| Secondary Actors | Web Service | |
| Trigger | admin memilih fungsionalitas fasilitas | |
| Main Flow | Step | Action |
|  | 1. | admin memilih menu info kamar |
|  | 2.  Include::Request data kamar | Sistem akan melakukan permintaan data ke web service |
|  | 3. | Sistem menampilkan halaman yang berisi list data kamar |
|  | 4. | admin memilih untuk menampilkan detail kamar |
| Extension | Step | Branching Action |
|  | 3.1 | Sistem menampikan toast tidak ada koneksi internet |
|  | 3.2 | Sistem menampilkan halaman data kosong karena tidak ada data |
|  | **4.1** | **Sistem menyajikan info data kamar** |

Tabel 1.4 *Usecase Scenario penyajian info fasilitas*

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Penyajian Info fasilitas | |
| **Related Requirement** |  | |
| Goal In Context | Mendapatkan List Info fasilitas | |
| Description | Fungsionalitas ini digunakan admin untuk melihat daftar fasilitas yang tersedia | |
| Related Use Case | Penyajian Info data fasilitas | |
| Preconditions | Terhubung ke jaringan internet, Sudah berada di dalam beranda aplikasi | |
| Successful End Condition | admin bisa melihat data fasilitas | |
| Failed End Condition | admin tidak mendapatkan data fasilitas | |
| Primary Actors | user | |
| Secondary Actors | Web Service | |
| Trigger | admin memilih fungsionalitas fasilitas | |
| Main Flow | Step | Action |
|  | 1. | admin memilih menu info kamar |
|  | 2.  Include::Request data fasilitas | Sistem akan melakukan permintaan data ke web service |
|  | 3. | Sistem menampilkan halaman yang berisi list data fasilitas |
|  | 4. | admin memilih untuk menampilkan detail kamar |
| Extension | Step | Branching Action |
|  | 3.1 | Sistem menampikan toast tidak ada koneksi internet |
|  | 3.2 | Sistem menampilkan halaman data kosong karena tidak ada data |
|  | **4.1** | **Sistem menyajikan info data fasilitas** |

Tabel 1.5 *Usecase Scenario Pembayaran*

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | pembayaran | |
| **Related Requirement** |  | |
| Goal In Context | Mendapatkan List pembayaran | |
| Description | Fungsionalitas ini digunakan admin untuk melakukan konfirmasi pembayaran | |
| Related Use Case | Penyajian Info data pembayaran | |
| Preconditions | Terhubung ke jaringan internet, Sudah berada di dalam beranda aplikasi | |
| Successful End Condition | admin bisa melihat data pembayaran | |
| Failed End Condition | admin tidak mendapatkan data pemabayaran | |
| Primary Actors | admin | |
| Secondary Actors | Web Service | |
| Trigger | admin memilih fungsionalitas pembayaran | |
| Main Flow | Step | Action |
|  | 1. | admin memilih menu pembayaran |
|  | 2.  Include::Request pembayaran | Sistem akan melakukan permintaan data ke web service |
|  | 3. | Sistem menampilkan halaman yang berisi list data pembayaran |
|  | 4. | admin memilih untuk menampilkan data pembayaran |
| Extension | Step | Branching Action |
|  | 3.1 | Sistem menampikan toast tidak ada koneksi internet |
|  | 3.2 | Sistem menampilkan halaman data kosong karena tidak ada data |
|  | **4.1** | **Sistem menyajikan info data pembayaran** |

Tabel 1.6 *Usecase Scenario notifikasi*

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | notifikasi | |
| **Related Requirement** |  | |
| Goal In Context | Mendapatkan notifikasi | |
| Description | Fungsionalitas ini digunakan admin untuk memberikan notifikasi kepada user | |
| Related Use Case | Penyajian notifikasi kepada user | |
| Preconditions | Terhubung ke jaringan internet, Sudah berada di dalam beranda aplikasi | |
| Successful End Condition | admin bisa melakukan push notifikasi kepada user | |
| Failed End Condition | admin tidak bisa melakukan push notifikasi kepada user | |
| Primary Actors | admin | |
| Secondary Actors | Web Service | |
| Trigger | admin memilih fungsionalitas notifikasi | |
| Main Flow | Step | Action |
|  | 1. | admin memilih menu notifikasi |
|  | 2.  Include::Request notifikasi | Sistem akan melakukan permintaan data ke web service |
|  | 3. | Sistem menampilkan halaman yang berisi notifikasi |
|  | 4. | admin memilih untuk menampilkan halaman notifikasi |
| Extension | Step | Branching Action |
|  | 3.1 | Sistem menampikan toast tidak ada koneksi internet |
|  | 3.2 | Sistem menampilkan halaman data kosong karena tidak ada data |
|  | **4.1** | **Sistem menyajikan notifikasi** |

