

**ANALISIS DAN PERANCANGAN SISTEM INDEKOS
MENGUNAKAN METODE UNIFIED MODELING LANGUAGE (UML)**

Skripsi

untuk memenuhi sebagian persyaratan

mencapai derajat Sarjana S-1

Program Studi Teknik Informatika



diajukan oleh

Muhammad Fuad Adib

10650008

kepada

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS SAINS DAN TEKNOLOGI

UIN SUNAN KALIJAGA

YOGYAKARTA

2014

**ANALISIS DAN PERANCANGAN SISTEM INDEKOS
MENGUNAKAN METODE UNIFIED MODELING LANGUAGE (UML)**

PENGESAHAN SKRIPSI/TUGAS AKHIR

Nomor :

Skripsi/Tugas Akhir dengan judul : Analisis dan Perancangan Sistem Indekos Menggunakan Metode Unified Modeling Language (UML)

Yang dipersiapkan dan disusun oleh :

Nama : Muhammad Fuad Adib

NIM : 10650008

Telah dimunaqasyahkan pada :

Nilai Munaqasyah :

Dan dinyatakan telah diterima oleh Fakultas Sains dan Teknologi UIN Sunan Kalijaga

TIM MUNAQASYAH :

Ketua Sidang

NIP.

Penguji I

Penguji II

NIP.

NIP.

Yogyakarta,

UIN Sunan Kalijaga

Fakultas Sains dan Teknologi

DEKAN

NIP:

PERNYATAAN

Dengan ini saya menyatakan bahwa skripsi ini tidak terdapat karya yang pernah diajukan untuk memperoleh gelar kesarjanaan di suatu Perguruan Tinggi, dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka.

Yogyakarta,

Muhammad Fuad Adib

10650008

KATA PENGANTAR

Assalamu'alaikumWr. Wb.

Segala puji bagi Allah swt yang maha pengasih dan lagi maha penyayang, dengan segala kasih dan sayang-Nya sehingga penyusunan skripsi dengan judul “Analisis dan Perancangan Sistem Indekos Menggunakan Metode *Unified Modeling Language* (UML)” dapat berjalan sebagaimana mestinya. Shalawat dan salam kita tunjukkan kepada guru umat manusia, seorang revolusioner sejari, yaitu Nabi Muhammad SAW yang dalam sejarah kehidupannya, mengajarkan banyak hal kepada kita, salah satunya adalah kemampuan untuk tetap istiqomah berada di jalan-Nya.

Sebagai salah satu syarat memperoleh gelar kesarjanaan pada program studi Teknik Informatika, penyusunan skripsi adalah hal mutlak bagi setiap mahasiswa yang ingin mendapatkan gelar sarjana. Mendapatkan gelar sarjana adalah sesuatu hal yang tidak bisa dilepaskan dari niat awal seseorang kuliah, gelar sarjana juga adalah balasan kita, mahasiswa, terhadap segala bentuk pengorbanan yang diberikan orang tua kita, walaupun sejatinya itu tidak cukup.

Kalaupun harus didedikasikan, maka skripsi ini penulis dedikasikan kepada kedua orang tua yang sangat dicintai, Bapak dan Ibu (Fathullah dan Hamidah), yang tak henti – hentinya memberikan do’a kepada anak – anaknya agar kelak sukses dan berguna, sudah banyak pengorbanan yang mereka lakukan untuk semua anaknya, semoga Allah membalas jasa – jasa mereka Aamiin.

Kepada kakak (Kamal) dan adik – adik (Adil, Azha, Wardah dan Wafi) yang sangat saya sayangi, yang tak henti – hentinya memberikan motivasi. Selanjutnya penulis tidak lupa menghaturkan banyak terima kasih kepada semua pihak yang telah membantu dalam penyusunan baik secara langsung maupun tidak langsung. Sebagai rasa hormat dan ucapan terima kasih penyusun sampaikan kepada:

1. Bapak Prof. Dr. H. Akh. Minhaji, M.A., Ph.D., selaku Dekan Fakultas Sains dan Teknologi UIN Sunan Kalijaga Yogyakarta beserta para dosen dan seluruh karyawan/staf pegawai atas bantuan yang diberikan selama penulis mengikuti studi.
2. Bapak Agus Mulyanto, S.Si., M.Kom., selaku Ketua Program Studi Teknik Informatika UIN Sunan Kalijaga Yogyakarta.
3. Bapak Nurochman, S.Kom., M.Kom., selaku Dosen Pembimbing Akademik.
4. Bapak Sumarsono, ST., M.Kom., selaku Dosen Pembimbing yang telah memberikan arahan dan bimbingan selama penelitian.
5. Bapak dan Ibu Dosen TIF UIN SUKA yang telah mendidik dan mengajarkan tentang semua, terima kasih telah bersusah payah mendidik kami dengan sungguh – sungguh.
6. Teman – teman TIF UIN SUKA umumnya dan khususnya INFORMATICS ENGINEERING 2010 (MONSTER INFORMATICS) yang telah mengisi hari – hari selama perkuliahan.

Penulis menyadari masih banyak kekurangan dan kelemahan dalam penelitian ini. Oleh karena itu demi perkembangan penelitian selanjutnya penulis sangat

mengharap kritik dan saran dari pembaca. Akhirnya semoga penelitian ini bermanfaat bagi pembaca.

Wassalamu'alaikumWr. Wb

Yogyakarta, 08 Nopember 2013

Muhammad Fuad Adib

10650008

DAFTAR ISI

| | |
|--|-----|
| HALAMAN JUDUL..... | i |
| PENGESAHAN SKRIPSI | ii |
| PERNYATAAN..... | iii |
| KATA PENGANTAR | iv |
| DAFTAR ISI..... | vii |
| DAFTAR TABEL..... | xi |
| DAFTAR GAMBAR | xii |
| DAFTAR LAMPIRAN..... | xv |
| INTISARI..... | xvi |
| BAB I PENDAHULUAN | 1 |
| A. Latar Belakang | 1 |
| B. Rumusan Masalah | 4 |
| C. Batasan Masalah..... | 4 |
| D. Tujuan Penelitian | 5 |
| E. Manfaat Penelitian | 5 |
| BAB II TINJAUAN PUSTAKA DAN LANDASAN TEORI | 6 |
| A. Tinjauan Pustaka | 6 |
| B. Landasan Teori..... | 9 |

| | |
|---|-----------|
| 1. Berorientasi Objek..... | 9 |
| 2. Unified Modeling Language (UML)..... | 9 |
| 3. Diagram <i>Use Case</i> | 10 |
| 4. Diagram Aktivitas | 11 |
| 5. Diagram sekuensial | 11 |
| 6. Diagram Kelas | 12 |
| 7. Diagram Entity Relationship (E-R)..... | 13 |
| 8. MySQL..... | 13 |
| 9. Global Positioning System (GPS) | 15 |
| 10. Android | 15 |
| 11. Google Maps..... | 16 |
| BAB III METODE PENGEMBANGAN SISTEM | 17 |
| A. Metode Pengembangan Sistem | 17 |
| 1. Pengumpulan Data | 17 |
| 2. Analisis Sistem | 17 |
| 3. Perancangan dan Pemrograman Sistem..... | 18 |
| 4. Pengembangan dan Pengujian Sistem | 18 |
| B. Alat Penelitian..... | 19 |
| 1. Perangkat Lunak (<i>Software</i>)..... | 19 |
| 2. Perangkat Keras (<i>Hardware</i>)..... | 19 |

| | |
|---|-----------|
| C. Arsitektur Sistem Indekos | 20 |
| BAB IV ANALISIS DAN PENGEMBANGAN SISTEM..... | 21 |
| A. Unified Modeling Language (UML)..... | 21 |
| 1. Diagram <i>Use Case</i> | 21 |
| 2. Diagram Aktivitas | 22 |
| 3. Diagram Sekuensial..... | 37 |
| 4. Diagram Kelas | 42 |
| B. Perancangan Basis data (<i>Database</i>) | 43 |
| 1. Diagram Entity Relationship (E-R)..... | 43 |
| 2. Perancangan Tabel | 48 |
| C. Perancangan Antarmuka | 54 |
| 1. Antarmuka Admin | 54 |
| 2. Antarmuka Pemilik Indekos | 58 |
| 3. Antarmuka Pengguna (<i>User</i>)..... | 63 |
| BAB V IMPLEMENTASI DAN PENGUJIAN SISTEM..... | 68 |
| A. Implementasi Sistem | 68 |
| 1. Basis data (<i>Database</i>)..... | 68 |
| 2. Aplikasi Website server..... | 75 |
| 3. Aplikasi Mobile client | 83 |
| B. Pengujian Sistem..... | 89 |

| | |
|---|----|
| 1. Pengujian alpha | 89 |
| 2. Pengujian beta | 90 |
| C. Rencana Pengujian | 91 |
| 1. Pengujian Alpha | 91 |
| 2. Pengujian Beta..... | 91 |
| BAB VI HASIL DAN PEMBAHASAN | 93 |
| A. Hasil Pengujian Sistem | 93 |
| B. Pengujian Alpha | 93 |
| C. Pengujian Beta | 94 |
| 1. Hasil pengujian pemilik indekos | 94 |
| 2. Hasil pengujian pengguna indekos | 94 |
| BAB VII PENUTUP | 96 |
| A. Kesimpulan | 96 |
| B. Saran..... | 96 |
| DAFTAR PUSTAKA | 98 |
| LAMPIRAN | 99 |