Tabel 1.7 *Usecase Scenario info daftar akun*

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Informasi daftar akun | |
| **Related Requirement** |  | |
| Goal In Context | Mendapatkan data akun | |
| Description | Fungsionalitas ini digunakan user untuk memverifikasi data user | |
| Related Use Case | Informasi daftar akun | |
| Preconditions | Terhubung ke jaringan internet, Sudah berada di dalam beranda aplikasi | |
| Successful End Condition | User dapat melihat data akun | |
| Failed End Condition | user tidak bisa melihat data akun | |
| Primary Actors | user | |
| Secondary Actors | Web Service | |
| Trigger | user memilih fungsionalitas informasi daftar akun | |
| Main Flow | Step | Action |
|  | 1. | user memilih menu informasi data akun |
|  | 2.  Include::Request info data akun | Sistem akan melakukan permintaan data ke web service |
|  | 3. | Sistem menampilkan halaman yang berisi informasi data akun |
|  | 4. | user memilih untuk menampilkan halaman informasi data akun |
| Extension | Step | Branching Action |
|  | 3.1 | Sistem menampikan toast tidak ada koneksi internet |
|  | 3.2 | Sistem menampilkan halaman data kosong karena tidak ada data |
|  | **4.1** | **Sistem menyajikan info data akun** |

Tabel 1.8 *Usecase Scenario autentifikasi data user*

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Autentifikasi data user | |
| **Related Requirement** |  | |
| Goal In Context | Mendapatkan autentifikasi data user | |
| Description | sistem akan melakukan proses autentifikasi pada data user memvalidasi data tersebut ke web service | |
| Related Use Case | Autentifikasi data user | |
| Preconditions | Terhubung ke jaringan internet, Sudah berada di dalam beranda aplikasi | |
| Successful End Condition | sistem dapat meverifikasi data user | |
| Failed End Condition | user tidak bisa memverifikasi data user | |
| Primary Actors | user | |
| Secondary Actors | Web Service | |
| Trigger | user memilih fungsionalitas login | |
| Main Flow | Step | Action |
|  | 1. | user memilih menu login |
|  | 2.  Include::Request autentifikasi data user | Sistem akan melakukan permintaan data ke web service |
|  | 3. | Sistem menampilkan halaman login |
|  | 4. | user memilih untuk menampilkan halaman login |
| Extension | Step | Branching Action |
|  | 3.1 | Sistem menampikan toast tidak ada koneksi internet |
|  | 3.2 | Sistem menampilkan halaman data kosong karena tidak ada data |
|  | **4.1** | **Sistem menutentifikasi data user** |

Tabel 1.8 *Usecase Scenario history pembayaran*

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | History pembayaran | |
| **Related Requirement** |  | |
| Goal In Context | Dapat menampilkan history pembayaran | |
| Description | Fungsionalitas ini digunakan user untuk melihat history pembayaran yang telah di lakukan | |
| Related Use Case | History pembayaran | |
| Preconditions | Terhubung ke jaringan internet, Sudah berada di dalam beranda aplikasi | |
| Successful End Condition | sistem dapat menampilkan history pembayaran | |
| Failed End Condition | sistem tidak bisa menampilkan history pembayaran | |
| Primary Actors | user | |
| Secondary Actors | Web Service | |
| Trigger | user memilih fungsionalitas history pembayaran | |
| Main Flow | Step | Action |
|  | 1. | user memilih menu pembayaran |
|  | 2.  Include::Request history pembayaran | Sistem akan melakukan permintaan data ke web service |
|  | 3. | Sistem menampilkan halaman history pembayaran |
|  | 4. | user memilih untuk menampilkan halaman pembayaran |
| Extension | Step | Branching Action |
|  | 3.1 | Sistem menampikan toast tidak ada koneksi internet |
|  | 3.2 | Sistem menampilkan halaman data kosong karena tidak ada data |
|  | **4.1** | **Sistem menampilkan history pembayaran** |

Tabel 1.9 *Usecase Scenario lihat daftar kamar*

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Lihat daftar kamar | |
| **Related Requirement** |  | |
| Goal In Context | Dapat menampilkan list daftar kamar | |
| Description | Fungsionalitas ini digunakan user untuk melihat daftar fasilitas yang tersedia di kamar tersebut | |
| Related Use Case | Lihat daftar kamar | |
| Preconditions | Terhubung ke jaringan internet, Sudah berada di dalam beranda aplikasi | |
| Successful End Condition | sistem dapat menampilkan list daftar kamar | |
| Failed End Condition | sistem tidak bisa menampilkan daftar kamar | |
| Primary Actors | user | |
| Secondary Actors | Web Service | |
| Trigger | user memilih fungsionalitas lihat daftar kamar | |
| Main Flow | Step | Action |
|  | 1. | user memilih menu lihat daftar kamar |
|  | 2.  Include::Request lihat daftar kamar | Sistem akan melakukan permintaan data ke web service |
|  | 3. | Sistem menampilkan halaman lihat daftar kamar |
|  | 4. | user memilih untuk menampilkan lihat daftar kamar |
| Extension | Step | Branching Action |
|  | 3.1 | Sistem menampikan toast tidak ada koneksi internet |
|  | 3.2 | Sistem menampilkan halaman data kosong karena tidak ada data |
|  | **4.1** | **Sistem menampilkan lihat daftar kamar** |