DAFTAR TABEL

| | |
|--|----|
| 1. Tabel 2.1 (Perbandingan studi Pustaka) | 8 |
| 2. Tabel 2.2 (Daftar simbol diagram <i>use case</i>) | 10 |
| 3. Tabel 2.3 (Daftar simbol diagram aktivitas) | 11 |
| 4. Tabel 2.4 (Daftar simbol diagram sekuensial) | 12 |
| 5. Tabel 2.5 (Daftar simbol diagram kelas) | 12 |
| 6. Tabel 4.1 (Provinsi) | 48 |
| 7. Tabel 4.2 (Kab_kota) | 48 |
| 8. Tabel 4.3 (Pemilik) | 49 |
| 9. Tabel 4.4 (Lupa_password) | 49 |
| 10. Tabel 4.5 (Indekos) | 50 |
| 11. Tabel 4.6 (Kamar_fasilitas_int) | 50 |
| 12. Tabel 4.7 (Fasilitas_master) | 51 |
| 13. Tabel 4.8 (Fasilitas_eks) | 51 |
| 14. Tabel 4.9 (Fasilitas_int) | 51 |
| 15. Tabel 4.10 (Kamar) | 52 |
| 16. Tabel 4.11 (Indekos_fasilitas_eks) | 53 |
| 17. Tabel 4.12 (Admin) | 53 |
| 18. Tabel 5.1 (Rencana pengujian alpha) | 89 |
| 19. Tabel 5.2 (Rencana pengujian beta bagian pemilik) | 90 |
| 20. Tabel 5.3 (Rencana pengujian beta bagian pengguna) | 90 |
| 21. Tabel 5.4 (Daftar penguji tahap alpha) | 91 |
| 22. Tabel 5.5 (Daftar penguji tahap beta pemilik indekos) | 92 |
| 23. Tabel 5.6 (Daftar Penguji tahap beta pengguna indekos) | 92 |
| 24. Tabel 6.1 (Hasil pengujian sistem tahap alpha) | 93 |
| 25. Tabel 6.2 (Hasil pengujian sistem pemilik indekos) | 94 |
| 26. Tabel 6.3 (Hasil pengujian sistem pengguna indekos) | 94 |

DAFTAR GAMBAR

| | |
|---|----|
| 1. Gambar 2.1 (Daftar simbol diagram E-R)..... | 13 |
| 2. Gambar 3.1 (Arsitektur sistem indekos) | 18 |
| 3. Gambar 4.1 (Diagram <i>Use Case</i> admin) | 19 |
| 4. Gambar 4.2 (Diagram <i>Use Case</i> pengguna indekos) | 19 |
| 5. Gambar 4.3 (Diagram <i>Use Case</i> pemilik indekos) | 20 |
| 6. Gambar 4.4 (Diagram Aktivitas admin provinsi) | 21 |
| 7. Gambar 4.5 (Diagram Aktivitas admin kab kota) | 22 |
| 8. Gambar 4.6 (Diagram Aktivitas admin fasilitas master) | 23 |
| 9. Gambar 4.7 (Diagram Aktivitas admin fasilitas eksternal) | 24 |
| 10. Gambar 4.8 (Diagram aktivitas pemilik data pribadi/profil) | 25 |
| 11. Gambar 4.9 (Diagram aktivitas pemilik ubah password) | 26 |
| 12. Gambar 4.10 (Diagram aktivitas pemilik indekos) | 27 |
| 13. Gambar 4.11 (Diagram aktivitas pemilik kamar) | 28 |
| 14. Gambar 4.12 (Diagram aktivitas fasilitas internal) | 29 |
| 15. Gambar 4.13 (Diagram aktivitas kamar fasilitas internal) | 30 |
| 16. Gambar 4.14 (Diagram aktivitas pemilik kontrak) | 31 |
| 17. Gambar 4.15 (Diagram aktivitas indekos fasilitas eksternal) | 32 |
| 18. Gambar 4.16 (Diagram aktivitas pengguna indekos terdekat) | 33 |
| 19. Gambar 4.17 (Diagram aktivitas pengguna pencarian indekos kamar) | 34 |
| 20. Gambar 4.18 (Diagram aktivitas pengguna sinkronisasi data) | 35 |
| 21. Gambar 4.19 (Diagram aktivitas pengguna rute indekos) | 36 |
| 22. Gambar 4.20 (Diagram sekuensial admin) | 37 |
| 23. Gambar 4.21 (Diagram sekuensial pemilik indekos kamar) | 38 |
| 24. Gambar 4.22 (Diagram sekuensial pemilik kontrak) | 39 |
| 25. Gambar 4.23 (Diagram sekuensial pengguna cari indekos, rute indekos) | 40 |
| 26. Gambar 4.24 (Diagram sekuensial pengguna sinkronisasi data) | 41 |
| 27. Gambar 4.25 (Diagram kelas) | 42 |
| 28. Gambar 4.26 (Diagram <i>Entity Relationship</i> E-R) | 43 |
| 29. Gambar 4.27 (Diagram E-R provinsi, kab_kota dan pemilik) | 44 |
| 30. Gambar 4.28 (Diagram E-R kab_kota, indekos dan pemilik) | 44 |
| 31. Gambar 4.29 (Diagram E-R indekos, fasilitas_eks dan kamar) | 45 |
| 32. Gambar 4.30 (Diagram E-R pemilik, fasilitas_int dan kamar) | 45 |
| 33. Gambar 4.31 (Diagram E-R kab_kota, fasilitas_eks_dan fasilitas_master) | 46 |
| 34. Gambar 4.32 (Diagram E-R entitas admin) | 46 |
| 35. Gambar 4.33 (Diagram lain <i>Entity Relationship</i>) | 47 |
| 36. Gambar 4.34 (Halaman login admin) | 54 |
| 37. Gambar 4.35 (Halaman depan <i>homepage</i>) | 54 |
| 38. Gambar 4.36 (Halaman provinsi) | 55 |
| 39. Gambar 4.37 (Antarmuka halaman kab kota)..... | 56 |
| 40. Gambar 4.38 (Halaman fasilitas master) | 56 |
| 41. Gambar 4.39 (Halaman fasilitas eksternal) | 57 |

| | |
|--|----|
| 42. Gambar 4.40 (Halaman login, daftar konfirmasi email dan lupa password) | 58 |
| 43. Gambar 4.41 (Halaman pemilik pengisian data pribadi) | 59 |
| 44. Gambar 4.42 (Halaman depan <i>homepage</i>) | 59 |
| 45. Gambar 4.43 (Halaman profil) | 60 |
| 46. Gambar 4.44 (Halaman ubah password) | 60 |
| 47. Gambar 4.45 (Halaman depan indekos) | 61 |
| 48. Gambar 4.46 (Halaman tambah indekos) | 61 |
| 49. Gambar 4.47 (Halaman kamar) | 62 |
| 50. Gambar 4.48 (Halaman kontrak) | 62 |
| 51. Gambar 4.49 (Halaman <i>Splash screen</i>) | 63 |
| 52. Gambar 4.50 (Halaman depan <i>homepage</i>) | 64 |
| 53. Gambar 4.51 (Halaman sinkronisasi data) | 64 |
| 54. Gambar 4.52 (Halaman cari indekos) | 65 |
| 55. Gambar 4.53 (Halaman daftar indekos) | 65 |
| 56. Gambar 4.54 (Halaman detail indekos) | 66 |
| 57. Gambar 4.55 (Halaman rute indekos) | 66 |
| 58. Gambar 4.56 (Halaman detail kamar) | 67 |
| 59. Gambar 5.1 (<i>Database</i> db_indekos) | 68 |
| 60. Gambar 5.2 (Struktur tabel admin) | 69 |
| 61. Gambar 5.3 (Struktur tabel pemilik) | 69 |
| 62. Gambar 5.4 (Struktur tabel provinsi) | 70 |
| 63. Gambar 5.5 (Struktur tabel kab_kota) | 70 |
| 64. Gambar 5.6 (Struktur tabel indekos) | 71 |
| 65. Gambar 5.7 (Struktur tabel kamar) | 71 |
| 66. Gambar 5.8 (Struktur tabel fasilitas_master) | 72 |
| 67. Gambar 5.9 (Struktur tabel fasilitas_eks) | 72 |
| 68. Gambar 5.10 (Struktur tabel fasilitas_int) | 73 |
| 69. Gambar 5.11 (Struktur tabel indekos_fasilitas_eks) | 73 |
| 70. Gambar 5.12 (Struktur tabel kamar_fasilitas_int) | 74 |
| 71. Gambar 5.13 (Struktur tabel lupa_password) | 74 |
| 72. Gambar 5.14 (Tampilan admin login) | 75 |
| 73. Gambar 5.15 (Tampilan admin provinsi) | 75 |
| 74. Gambar 5.16 (Tampilan admin kab kota) | 76 |
| 75. Gambar 5.17 (Tampilan admin fasilitas eksternal) | 76 |
| 76. Gambar 5.18 (Tampilan admin fasilitas master) | 77 |
| 77. Gambar 5.19 (Tampilan pemilik login) | 77 |
| 78. Gambar 5.20 (Tampilan pengisian data pribadi) | 78 |
| 79. Gambar 5.21 (Tampilan pemilik lupa password) | 78 |
| 80. Gambar 5.22 (Tampilan pemilik indekos) | 79 |
| 81. Gambar 5.23 (Tampilan pemilik indekos detail) | 79 |
| 82. Gambar 5.24 (Tampilan pemilik fasilitas eksternal) | 80 |
| 83. Gambar 5.25 (Tampilan pemilik kamar) | 80 |
| 84. Gambar 5.26 (Tampilan pemilik kamar tambah) | 81 |
| 85. Gambar 5.27 (Tampilan pemilik kamar fasilitas internal) | 81 |
| 86. Gambar 5.28 (Tampilan pemilik fasilitas internal) | 82 |

| | |
|---|----|
| 87. Gambar 5.29 (Tampilan pemilik kontrak) | 82 |
| 88. Gambar 5.30 (Tampilan pengguna <i>splash screen</i>) | 83 |
| 89. Gambar 5.31 (Tampilan halaman depan <i>homepage</i>) | 83 |
| 90. Gambar 5.32 (Tampilan sinkronisasi data) | 84 |
| 91. Gambar 5.33 (Tampilan daftar provinsi) | 84 |
| 92. Gambar 5.34 (Tampilan daftar kab kota) | 85 |
| 93. Gambar 5.35 (Tampilan daftar fasilitas master) | 85 |
| 94. Gambar 5.36 (Tampilan daftar fasilitas eksternal) | 86 |
| 95. Gambar 5.37 (Tampilan daftar indekos) | 86 |
| 96. Gambar 5.38 (Tampilan detail indekos) | 87 |
| 97. Gambar 5.39 (Tampilan rute indekos) | 88 |
| 98. Gambar 5.39 (Tampilan detail kamar) | 88 |