Tabel 1.10 *Usecase Scenario login ke sistem*

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Login ke sistem | |
| **Related Requirement** |  | |
| Goal In Context | Dapat masuk ke halaman utama aplikasi | |
| Description | Fungsionalitas ini digunakan user untuk masuk kedalam aplikasi | |
| Related Use Case | Login ke sistem | |
| Preconditions | Terhubung ke jaringan internet, Sudah berada di dalam beranda aplikasi | |
| Successful End Condition | sistem dapat manampilkan halaman utama | |
| Failed End Condition | sistem tidak bisa menampilkan halaman utama | |
| Primary Actors | user | |
| Secondary Actors | Web Service | |
| Trigger | user memilih fungsionalitas login ke sistem | |
| Main Flow | Step | Action |
|  | 1. | user memilih menu login ke sistem |
|  | 2.  Include::Request login ke sistem | Sistem akan melakukan permintaan data ke web service |
|  | 3. | Sistem menampilkan halaman utama |
|  | 4. | user memilih untuk menampilkan halaman utama |
| Extension | Step | Branching Action |
|  | 3.1 | Sistem menampikan toast tidak ada koneksi internet |
|  | 3.2 | Sistem menampilkan halaman data kosong karena tidak ada data |
|  | **4.1** | **Sistem menampilkan login ke sistem** |

Tabel 1.11 *Usecase Scenario membuat akun*

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Membuat akun | |
| **Related Requirement** |  | |
| Goal In Context | Dapat membuat akun | |
| Description | Fungsionalitas ini digunakan user untuk membuat akun agar bisa masuk kedalam aplikasi | |
| Related Use Case | Membuat akun | |
| Preconditions | Terhubung ke jaringan internet, Sudah berada di dalam beranda aplikasi | |
| Successful End Condition | sistem dapat membuat akun | |
| Failed End Condition | sistem tidak bisa membuat akun | |
| Primary Actors | user | |
| Secondary Actors | Web Service | |
| Trigger | user memilih fungsionalitas membuat akun | |
| Main Flow | Step | Action |
|  | 1. | user memilih menu membuat akun |
|  | 2.  Include::Request membuat akun | Sistem akan melakukan permintaan data ke web service |
|  | 3. | Sistem menampilkan halaman membuat akun |
|  | 4. | user memilih untuk menampilkan halaman membuat akun |
| Extension | Step | Branching Action |
|  | 3.1 | Sistem menampikan toast tidak ada koneksi internet |
|  | 3.2 | Sistem menampilkan halaman data kosong karena tidak ada data |
|  | **4.1** | **Sistem menampilkan membuat halaman akun** |

Tabel 1.12 *Usecase Scenario lihat fasilitas*

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Lihat fasilitas | |
| **Related Requirement** |  | |
| Goal In Context | Dapat melihat halaman lihat fasilitas | |
| Description | Fungsionalitas ini digunakan user untuk melihat fasilitas yang ada dalam kamar | |
| Related Use Case | Lihat fasilitas | |
| Preconditions | Terhubung ke jaringan internet, Sudah berada di dalam beranda aplikasi | |
| Successful End Condition | sistem dapat menampilkan halaman lihat fasilitas | |
| Failed End Condition | sistem tidak bisa menampilkan halaman fasilitas | |
| Primary Actors | user | |
| Secondary Actors | Web Service | |
| Trigger | user memilih fungsionalitas lihat daftar kamar | |
| Main Flow | Step | Action |
|  | 1. | user memilih menu daftar kamar |
|  | 2.  Include::Request lihat fasilitas | Sistem akan melakukan permintaan data ke web service |
|  | 3. | Sistem menampilkan halaman lihat fasilitas |
|  | 4. | user memilih untuk menampilkan halaman fasilitas |
| Extension | Step | Branching Action |
|  | 3.1 | Sistem menampikan toast tidak ada koneksi internet |
|  | 3.2 | Sistem menampilkan halaman data kosong karena tidak ada data |
|  | **4.1** | **Sistem menampilkan halaman fasilitas** |

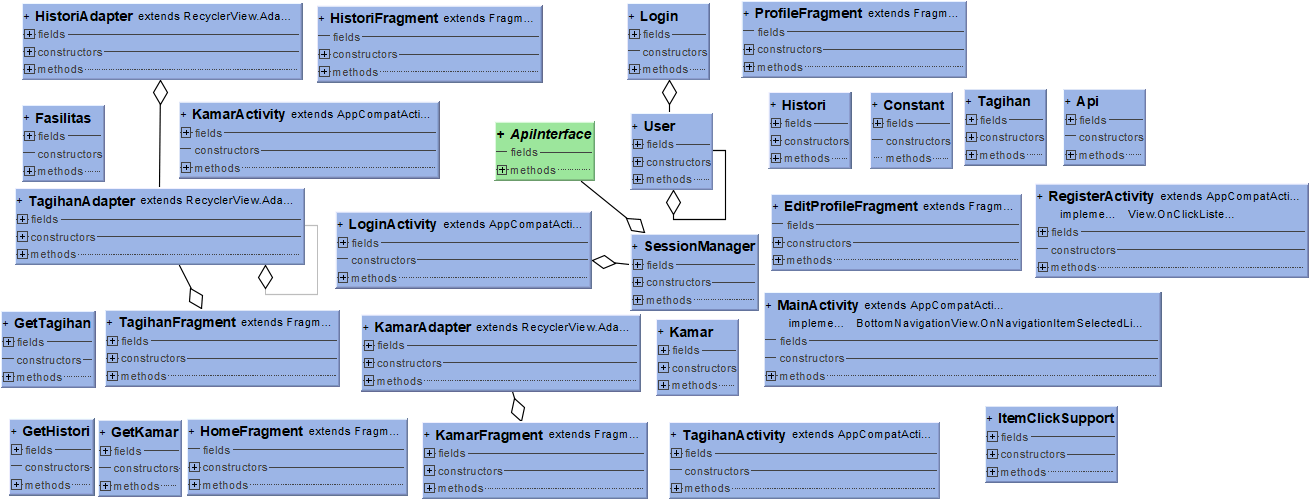
## Model Activity Diagram

A screenshot of a cell phone

Description automatically generated

# MODEL ANALISIS

## Diagram Class Analisis



Gambar 14 Sequance diagram

## Identifikasi Kelas Analisis

|  |  |  |
| --- | --- | --- |
| Nama Kelas | Daftar Tanggung-Jawab (Method) | Daftar Atribut |
| KamarActivity |  |  |
| LoginActivity |  |  |
| MainActivity |  |  |
| RegisterActivity |  |  |
| TagihanActivity |  |  |

## Realisasi Kelas Terhadap Use Case

|  |  |  |
| --- | --- | --- |
| **No** | **Usecase** | **Realisasi Kelas** |
|  | *Penyajian Info Acara* | Acara,Siswa,Sekolah |
|  | *Penyajian Info Prestasi* | Prestasi,Siswa,Sekolah |
|  | *Penyajian Info Berita* | Berita,Siswa,Sekolah |

### Identifikasi Kelas Analisis tiap Paket

|  |  |  |  |
| --- | --- | --- | --- |
| No | Nama Paket | Nama Kelas Analisis | Jenis Kelas |
|  | Sekolah | 1. Acara 2. Prestasi 3. Berita 4. Fasilitas 5. Ekskul 6. Profil Sekolah 7. Kalender Akademik | 1. Entity 2. Entity 3. Entity 4. Entity 5. Entity 6. Entity 7. Entity |
|  | Control | 1. Admin 2. Siswa 3. Favorit 4. Search 5. Login | 1. Control 2. Control 3. Control 4. Control 5. Control |

## Analisis Data

A screenshot of a social media post

Description automatically generated

Gambar 15 Database Web Service

# 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.

## Perancangan Detil Elemen Logical View

## Diagram Kelas Keseluruhan

## Algoritma / Query

### KamarActivity

Nama Kelas : KamarActivity

Nama Operasi : onCreate dan getData

Algoritma :

|  |
| --- |
|  |

### LoginActivity

Nama kelas : LoginActivity

Nama operasi : onCreate dan login

Algoritma :

|  |
| --- |
|  |

### MainActivity

Nama kelas : MainActivity

Nama operasi : onCreate, onNavigationItemSelected, onCreateOptionsMenu, loadFragment, onClick dan card

Algoritma :

|  |
| --- |
|  |

### RegisterActivity

Nama kelas : RegisterActivity

Nama operasi : onCreate, userSignUp dan onClick

Algoritma :

|  |
| --- |
|  |

### TagihanActivity

Nama kelas : TagihanActivity

Nama operasi : onCreate dan getData

Algoritma :

|  |
| --- |
|  |

### EditProfileFragment

Nama kelas : EditProfileFragment

Nama operasi : EditProfileFragment dan onCreateView

Algoritma :

|  |
| --- |
|  |

### HistoriFragment

Nama kelas : HistoriFragment

Nama operasi : onCreate dan onCreateView

Algoritma :

|  |
| --- |
|  |

### HomeFragment

Nama kelas : HomeFragment

Nama operasi : onCreate dan onCreateView

Algoritma :

|  |
| --- |
|  |

### KamarFragment

Nama kelas : KamarFragment

Nama operasi : onCreate, onCreateView, clickItemDetail, getDate, reloadView

Algoritma :

|  |
| --- |
|  |

### ProfileFragment

Nama kelas : ProfileFragment

Nama operasi : onCreate dan onCreateView

Algoritma :

|  |
| --- |
|  |

### TagihanFragment

Nama kelas : TagihanFragment

Nama operasi : onCreate, onCreateView dan getData

Algoritma :

|  |  |
| --- | --- |
| |  | | --- | |  | |

## Perancangan Data

Tabel 5‑1 *Tabel User*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nama Field** | **Tipe Data** | **Panjang** | **Kunci** | **Keterangan** |
| Id\_user | varchar | 16 | Primary Key | Auto\_Increment,  Not\_null |
| Nama | Varchar | 50 |  | Not null |
| Password | varchar | 100 |  | Not null |
| jk | Enum | L,P |  | Not null |
| no\_hp | varchar | 12 |  | Not null |
| email | Varchar | 35 |  | Not null |
| alamat | varchar | 35 |  | Not null |
| status | tinyint | 1 |  | Not null |
| akses | enum | Admin,User |  | Not null |

Tabel 5‑2 *Tabel Pembayaran*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nama Field** | **Tipe Data** | **Panjang** | **Kunci** | **Keterangan** |
| Id\_pembayaran | int | 11 | Primary Key | Auto\_Increment,  Not\_null |
| Id\_user | Varchar | 16 | Foreign Key references Tabel user(id\_user) | Not null |
| jenis | enum | Cash, transfer |  | Not null |
| bukti | varchar | 100 |  | Not null |
| Tgl\_bayar | date |  |  | Not null |
| status | tinyint | 1 |  | Not null |
| hapus | tinyint | 1 |  | Not null |