DAFTAR LAMPIRAN

| | |
|--|-----|
| A. Daftar kuesioner pemilik indekos | 99 |
| B. Daftar sekolah SMAN Daerah Istimewa Yogyakarta | 104 |
| C. Daftar Universitas Negeri, Swasta dan Institut di Daerah Istimewa Yogyakarta | 106 |
| D. Daftar potongan <i>Source Code</i> Program | 107 |

ANALISIS DAN PERANCANGAN SISTEM INDEKOS

MENGGUNAKAN METODE UNIFIED MODELING LANGUAGE (UML)

Muhammad Fuad Adib
10650008

INTISARI

Indekos adalah sebuah jasa yang menawarkan sebuah kamar/tempat untuk ditinggali dengan sejumlah pembayaran teruntuk setiap periode, bagi pemilik indekos sudah banyak melakukan promosi agar indekosnya dapat dengan mudah dicari oleh pencari indekos, namun selama ini masih kurangnya promosi yang dilakukan seperti penyebaran iklan, website dan lainnya. Dan masih belum adanya yang bisa mengatur setiap kamar yang ada dalam indekos yang dimiliki, dan juga belum adanya yang dapat mengatur waktu sewa – menyewa bagi setiap kamar. Dan bagi pencari indekos mencari indekos disuatu tempat tidaklah mudah apalagi sesuai dengan fasilitas yang diinginkan. Tujuan dari penelitian ini agar pemilik indekos dapat mempromosikan indekosnya, dan dapat mengatur setiap kamar yang ada dan juga dapat mengatur waktu sewa – menyewa bagi setiap kamar, dan tujuan bagi pengguna/pencari indekos agar dapat dengan mudah mencari indekos yang sesuai dengan fasilitas yang diinginkan.

Metode penelitian yang dilakukan adalah dengan wawancara terhadap para pengguna aplikasi, analisis kebutuhan hasil dari wawancara, perancangan dan implementasi program. Perancangan sistem menggunakan UML(*Unified Modeling Language*) dan sistem dijalankan pada dua *platform* yaitu *Website* sebagai *server* dan *Android* untuk pengguna.

Sistem yang dihasilkan bagi pemilik indekos dapat menjadi tempat untuk mempromosikan, mengatur setiap kamar dan mengatur sewa – menyewa kamar. dan bagi pengguna/pencari indekos sistem yang dihasilkan dapat mencari indekos sesuai fasilitas yang diinginkan.

Kata kunci: Indekos, Server, Android, UML

SYSTEM ANALYSIS AND DESIGN LODGER
USING METHOD UNIFIED MODELING LANGUAGE (UML)

Muhammad Fuad Adib
10650008

ABSTRACT

Lodger is a service that offers a room / place to live with a number of payments each period will belong, for for homestay owners have been doing a lot of promotion so that his lodger can be easily searched by search lodger, however, for this is still lacking was the campaign undertaken such as the spread of advertising, website and other. And still the lack of which could regulate every room in homestay owned, and also the absence of which can set the lease time for each room. And the search for a boarder looking for a lodger in one place is not easy especially in accordance with the desired facility. The purpose of this research in order to promote the homestay owner of his lodger, and can organize every room there and also to set the time of the lease for each room. And objectives for the user / searcher lodger, to easily search for lodger in accordance with the desired facility.

Research methodology is the interview of the user application , the results of the needs analysis interview , program design and implementation . System design using UML (Unified Modeling Language) and the system is run on the Android platform.

The resulting system for owners lodger can be a place to promote, organize and arrange each room rental rooms. And for the user / searcher lodger, the resulting system can search for desired facilities boarder appropriate.

Keywords: Lodger , Server , Android , UML

BAB I

PENDAHULUAN

A. Latar Belakang

Dalam Kamus Besar Bahasa Indonesia Indekos adalah sebuah jasa yang menawarkan sebuah kamar/tempat untuk ditinggali dengan sejumlah pembayaran teruntuk setiap priode, bisnis indekos masih sangat menguntungkan, indekos banyak dicari oleh seorang yang ingin tinggal didaerah tertentu yang bukan tempat tinggal aslinya, seperti karyawan yang bekerja diperusahaan tertentu yang jauh jika pulang – pergi dari kantor – kerumah yang akhirnya mencari tempat tinggal yang dekat dengan kantor, atau juga seorang mahasiswa yang kuliah didaerah yang jauh dari tempat tinggalnya, akan mencari tempat tinggal yang dekat dengan kampusnya agar tidak menghabiskan waktu hanya untuk pulang – pergi dan juga agar kuliahnya lebih mudah diatur.

Perusahaan atau instansi seperti kampus, universitas atau sekolah dalam menyeleksi karyawan, mahasiswa atau siswa baru jarang sekali dilihat dari daerah dimana tempat perusahaan atau instansi itu berdiri tapi dari aspek lain, akhirnya mereka yang diterima harus mencari tempat tinggal baru untuk sementara didaerah yang mereka belum tahu sebelumnya, mencari indekos didaerah yang baru sangatlah sulit apalagi belum tentu ketika menemukan indekos, indekos tersebut masih kosong untuk bisa ditinggali.

Mencari indekos tidaklah mudah selain tempatnya yang masih kosong atau tidak, terkadang ada aspek lain seperti fasilitas atau harga, mungkin dapat menemukan indekos yang masih kosong tapi harganya bisa saja mahal dan sangat mahal yang akhirnya batal untuk menempati indekos tersebut, dan mungkin ada juga yang mencari berdasarkan fasilitas yang ada didalam indekos tersebut, seperti televisi, tempat tidur, dan lain sebagainya yang sudah disediakan oleh pemilik indekos. Faktor – faktor tersebut biasanya menjadi salah satu pertimbangan dalam memilih indekos didaerah baru.

Sudah banyak cara promosi – promosi indekos seperti menyebarkan kertas iklan/promosi ditempat – tempat ramai atau membuat pemberitahuan yang dipasang didepan rumahnya, contoh “menerima indekos untuk putra” dan menjadikan website sebagai media promosi untuk indekos dengan keterangan – keterangannya, namun masih ada kelemahan/kekurangan dari cara – cara yang diuraikan diatas, untuk menyebarkan iklan/promosi, perlu seorang kurir untuk yang menyebarkannya dan belum tentu yang mendapatkan iklan/promosi tersebut sedang mencari indekos. Jika memasang iklan/promosi hanya didepan rumah hanya orang – orang yang berada didekat rumah saja yang tahu, dan juga bagi yang mencari belum tahu apa sajakah fasilitas – fasilitas yang ada didalam indekos tersebut. Promosi menggunakan media internet/website lebih baik dari yang sebelumnya informasi bisa dilihat oleh semua orang, namun kebanyakan website masih hanya memberikan informasi pasif, contohnya “info indekos A, terdapat 5 ruangan harga Rp. 100.000/bulan, fasilitas B, C dan D didaerah E”, dari informasi tersebut sudah cukup memadai, namun sebagai pencari indekos tidak

tahu ruangan tersebut berapa yang masih kosong, dan yang sudah ada isinya, dan alamat bagi yang belum tahu daerah baru, akan susah sekali mencari alamat yang diberikan diwebsite, jadi masih terdapat kekurangan dari promosi – promosi tersebut.

Android adalah salah satu *smartphone* yang mendukung GPS (*Global Positioning System*) yang berfungsi sebagai alat koordinat dari suatu tempat atau posisi yang telah diketahui, GPS di *smartphone* sering digunakan untuk menentukan posisi dari tempat berada, dan juga sering digunakan untuk mencari tempat / daerah, atau menghitung jarak dari kota/tempat A ke kota/tempat B berapakah jaraknya. GPS dapat digunakan untuk menentukan letak dimana tempat indeks yang diinginkan, menentukan rute untuk menuju tempat indeks, menghitung jarak terdekat dari pengguna/pencari indeks ke tempat indeks tersebut berada, dan juga menghitung jarak terdekat dari tempat tertentu yang diinginkan, contohnya “ingin mencari indeks yang berada didekat kampus UIN Sunan Kalijaga.”

Implementasi dari program yang akan dibuat berbasis website dan *mobile android* yang berorientasi objek, untuk itu perancangan sistem menggunakan metode *Unified Modeling Language* (UML), karena UML adalah perancangan sistem yang memfokuskan pada pengembangan sistem yang berorientasi objek, menggunakan UML pada pemrograman berorientasi objek akan mempermudah pada saat implementasi program sebab perancangan sistem dan program sistem sama berorientasi objek.

Dari latar belakang tersebut penulis melakukan penelitian untuk merancang dan membuat sistem indekos untuk bisa digunakan bagi seorang pendatang atau perantau dalam mencari indekos yang bisa menjadi tempat tinggalnya sementara di daerah tersebut dengan semua fasilitas yang diinginkan. Dan lebih detail lagi dapat melihat setiap ruangan tersebut masih kosong atau tidak dan melihat setiap fasilitas yang ada didalamnya, dengan rancangan seperti ini diharapkan pencari indekos tidak hanya lihat indekos ada dimana saja tetapi bisa lihat juga masih ada ruangan kosong atau tidak diindekos tersebut dan fasilitas yang ada didalamnya.

B. Rumusan Masalah

1. Bagaimana merancang agar sistem dapat mengatur data ruangan indekos untuk pemilik indekos tersebut
2. Bagaimana merancang sistem untuk mengatur waktu sewa – menyewa ruangan dan atau perpanjangan sewa bagi penyewa/pengguna dan pemilik indekos tersebut
3. Bagaimana menjadikan sistem sebagai tempat mencari indekos yang masih kosong atau masih bisa ditempati

C. Batasan Masalah

1. Sistem untuk *user* dibuat berbasis mobile Android.
2. Sistem untuk pemilik indekos dibuat berbasis website menggunakan *framework PHP Codeigniter*
3. Untuk tampilan website penulis menggunakan *bootstrap*
4. Fasilitas terdapat dua internal dan eksternal, internal adalah fasilitas yang ada didalam kamar, dan eksternal adalah tempat – tempat terdekat indekos, penulis

membatasi kampus/universitas dan sekolah dan membatasi wilayah untuk Sleman, Bantul dan Kota Yogyakarta.

5. Diagram UML yang digunakan adalah diagram *use case*, diagram aktifitas, diagram kelas, diagram squensial dan diagram E-R (Entity Relationship) untuk rancangan basis datanya
6. Untuk keamanan, penulis hanya sampai validasi akun pemilik indekos

D. Tujuan Penelitian

1. Merancang sistem yang dapat mengatur data ruangan indekos untuk pemilik indekos tersebut
2. Merancang sistem yang dapat mengatur waktu sewa – menyewa ruangan dan atau perpanjangan sewa bagi penyewa/pengguna dan pemilik indekos tersebut
3. Merancang sistem agar menjadi tempat untuk mencari indekos yang masih kosong atau masih bisa ditempati.

E. Manfaat Penelitian

Manfaat yang didapatkan dari hasil penelitian ini untuk pencari indekos seperti perantau atau mahasiswa luar kota adalah dapat menjadi tempat untuk mencari indekos di daerah yang akan ditinggali, dengan semua fasilitas yang sesuai keinginan. Dan bagi pemilik indekos dapat menjadi tempat untuk mengiklankan atau mempromosikan indekos agar lebih banyak lagi yang datang untuk menempati, dan juga dapat mengatur setiap ruang bagi pemilik indekos. Dan bagi penulis manfaat dari penelitian ini adalah dapat menjadi ilmu lebih untuk belajar membuat software yang mengikuti aturan siklus hidup pengembangan perangkat lunak (*Software Development Life Cycle*).

BAB II

TINJAUAN PUSTAKA DAN LANDASAN TEORI

A. Tinjauan Pustaka

Penelitian ini dikembangkan dengan menggunakan beberapa referensi penelitian yang berhubungan dengan objek pembahasan. Penggunaan referensi ditujukan untuk memberikan batasan – batasan sistem yang nantinya dapat dikembangkan lebih lanjut, dengan mengacu pada referensi yang digunakan, diharapkan pengembangan sistem nanti dapat melahirkan suatu sistem baru yang belum ada pada referensi sebelumnya.

Penelitian yang dilakukan oleh Akhmad Hanif, 2013 mengenai “Pencarian Tempat Kos Dengan Teknologi Augmented Reality Berbasis Smartphone Android” penelitian ini mengembangkan suatu sistem pencariankos/indekos di Yogyakarta dengan memanfaatkan teknologi *Augmented reality* berbasis *smartphone* android. Sistem yang dihasilkan dapat memberikan informasi indekos dengan fasilitas yang diberikan dan dapat menunjukkan rute untuk menuju indekos tersebut.

Penelitian yang dilakukan oleh Sholihah, 2012 mengenai “Sistem Pendukung Keputusan untuk Pemilihan Pondok Pesantren sebagai tempat tinggal Mahasiswa di D.I. Yogyakarta.” Merupakan suatu sistem pendukung keputusan yang dapat merekomendasikan pesantren bagi tempat tinggal mahasiswa di D.I.Yogyakarta. Sistem ini memberi keluaran rekomendasi rekomendasi pondok pesantren bagi

mahasiswa yang tinggal di D.I. Yogyakarta dengan beberapa ketentuan yang dipilih sebelumnya.

Penelitian yang dilakukan oleh Hardi Saputra, 2012 mengenai “Implementasi *Global Positioning System* (GPS) untuk Pariwisata Daerah Istimewa Yogyakarta pada *Mobile Device* Berbasis Android.” Penelitian yang dibangun melalui dua tahap yaitu pembangunan web untuk admin sebagai *server*. Yang dibangun menggunakan PHP dan MYSQL. Dan tahap kedua adalah membangun aplikasi untuk *smartphone* android sebagai *client* dan dibangun menggunakan Java. Sistem yang dibangun mampu menampilkan tempat – tempat wisata, penginapan, tempat belanja, Stasiun Pengisian Bahan Bakar Umum (SPBU), dan Anjungan Tunai Mandiri (ATM) berdasarkan lokasi terdekat dari posisi pengguna.

Penelitian yang dilakukan oleh Syifa Qurrotu ‘Aini, 2012 mengenai “Sistem Informasi Geografis Berbasis Mobile (Pemetaan Objek Wisata Religi Studi Kasus Jateng-DIY).” Penelitian ini merancang sebuah Sistem Informasi Geografis Berbasis Mobile, yang dibangun menggunakan IDE Eclipse, aplikasi dijalankan pada perangkat *mobile platform* Android OS. Sistem yang dibangun mampu memberikan informasi mengenai objek pariwisata religi yang ada di Jawa Tengah dan di Daerah Istimewa Yogyakarta beserta fasilitas – fasilitas yang ada disekitar objek pariwisata seperti SPBU, ATM dan restoran. Selain itu, sistem juga mampu menampilkan peta letak objek wisata, serta mampu mencari objek wisata yang terdekat dengan posisi pengguna.

Tabel 2.1 (Perbandingan studi pustaka)

| Peneliti | Penelitian | Tools | Metode | Hasil |
|---------------------------|---|---------------------|--------|---|
| Akhmad Hanif, 2013 | Pencarian Tempat Kos Dengan Teknologi Augmented Reality Berbasis Smartphone Android | Android, PHP, MySQL | SDLC | Memberikan informasi indekos, fasilitas dan rute jalan menuju menuju indekos. |
| Syifa Qurrotu 'Aini, 2012 | Sistem Informasi Geografis Berbasis Mobile (Pemetaan Objek Wisata Religi Studi Jateng-DIY) | PHP, Android, MySQL | SDLC | Memberikan informasi tempat pariwisata dan fasilitas terdekat didaerah tersebut |
| Hardi Saputra, 2012 | Implementasi <i>Global Positioning System</i> (GPS) untuk Pariwisata Daerah Istimewa Yogyakarta pada <i>Mobile Device</i> Berbasis Android. | PHP, MySQL, Android | SDLC | Memberikan informasi wisata, dan penginapan, tempat belanja, SPBU, ATM. |
| Sholihah, 2012 | Sistem Pendukung Keputusan untuk Pemilihan Pondok Pesantren sebagai tempat tinggal Mahasiswa di D.I. Yogyakarta. | PHP, MySQL | SDLC | Memberikan rekomendasi pondok pesantren untuk mahasiswa. |
| M Fuad Adib, 2014 | Analisis dan Perancangan Sistem Indekos Menggunakan Metode Unified Modeling Language (UML) | PHP, MySQL, Android | SDLC | Memberikan informasi indekos,fasilitas yang ada, dan rute jalan ketempat indekos. |

B. Landasan Teori

1. Berorientasi Objek

Berorientasi objek atau *object oriented* merupakan paradigma baru dalam rekayasa perangkat lunak yang memandang sistem sebagai kumpulan objek – objek diskrit yang saling berinteraksi. Yang dimaksud berorientasi objek adalah bahwa mengorganisasikan perangkat lunak sebagai kumpulan objek – objek diskrit yang bekerja sama antar informasi atau struktur data dan perilaku (*behavior*) yang mengaturnya. (Sholiq, 2006)

2. Unified Modeling Language (UML)

Notasi UML dibuat sebagai kolaborasi dari Grady booch, DR. Jame Rumbough, Ivar Jacobson, Rebecca Wirfs-Brock, Peter Yourdon, dan lainnya. Jacobson menulis tentang pendefinisian persyaratan – persyaratan sistem yang di sebut *use case*.Juga mengembangkan sebuah metode untuk perancangan sistem yang disebut *Object-Oriented Software Enginnering* (OOSE) yang berfokus pada analisis.Booch, Rumbough dan Jacobson biasa disebut dengan tiga sekawan (*tree amigos*). Semuanya bekerja di *Rational Software Corporation* dan berfokus pada standarisasi dan perbaikan ulang UML.


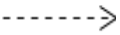



Pembangunan beberapa metode menjadi UML dimulai 1993. Setiap orang dari tiga sekawan di rational mulai menggabungkan idenya dengan metode - metode lainnya. Pada akhir tahun 1995 *Unified Method* versi 0.8 diperkenalkan.*Unified Method* diperbaiki dan diubah menjadi UML pada tahun 1996, UML 1.0 disahkan dan diberikan pada *Object Technology Group* (OTG) pada tahun 1997, dan pada

tahun itu juga beberapa perusahaan pengembang utama perangkat lunak mulai mengadopsinya. Pada tahun yang sama OMG merilis UML 1.1 sebagai standar industri. (Sholiq, 2006)

3. Diagram *Use Case*

Diagram *use case* atau *use case diagram* menyajikan interaksi antara *use case* dan aktor. Dimana, aktor dapat berupa orang, peralatan, atau sistem lain yang berinteraksi dengan sistem yang sedang dibangun. *Use case* menggambarkan fungsionalitas sistem atau persyaratan – persyaratan yang harus dipenuhi sistem dari pandangan pemakai. (Sholiq, 2006)






Tabel 2.2 (Daftar simbol diagram *use case*)

| No | Gambar | Nama | Keterangan |
|----|---|--------------------|---|
| 1 |  | <i>Actor</i> | Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan <i>use case</i> . |
| 2 |  | <i>Include</i> | Menspesifikasikan bahwa <i>use case</i> sumber secara <i>eksplisit</i> . |
| 3 |  | <i>Association</i> | Apa yang menghubungkan antara objek satu dengan objek lainnya. |
| 4 |  | <i>System</i> | Menspesifikasikan paket yang menampilkan sistem secara terbatas. |
| 5 |  | <i>Use Case</i> | Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor |

4. Diagram Aktivitas

Diagram aktifitas atau *Activity diagram* menggambarkan aliran fungsionalitas sistem. Pada tahap pemodelan bisnis, diagram aktivitas dapat digunakan untuk menunjukkan aliran kerja bisnis (*business work-flow*). Dapat juga digunakan untuk menggambarkan aliran kejadian (*flow of events*) dalam *use case*. (Sholih, 2006)

Tabel 2.3 (Daftar simbol diagram aktivitas)



| No | Gambar | Nama | Keterangan |
|----|---|----------------------------|--|
| 1 |  | <i>Association</i> | Menghubungkan antar aktivitas satu dengan aktivitas lain. |
| 2 |  | <i>Action</i> | State dari sistem yang mencerminkan eksekusi dari suatu aksi |
| 3 |  | <i>Initial Node</i> | Bagaimana objek dibentuk atau diawali. |
| 4 |  | <i>Activity Final Node</i> | Bagaimana objek dibentuk dan dihancurkan |
| 5 |  | <i>Fork Node</i> | Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran |

5. Diagram sekuensial

Diagram sekuensial menggambarkan kelakuan/prilaku objek pada proses dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambarkan diagram sekuensial maka harus diketahui objek – objek yang terlibat dalam sebuah proses beserta metode – metode yang dimiliki kelas yang diinisialisasikan menjadi objek.