Tabel 5‑3 *Tabel Kamar*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nama Field** | **Tipe Data** | **Panjang** | **Kunci** | **Keterangan** |
| Id\_kamar | int | 11 | Primary Key | Auto\_Increment,  Not\_null |
| Kode\_kamar | Varchar | 10 |  | Not null |
| harga | int | 11 |  | Not null |
| Status | tinyint | 1 |  | Not null |
| Id\_user | varchar | 16 | Foreign Key references Tabel user(id\_user) | Not null |
| foto | varchar | 100 |  | Not null |

Tabel 5‑4 *Tabel Detail Kamar*

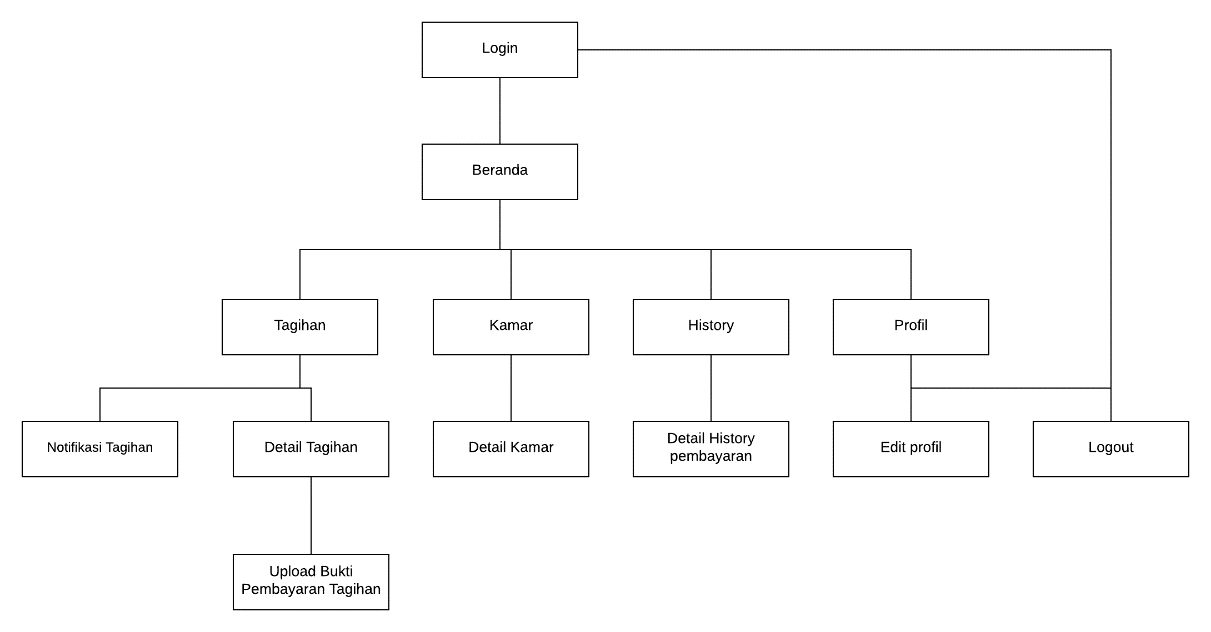
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nama Field** | **Tipe Data** | **Panjang** | **Kunci** | **Keterangan** |
| Id | int | 11 | Primary Key | Auto\_Increment,  Not\_null |
| Id\_kamar | int | 11 | Foreign Key references Tabel Kamar (id\_kamar) | Not null |
| Id\_fasiilitas | int | 11 | Foreign Key references Tabel Fasilitas (id\_fasilitas) | Not null |

Tabel 5‑4 *Tabel Fasilitas*

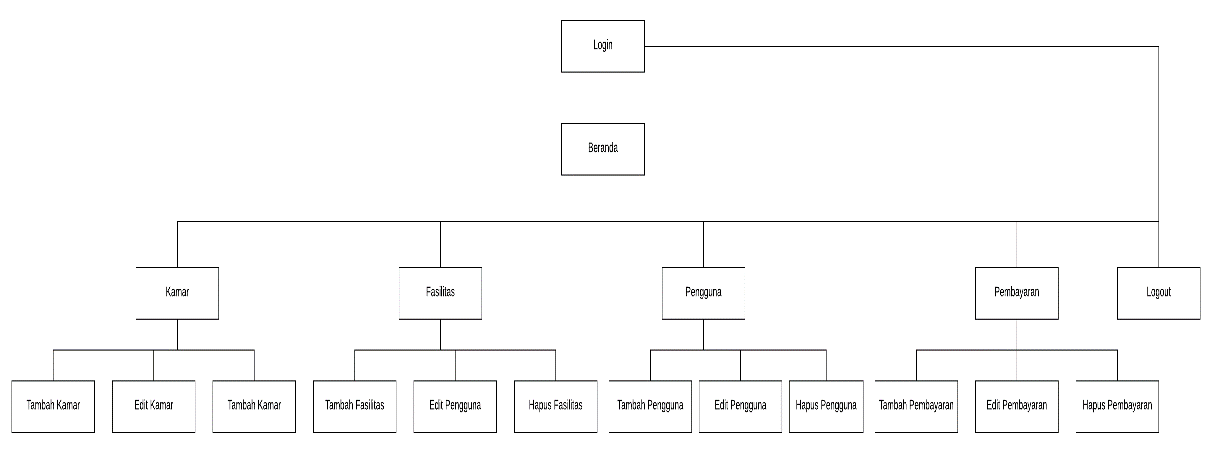
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nama Field** | **Tipe Data** | **Panjang** | **Kunci** | **Keterangan** |
| Id\_fasilitas | int | 11 | Primary Key | Auto\_Increment,  Not\_null |
| nama | varchar | 20 |  | Not null |