Diagram sekuensial atau *sequence diagram* digunakan untuk menunjukkan aliran fungsionalitas dalam *use case*, diagram sekuensial adalah diagram interaksi yang disusun berdasarkan urutan waktu. (Sholiq, 2006)


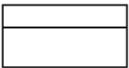
Tabel 2.4 (Daftar simbol diagram sekuensial)

| No | Gambar | Nama | Keterangan |
|----|---|-----------------|--|
| 1 |  | <i>LifeLine</i> | Objek <i>entity</i> , antarmuka yang saling berinteraksi. |
| 2 |  | <i>Message</i> | Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi |

6. Diagram Kelas

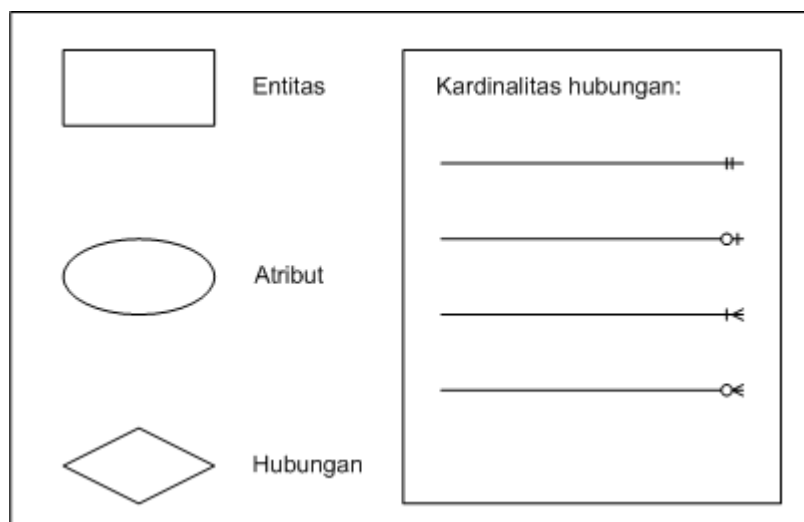
Diagram kelas atau *class diagram* menunjukkan interaksi antar kelas dalam sistem. Sebagai contoh nim 1234567 adalah objek dari kelas Mahasiswa. Kelas mengandung informasi dan tingkah laku (*behavior*) yang berkaitan dengan informasi tersebut. Kelas mahasiswa mengandung Nim, nama dan mempunyai tingkah laku untuk melihat jadwal kuliah. (Sholiq, 2006)

Tabel 2.5 (Daftar simbol diagram kelas)

| NO | GAMBAR | NAMA | KETERANGAN |
|----|---|-----------------------|---|
| 1 |  | <i>Generalization</i> | Hubungan dimana objek anak (<i>descendent</i>) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (<i>ancestor</i>). |
| 2 |  | <i>Class</i> | Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama. |

7. Diagram Entity Relationship (E-R)

Model E-R adalah suatu model yang digunakan untuk menggambarkan data dalam bentuk entitas, atribut dan hubungan antara entitas. Huruf E sendiri menyatakan entitas dan R menyatakan hubungan (dari kata *Relationship*). Model ini dinyatakan dalam bentuk diagram. Itulah sebabnya model E-R acapkali juga disebut sebagai diagram E-R. (Kadir, Dasar perancangan & implementasi database relasional 2009)



Gambar 2.1 (Daftar simbol diagram E-R)

8. MySQL

MySQL (dibaca: mai-se-kyu-el) merupakan *software* yang tergolong sebagai DBMS (*Database Management System*) yang bersifat *Open Source*. Open Source menyatakan bahwa software ini dilengkapi dengan *source code* (kode yang dipakai untuk membuat MySQL), selain tentu saja bentuk *executable*-nya atau kode yang dapat dijalankan secara langsung dalam sistem operasi, dan bisa diperoleh dengan cara *men-download* (mengunduh) di internet secara gratis. (Kadir, Tuntunan praktis belajar database menggunakan MySQL 2008)

MySQL dikembangkan oleh sebuah perusahaan Swedia bernama *MySQL AB* yang pada saat itu bernama *TcX DataKonsult AB* Sekitar tahun 1994-1995, namun cikal bakal kodenya sudah ada sejak 1979. Awalnya TcX membuat MySQL dengan tujuan mengembangkan aplikasi web untuk klien. TcX merupakan perusahaan pengembang software dan konsultan database. Saat ini MySQL sudah diakuisisi oleh Oracle Corp.

MySQL adalah salah satu jenis database server yang sangat terkenal dan banyak digunakan untuk membangun aplikasi web yang menggunakan database sebagai sumber dan pengelolaan datanya. Kepopuleran MySQL antara lain karena MySQL menggunakan SQL sebagai bahasa dasar untuk mengakses database-nya sehingga mudah untuk digunakan, kinerja *query* cepat, dan mencukupi untuk kebutuhan database perusahaan – perusahaan skala menengah – kecil. MySQL juga bersifat *open source* dan *free* pada berbagai platform (kecuali pada Windows, yang bersifat *shareware*). MySQL didistribusikan dengan lisensi *open source* GPL (*General Public License*) mulai versi 3.23, pada bulan juni 2000. Software MySQL bisa diunduh di <http://www.mysql.org> atau <http://www.mysql.com>.

MySQL merupakan database yang pertama kali didukung oleh bahasa pemrograman script untuk internet (PHP dan Perl). MySQL dan PHP dianggap sebagai pasangan software pengembangan aplikasi web yang ideal. MySQL lebih sering digunakan untuk membangun aplikasi berbasis web, umumnya pengembangan aplikasinya menggunakan bahasa pemrograman script PHP. (Arief, 2011)

9. Global Positioning System (GPS)

Dalam (Hardi,2012) Andre menyatakan GPS merupakan nama sebuah sistem navigasi global berbasis satelit yang dikembangkan oleh departemen pertahanan Amerika Serikat. Tetapi karena sistem ini adalah yang pertama kali serta satu – satunya didunia yang berfungsi secara penuh saat ini dan dapat digunakan setiap saat oleh setiap orang dibelahan dunia secara gratis, maka nama GPS menjadi terkenal dan sering dipakai sebagai alat navigasi berbasis satelit. Sistem ini menggunakan sejumlah satelit yang berada deorbit yang memancarkan sinyalnya ke bumi dan ditangkap oleh sebuah alat penerima.

Konsep dasar pada penentuan posisi dengan GPS adalah reseksi (pengikatan ke belakang) dengan jarak, yaitu dengan pengukuran jarak secara simultan ke beberapa satelit GPS yang koordinatnya telah diketahui. Pada pelaksanaan pengukuran penentuan posisi dengan GPS, pada dasarnya ada dua jenis alat penerima sinyal satelit (*receiver*) GPS yang dapat digunakan, yaitu :

- a. Tipe navigasi digunakan untuk penentuan posisi yang tidak menuntut ketelitian tinggi
- b. Tipe *geodetic* digunakan untuk penentuan posisi yang menuntut ketelitian tinggi

10. Android

Android adalah sebuah sistem operasi untuk perangkat *mobile* berbasis linux yang mencakup sistem operasi, *middleware* dan aplikasi. Android menyediakan *platform* yang berbeda bagi para pengembang untuk mnciptakan aplikasi mereka.

Awalnya Google Inc. membeli Android Inc. yang merupakan pendatang baru yang membuat peranti lunak untuk ponsel/*smartphone*. Kemudian untuk mengembangkan Android, dibentuklah *Open Handset Alliance*, konsorium dari 34 perusahaan peranti keras, peranti lunak dan telekomunikasi, termasuk Google, HTC, Intel, Motorola, Qualcomm, T-Mobile dan Nvidia (Safaat H,2011).

Android SDK (*Software Development Kit*) menyediakan alat dan API (*Application Programming Interface*) yang diperlukan untuk memulai mengembangkan aplikasi pada *platform* Android menggunakan bahasa pemrograman Java (Android,2013).

11. Google Maps

Google Maps adalah sebuah jasa peta globe virtual gratis dan *online* disediakan oleh Google dapat ditemukan di <http://maps.google.com>. Google Maps menawarkan peta yang dapat diseret dan gambar satelit untuk seluruh dunia dan baru – baru ini, dan juga menawarkan perencanaan rute dan pencari letak bisnis di U.S., Kanada, Jepang, Hongkong, Cina, UK, Irlandia (hanya pusat kota) dan beberapa bagian Eropa. Google Maps API merupakan aplikasi *interface* yang dapat diakses lewat javascript agar Google Maps dapat ditampilkan pada halaman web yang sedang kita bangun. Untuk dapat mengakses Google Maps, kita harus melakukan pendaftaran Api Key terlebih dahulu dengan data pendaftaran berupa nama domain web yang kita bangun. Kita dapat menambahkan fitur Google Maps dalam web kita sendiri dengan Google Maps API. Google Maps API adalah *library* JavaScript.

BAB III

METODE PENGEMBANGAN SISTEM

A. Metode Pengembangan Sistem

Beberapa metode yang digunakan pada penelitian kali ini meliputi, metode pengumpulan data, analisis sistem yang diawali dengan analisis permasalahan dan dilanjutkan analisis solusi permasalahan, perancangan sistem, pengembangan dan pengujian sistem.

1. Pengumpulan Data

Metode pengumpulan data yang digunakan dalam penelitian ini dilakukan dengan mempelajari literatur – literatur yang sudah ada yaitu dengan cara membaca atau mengambil informasi dari makalah, jurnal ilmiah, buku dan juga memanfaatkan *internet* sebagai sumber informasi, dengan cara melihat informasi yang disediakan oleh situs – situs *website*, forum diskusi, *mailing list* dan lain sebagainya. Dari studi literatur yang dilakukan maka akan didapati konsep – konsep mengenai bagaimana sistem indeks diterapkan pada penelitian – penelitian yang lebih dulu. Selain itu dari studi literature tersebut juga didapati bagaimana teori – teori mengenai teknik – teknik pemrograman yang diterapkan pada sistem indeks.

2. Analisis Sistem

Tahap ini dimaksudkan untuk memperoleh gambaran tentang sistem indeks yang akan dibuat, menganalisis keadaan – keadaan atau faktor – faktor yang dapat

Mempengaruhi kinerja sistem. Analisis sistem dimulai dengan menganalisis permasalahan – permasalahan yang ada dan kemudian memberikan solusi dari permasalahan yang terjadi. Analisis permasalahan didapatkan setelah pengumpulan data dalam penelitian dilakukan, dari permasalahan – permasalahan yang ada selanjutnya, menganalisis solusi dari setiap permasalahan yang terjadi.

3. Perancangan dan Pemrograman Sistem

Perancangan sistem dibangun dengan menggunakan pendekatan berorientasi objek (*object oriented*), untuk pendekatan berorientasi objek perancangan menggunakan diagram – diagram dalam UML (*Unified Modeling Language*) yang dapat menjelaskan/menerangkan bagaimana langkah – langkah setiap proses terjadi dengan berorientasi objek. Selanjutnya pada tahap perancangan sistem dilakukan juga perancangan antarmuka (*interface*) untuk tampilan sistem. Sistem yang dibangun berbasis *website* dan *android*, untuk pemrograman sistem yang digunakan adalah *php5* yang mendukung pemrograman berorientasi objek untuk *website*, dan *java* untuk pemrograman *android*.

4. Pengembangan dan Pengujian Sistem

Pengembangan sistem yang akan terus berlanjut ketika saat penelitian berlangsung, dan pengujian dilakukan dalam dua jenis pengujian yaitu *alpha* dan *beta*, pengujian *alpha* (*alpha testing*) dilakukan pada saat program dibuat dan pengujian ini dilakukan oleh *programmer* atau dalam hal ini penulis sendiri, dan jenis pengujian yang dilakukan adalah *white-box* dan *black-box*, dan pengujian *beta* dilakukan pada saat sistem selesai dibangun dan pengujian ini dilakukan oleh pengguna sistem, dan jenis pengujian yang dilakukan adalah *black-box*.

B. Alat Penelitian

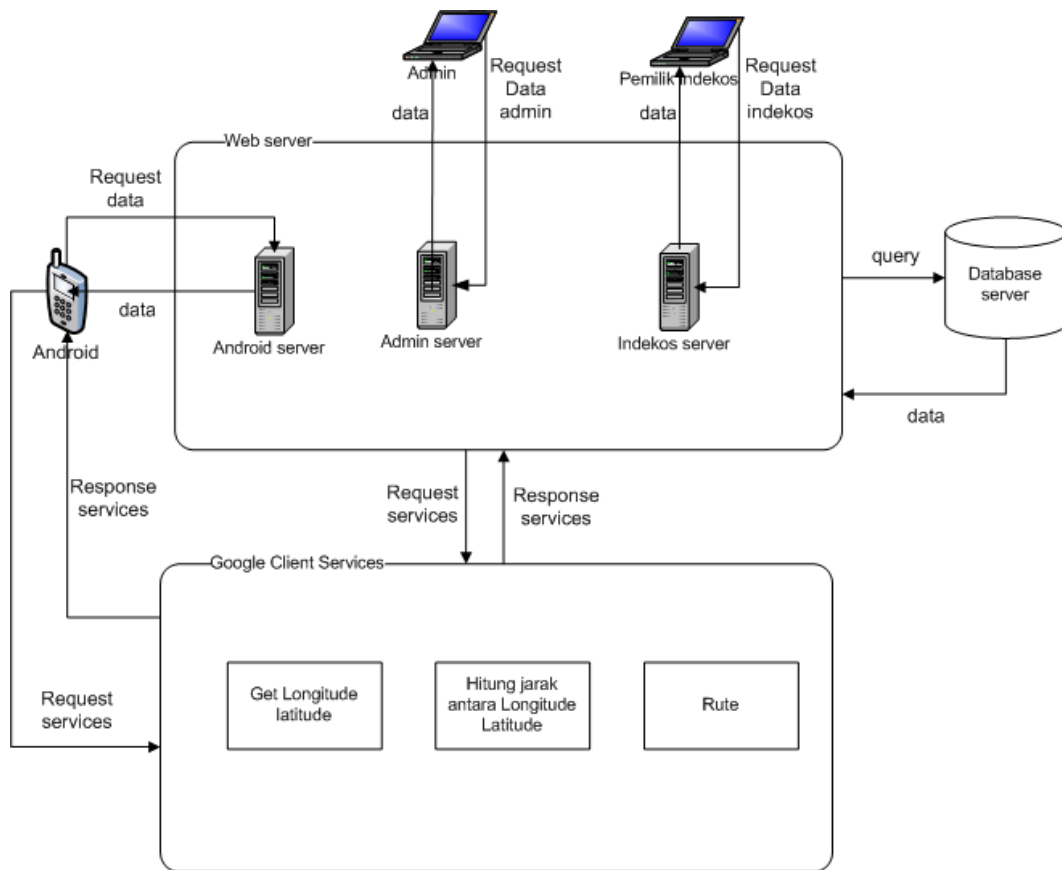
1. Perangkat Lunak (*Software*)

- a. *Eclipse Indigo & SDK Android 4.0*
- b. *Microsoft office visio 2007*
- c. *Windows 7 Professional 32 bit*
- d. *Paint*
- e. *Microsoft Word 2007*
- f. *XAMPP 1.7.4*

2. Perangkat Keras (*Hardware*)

- a. *Laptop Lenovo SL410*
- b. *Ram 2.00 GB*
- c. *Prosesor Intel (R) Core (TM) 2 Duo*
- d. *CPU P8700 @ 2.53GHz 2.53GHz*

C. Arsitektur Sistem Indekos



Gambar 3.1 (Aristektur sistem indekos)

Dari arsitekrut diatas untuk admin dan pemilik indekos hanya dapat mengakses *web server* untuk mengolah datanya, dan untuk pengguna android untuk mengoperasikan data dapat lewat *database client* SQLite yang ada pada android dan dapat mengambil atau sinkronisasi data pada *server* dan untuk mengolah rute indekos, mendapatkan *longitude* dan *latitude* dan menghitung jarak menggunakan *Google client services* yang datanya dikirim dari *database server* dan dari perangkat *web server* atau GPS yang ada pada android *mobile*.

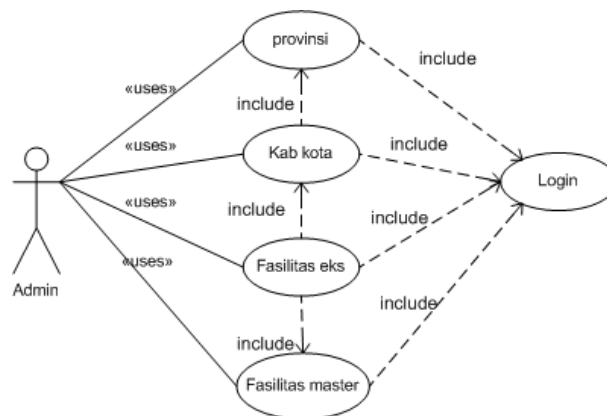
BAB IV

ANALISIS DAN PENGEMBANGAN SISTEM

A. Unified Modeling Language (UML)

1. Diagram *Use Case*

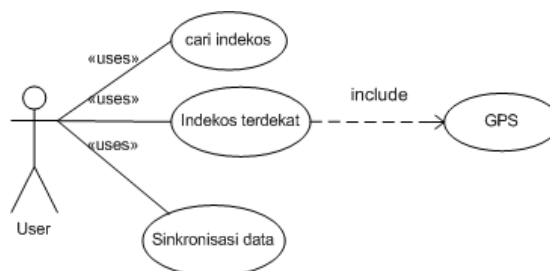
a. Admin indekos



Gambar 4.1 (Diagram *Use Case* admin indekos)

Dari diagram *Use Case* admin memiliki perilaku pada provinsi dan kab kota, fasilitas master, fasilitas eksternal, dan semua perilaku/proses yang terjadi dengan admin membutuhkan perilaku/proses login.