## Perancangan Arsitektural Menu

### Perancangan Antarmuka Menu User

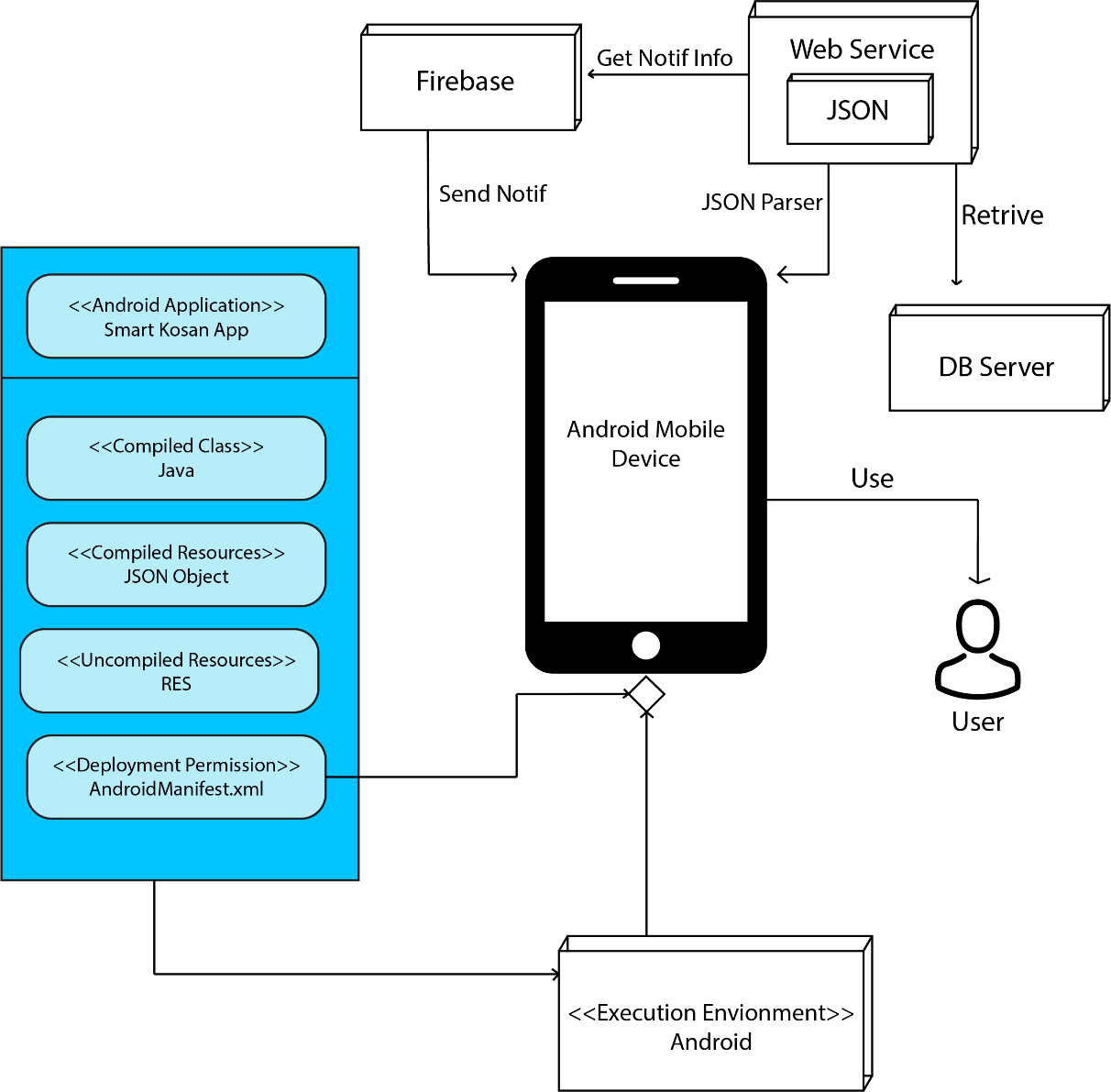


### Perancangan Antarmuka Menu Admin



## Perancangan Antarmuka

## Deployment Diagram



# Implementasi

## Lingkungan Implementasi

1. Lingkungan Hardware Pembanguan Perangkat Lunak Android :

|  |  |  |
| --- | --- | --- |
| No | Item | Spesifikasi |
| 1 | Processor |  |
| 2 | Hardisk |  |
| 3 | Memory |  |
| 4 | Smart Phone |  |

1. Lingkungan Software

Perangkat lunak yang digunakan dalam pembangunan perangkat ini

1. Sistem Operasi Windows 10 64 bit
2. Bahasa pemrograman PHP dan Java
3. DBMS MYSQL
4. Web Browser
5. Code Editor
6. Webserver : XAMPP
7. Pertukaran Data menggunakan JSON
8. Mobile Development menggunakan Android Studio
9. Web Development menggunakan Code Igniter
10. Notifikasi menggunakan Firebase

## Implementasi Data

* + - 1. Pembuatan Database

|  |
| --- |
| CREATE DATABASE db\_kosan; |

* + - 1. Pembuatan Tabel Fasilitas

|  |
| --- |
| CREATE TABLE `t\_fasilitas` (  `id\_fasilitas` int(11) NOT NULL,  `nama` varchar(20) NOT NULL  ); |

* + - 1. Pembuatan Tabel Kamar

|  |
| --- |
| CREATE TABLE `t\_kamar` (  `id\_kamar` int(11) NOT NULL,  `kode\_kamar` varchar(10) NOT NULL,  `harga` int(11) NOT NULL,  `status` tinyint(1) NOT NULL DEFAULT '0',  `foto` varchar(100) DEFAULT NULL,  `id\_user` varchar(16) DEFAULT NULL  FOREIGN\_KEY (‘id\_user’) REFERENCES ‘t\_user’ (‘id\_user’)  ); |

* + - 1. Pembuatan Tabel Detail Kamar

|  |
| --- |
| CREATE TABLE `t\_kamar\_detail` (  `id` int(11) NOT NULL,  `id\_kamar` int(11) NOT NULL,  `id\_fasilitas` int(11) NOT NULL,  FOREIGN\_KEY (‘id\_kamar’) REFERENCES ‘t\_kamar’ (‘id\_kamar’)  ); |

* + - 1. Pembuatan Tabel Pembayaran

|  |
| --- |
| CREATE TABLE `t\_pembayaran` (  `id\_pembayaran` int(11) NOT NULL,  `id\_user` varchar(16) NOT NULL,  `jenis` enum('cash','transfer') NOT NULL,  `bukti` varchar(100) NOT NULL,  `tgl\_bayar` date NOT NULL,  `status` tinyint(1) NOT NULL DEFAULT '0',  `hapus` tinyint(1) NOT NULL DEFAULT '0',  FOREIGN\_KEY (‘id\_user’) REFERENCES ‘t\_user’ (‘id\_user’)  ); |

* + - 1. Pembuatan Tabel User

|  |
| --- |
| CREATE TABLE `t\_user` (  `id\_user` varchar(16) NOT NULL,  `nama` varchar(50) NOT NULL,  `password` varchar(100) NOT NULL,  `jk` enum('L','P') NOT NULL,  `no\_hp` varchar(12) NOT NULL,  `email` varchar(35) NOT NULL,  `alamat` text NOT NULL,  `status` tinyint(1) NOT NULL DEFAULT '1',  `akses` enum('admin','user') NOT NULL DEFAULT 'user',  ); |

## Implementasi Kelas

Berikut ini merupakan tabel implmentasi kelas

|  |  |  |  |
| --- | --- | --- | --- |
| No | Nama Kelas | Nama File Fisik | Nama File Executable |
| 1 | KamarActivity | KamarActivity.java | KamarActivity.java |
| 2 | LoginActivity | LoginActivity.java | LoginActivity.java |
| 3 | MainActivity | MainActivity.java | MainActivity.java |
| 4 | RegisterActivity | RegisterActivity.java | RegisterActivity.java |
| 5 | TagihanActivity | TagihanActivity.java | TagihanActivity.java |
| 6 | HistoriAdapter | HistoriAdapter.java | HistoriAdapter.java |
| 7 | KamarAdapter | KamarAdapter.java | KamarAdapter.java |
| 8 | TagihanAdapter | TagihanAdapter.java | TagihanAdapter.java |
| 9 | EditProfieFragment | EditProfieFragment.java | EditProfieFragment.java |
| 10 | HistoriFragment | HistoriFragment.java | HistoriFragment.java |
| 11 | HomeFragment | HomeFragment.java | HomeFragment.java |
| 12 | KamarFragment | KamarFragment.java | KamarFragment.java |
| 13 | ProfileFragment | ProfileFragment.java | ProfileFragment.java |
| 14 | TagihanFragment | TagihanFragment.java | TagihanFragment.java |
| 15 | GetHistori | GetHistori.java | GetHistori.java |
| 16 | GetKamar | GetKamar.java | GetKamar.java |
| 17 | GetTagihan | GetTagihan.java | GetTagihan.java |
| 18 | SharedPref | SharedPref.java | SharedPref.java |
| 19 | Fasilitas | Fasilitas.java | Fasilitas.java |
| 20 | Histori | Histori.java | Histori.java |
| 21 | Kamar | Kamar.java | Kamar.java |
| 22 | ResultRegis | ResultRegis.java | ResultRegis.java |
| 23 | Tagihan | Tagihan.java | Tagihan.java |
| 23 | User | User.java | User.java |
| 24 | Api | Api.java | Api.java |
| 25 | Constant | Constant.java | Constant.java |
| 26 | ItemClickSupport | ItemClickSupport.java | ItemClickSupport.java |
| 27 | SessionManager | SessionManager.java | SessionManager.java |