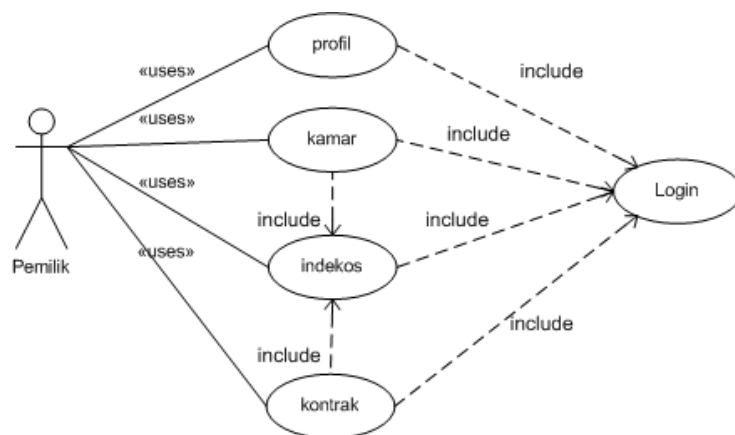
b. Pengguna indekos



Gambar 4.2 (Diagram *Use Case* pengguna indekos)

Dari diagram *use case* pengguna indekos mempunyai prilaku/proses yaitu cari indekos, indekos terdekat yang membutuhkan GPS dan sinkronisasi data, data diambil dari *server* indekos yang datanya telah dimasukan oleh pemilik indekos dan admin. dan semua prilaku/proses yang terjadi tidak membutuhkan prilaku/proses login.

c. Pemilik indekos



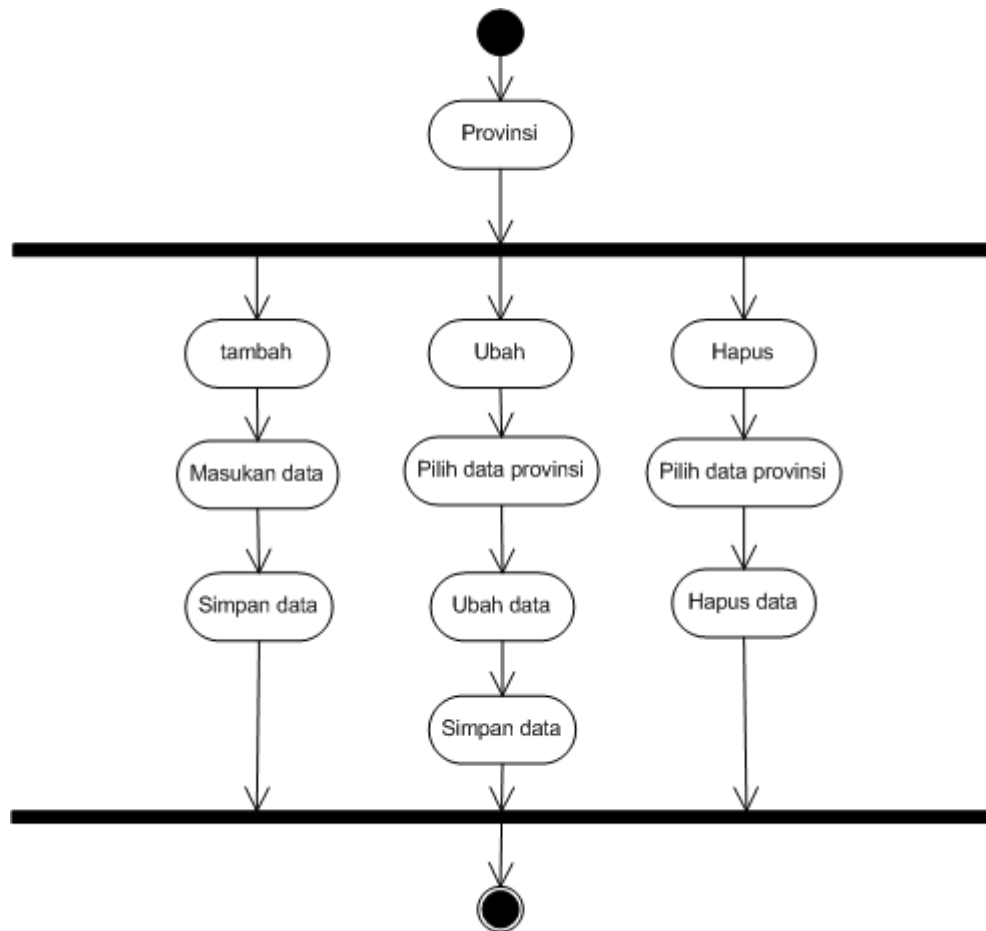
Gambar 4.3 (Diagram *Use Case* pemilik indekos)

Dari diagram *use case* pemilik indekos terdapat prilaku profil/data pribadi, indekos, kamar dan kontrak, dan semua prilaku ini membutuhkan prilaku/proses login.

2. Diagram Aktivitas

Setelah menemukan prilaku apa saja yang dilakukan oleh pengguna (*user*) dalam sistem dengan menggunakan diagram *Use Case*, mengubah setiap aktivitas pengguna yang lebih rinci kedalam diagram aktivitas (*Activity Diagram*), dan pada tahap ini akan didapatkan hasil alur yang terjadi ketika aktivitas tersebut berjalan.

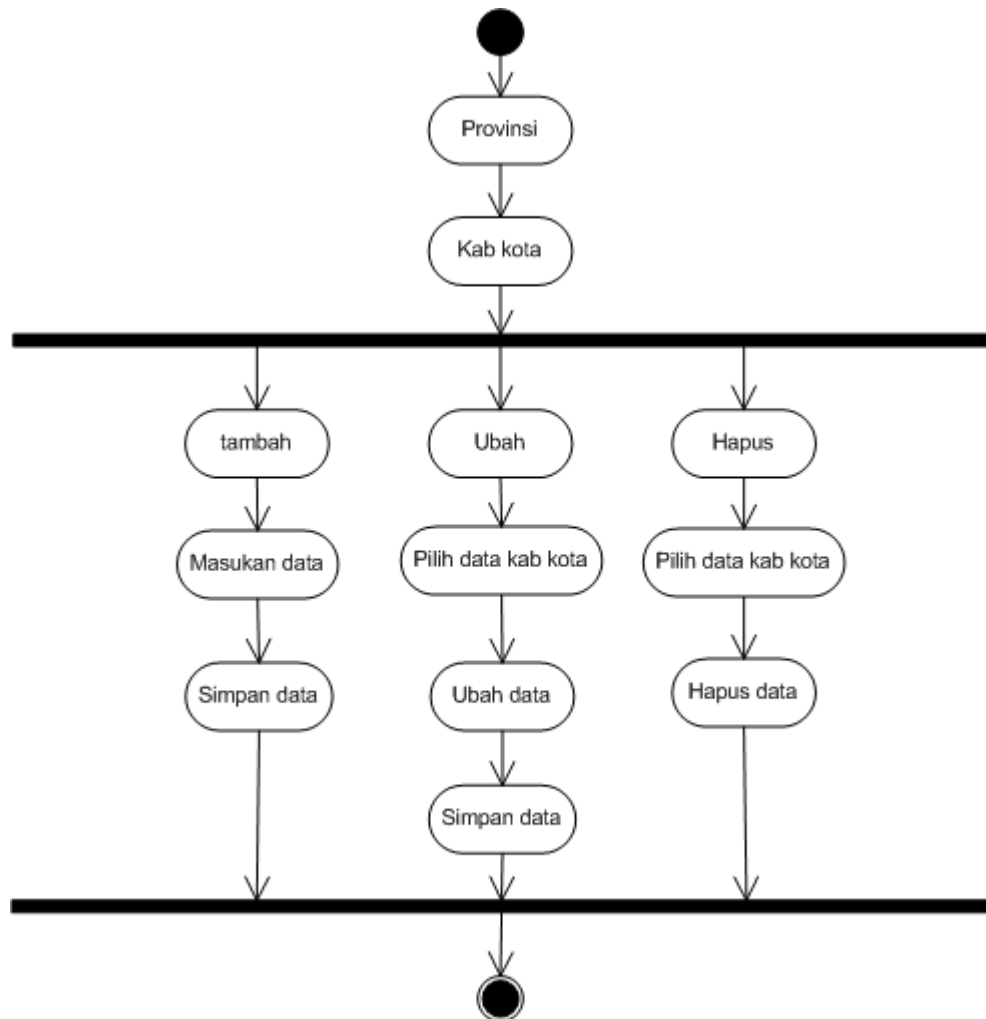
a. Admin provinsi



Gambar 4.4 (Diagram Aktivitas admin provinsi)

Dari diagram aktivitas diatas dapat dilihat bahwa pengguna(admin) pada proses/operasi provinsi dapat melakukan tiga aktivitas yaitu tambah provinsi, ubah provinsi dan hapus provinsi. Pada aktivitas tambah dan ubah terdapat proses cek kesesuaian data yang disebut sesuai adalah ketika inputan yang dibutuhkan pada *field* tersebut angka maka masukan data harus angka dan jika data tidak boleh kosong maka *field* tersebut harus terisi jika benar maka data sudah sesuai dengan yang dibutuhkan. Semua aktivitas yang terjadi pada admin ini langsung berhubungan dengan *server* indekos.

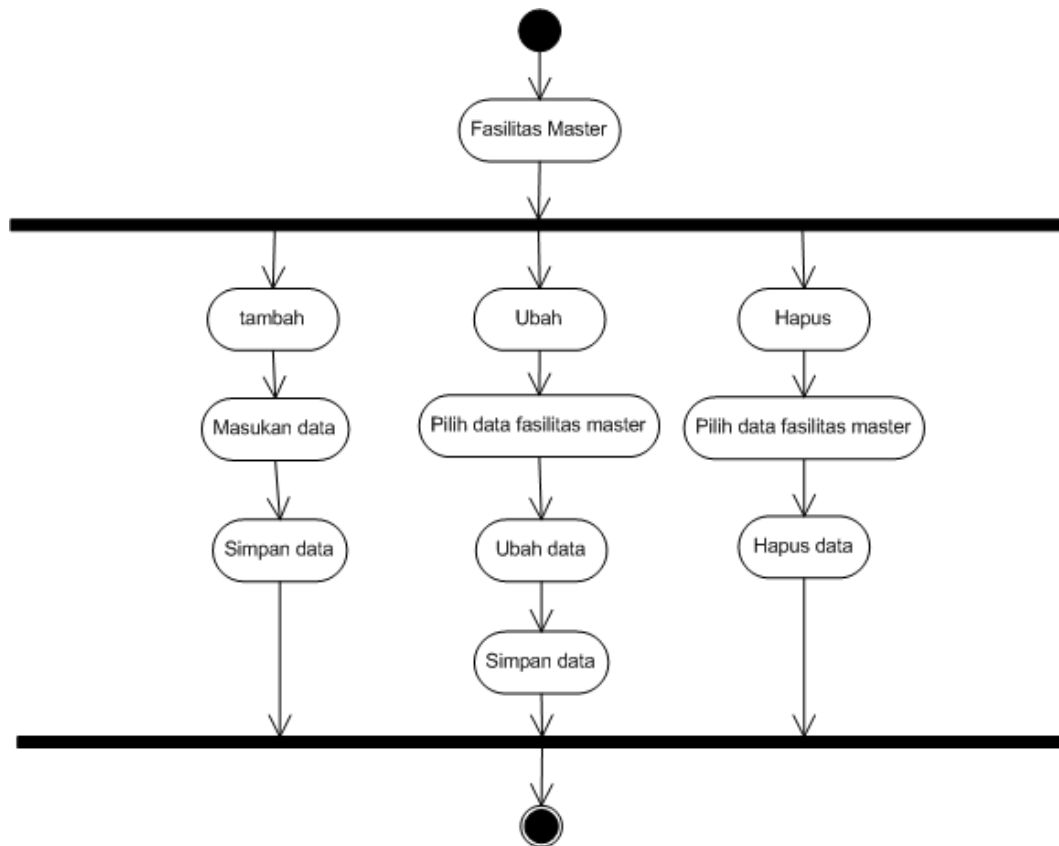
b. Admin kab kota



Gambar 4.5 (Diagram Aktivitas admin kab kota)

Diagram aktivitas admin kab kota diatas pada operasi kab kota memiliki tiga aktivitas yaitu tambah kab kota, ubah kab kota dan hapus kab kota, proses/operasi kab kota terjadi saat berada pada proses/operasi provinsi dan memilih provinsi id pada kota yang diinginkan. Aktivitas admin kab kota ini langsung terhubung dengan *server* indekos.