## Implementasi Antarmuka

|  |  |  |  |
| --- | --- | --- | --- |
| No | Nama Kelas | Nama File Fisik | Nama File Executable |
| 1 | Activity kamar | Activity\_kamar.xml | Activity\_kamar.xml |
| 2 | Activity Login | Activity\_login.xml | Activity\_login.xml |
| 3 | Activity Main | Activity\_main.xml | Activity\_main.xml |
| 4 | Activity register | Activity\_register.xml | Activity\_register.xml |
| 5 | Activity tagihan | Activity\_tagihan.xml | Activity\_tagihan.xml |
| 6 | Fragment Edit Profile | Fragment\_edit\_profile.xml | Fragment\_edit\_profile.xml |
| 7 | Fragment Histori | Fragment\_histori.xml | Fragment\_histori.xml |
| 8 | Fragment Home | Fragment\_home.xml | Fragment\_home.xml |
| 9 | Fragment Kamar | Fragment\_kamar.xml | Fragment\_kamar.xml |
| 10 | Fragment Profile | Fragment\_profile.xml | Fragment\_profile.xml |
| 11 | Fragment Tagihan | Fragment\_Tagihan.xml | HomeFragment.java |
| 12 | Item Histori | Item\_histori.xml | Item\_histori.xml |
| 13 | Item Kamar | Item\_kamar.xml | Item\_kamar.xml |
| 14 | Item Tagihan | Item\_tagihan.xml | Item\_tagihan.xml |
| 15 | List\_fasilitas | List\_fasiitas.xml | List\_fasilitas.xml |
| 16 | Dashboard | V\_Dashboard.php | V\_Dashboard.php |
| 17 | Fasilitas | V\_Fasilitas.php | V\_Fasilitas.php |
| 18 | Fasilitas Form Edit | V\_Fasilitas\_FormEdit.php | V\_Fasilitas\_FormEdit.php |
| 19 | Fasilitas Form Insert | V\_Fasilitas\_FormInsert.php | V\_Fasilitas\_FormInsert.php |
| 20 | Forget Pass Form | V\_Forget\_pass\_form.php | V\_Fasilitas\_FormInsert.php |
| 21 | Forget Password | V\_Forget\_Password.php | V\_Forget\_Password.php |
| 22 | Kamar | V\_Kamar.php | V\_Kamar.php |
| 23 | Kamar Form Edit | V\_Kamar\_FormEdit.php | V\_Kamar\_FormEdit.php |
| 23 | Kamar Form Insert | V\_Kamar\_FormInsert.php | V\_Kamar\_FormInsert.php |
| 24 | Login | V\_Login.php | V\_Login.php |
| 25 | Pembayaran | V\_Pembayaran.php | V\_Pembayaran.php |
| 26 | Pembayaran Form Edit | V\_Pembayaran\_FormEdit.php | V\_Pembayaran\_FormEdit.php |
| 27 | Pembayaran Form Insert | V\_Pembayaran\_FormInsert.php | V\_Pembayaran\_FormInsert.php |
| 28 | User | V\_User.php | V\_User.php |
| 29 | User Form Edit | V\_User\_FormEdit.php | V\_User\_FormEdit.php |
| 30 | User Form Insert | V\_User\_FormInsert.php | V\_User\_FormInsert.php |
| 31 | Welome Message | welcome\_message.php | welcome\_message.php |

# 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 2 buah subsistem, yaitu subsistem android dan web dengan jenis pengujian black box .

Tabel 22 Renca Pengujian Fungsional

|  |  |  |
| --- | --- | --- |
| No | Subsistem yang diuji | Jenis pengujian |
| 1 | Android | Black Box |
| 2 | Web Admin | Black Box |

1. Rencana pengujian acceptance

Rencana pengujian acceptance yang dilaukan adalah melakukan wawancara langsung kepada narasumber yakni pemilik kosan dan penghuni kosan 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 |  |  |  |
| 3 |  |  |  |
| 4 |  |  |  |

### 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 2 bagian, yaitu hasil pengujian subsistem android dan web admin.

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

|  |  |  |  |
| --- | --- | --- | --- |
| Kasus dan Hasil Uji (Data benar) | | | |
| Data Masukan | Hasil yang diharapkan | Pengamatan | Kesimpulan |
| Input : | *Filed -* terisi sesuai keinginan user | *Field -* terisi | Diterima |
|  |  |  | Diterima |
|  |  |  | Diterima |
|  |  |  | 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 |  | 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 |  | 3 | 6 |  |
| 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 SmartKosan Appdapat membantu pengguna untuk mengingat kapan harus membayar tagihan keperluan kosan.
2. Pengguna dapat melihat daftar kamar beserta fasilitas yang tersedia atau kosong.
3. Pengguna dapat melihat histori transaksi pembayaran yang pernah dilakukan sebelumnya.
4. Pengguna dapat mengedit informasi profil pribadi yang disediakan.
5. Pengguna dapat melakukan pembayaran tagihan kosan dengan mengirimkan bukti transaki pembayaran.

## Saran

Tugas besar yang berjudul “Dokumen Pembangunan Perangkat Lunak SmartKosan App” 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

# LAMPIRAN