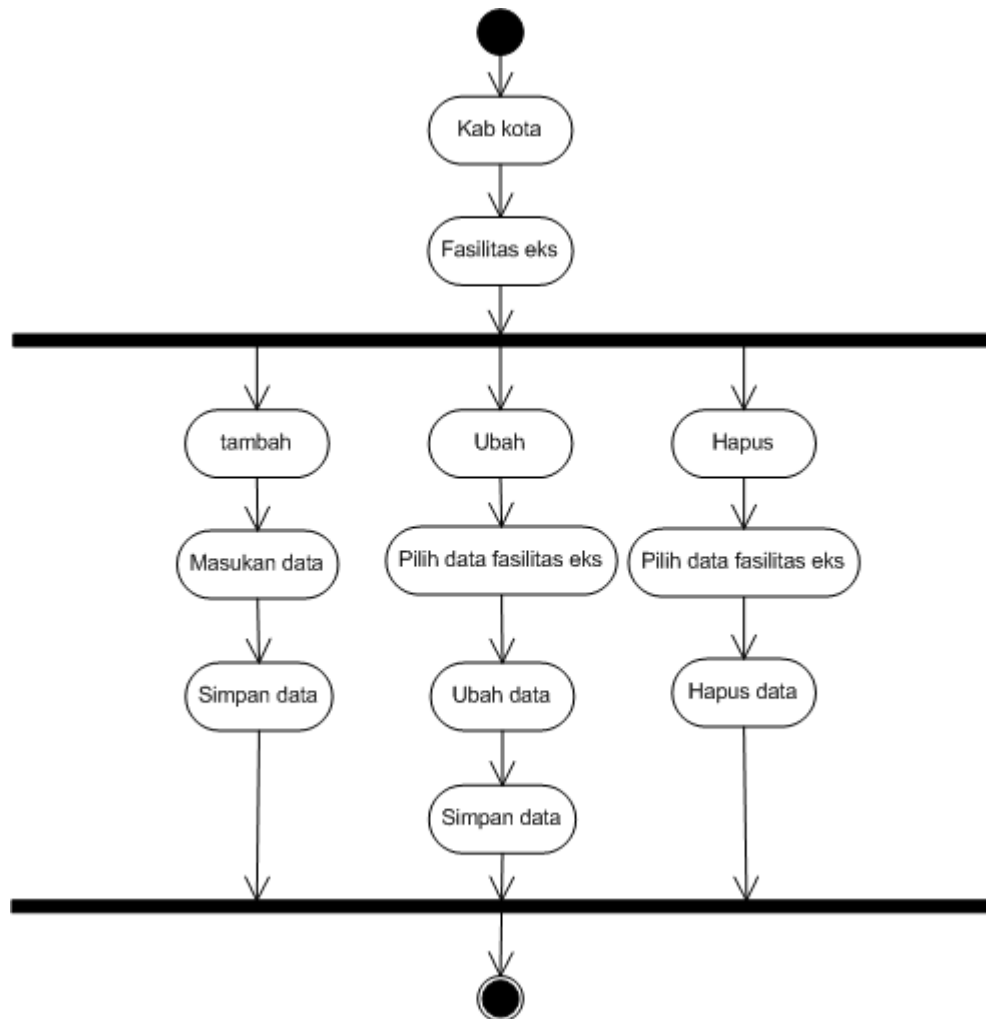
c. Admin fasilitas master



Gambar 4.6 (Diagram aktivitas admin fasilitas master)

Diagram aktivitas admin fasilitas master diatas mempunyai 3 prilaku/aktivitas yang sama seperti aktivitas provinsi, aktivitas yang ada pada fasilitas master adalah tambah fasilitas master, ubah fasilitas master, dan hapus fasilitas master. Aktivitas yang terjadi pada fasilitas provinsi ini berlangsung pada *server* indekos, fasilitas master digunakan untuk mengkategorikan fasilitas eksternal.

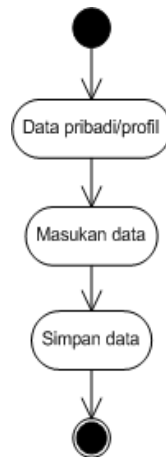
d. Admin fasilitas eksternal



Gambar 4.7 (Diagram Aktivitas admin fasilitas eksternal)

Diagram aktivitas admin fasilitas eksternal diatas memiliki tiga perilaku yang sama seperti kab kota, aktivitas itu adalah tambah, ubah dan hapus fasilitas eksternal. Seperti dilihat dalam diagram sebelum melakukan operasi pada fasilitas eksternal ini, diharuskan untuk memilik kab kota terlebih dahulu, kab kota tersebut digunakan untuk menentukan fasilitas tersebut berada di kab kota mana.

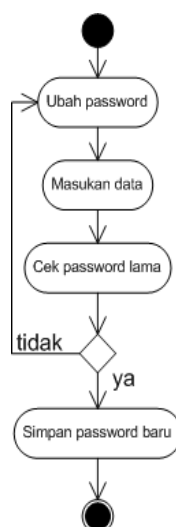
e. Pemilik data pribadi/profil



Gambar 4.8 (Diagram aktivitas pemilik data pribadi/profil)

Diagram aktivitas pemilik data pribadi/profil diatas hanya perilaku yang dimiliki hanya memasukan data dan menyimpan data. Aktivitas pemilik data pribadi/profil ini terjadi dan berhubungan dengan *server* indekos.

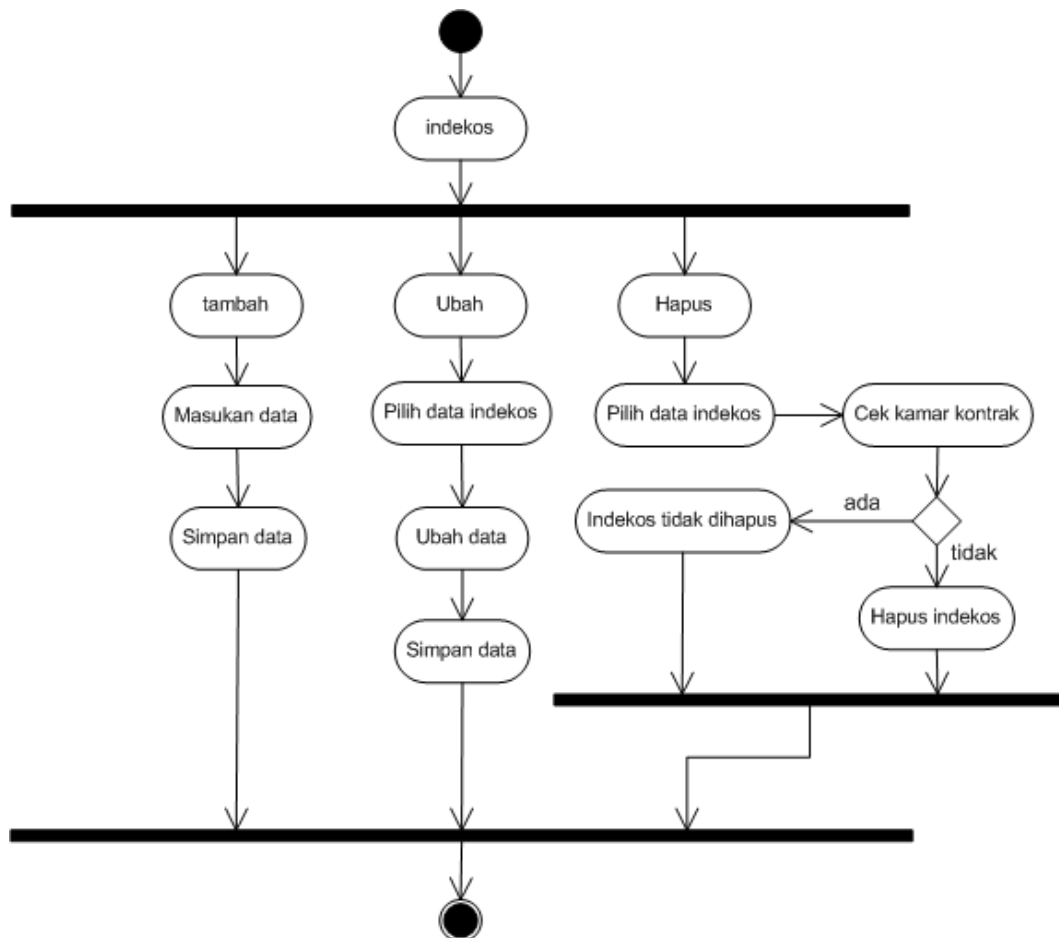
f. Pemilik ubah password



Gambar 4.9 (Diagram aktivitas pemilik ubah password)

Diagram aktivitas pemilik ubah password diatas, dapat dilihat bahwa aktivitas yang terjadi memasukan data, dan cek password lama setelah itu simpan data.

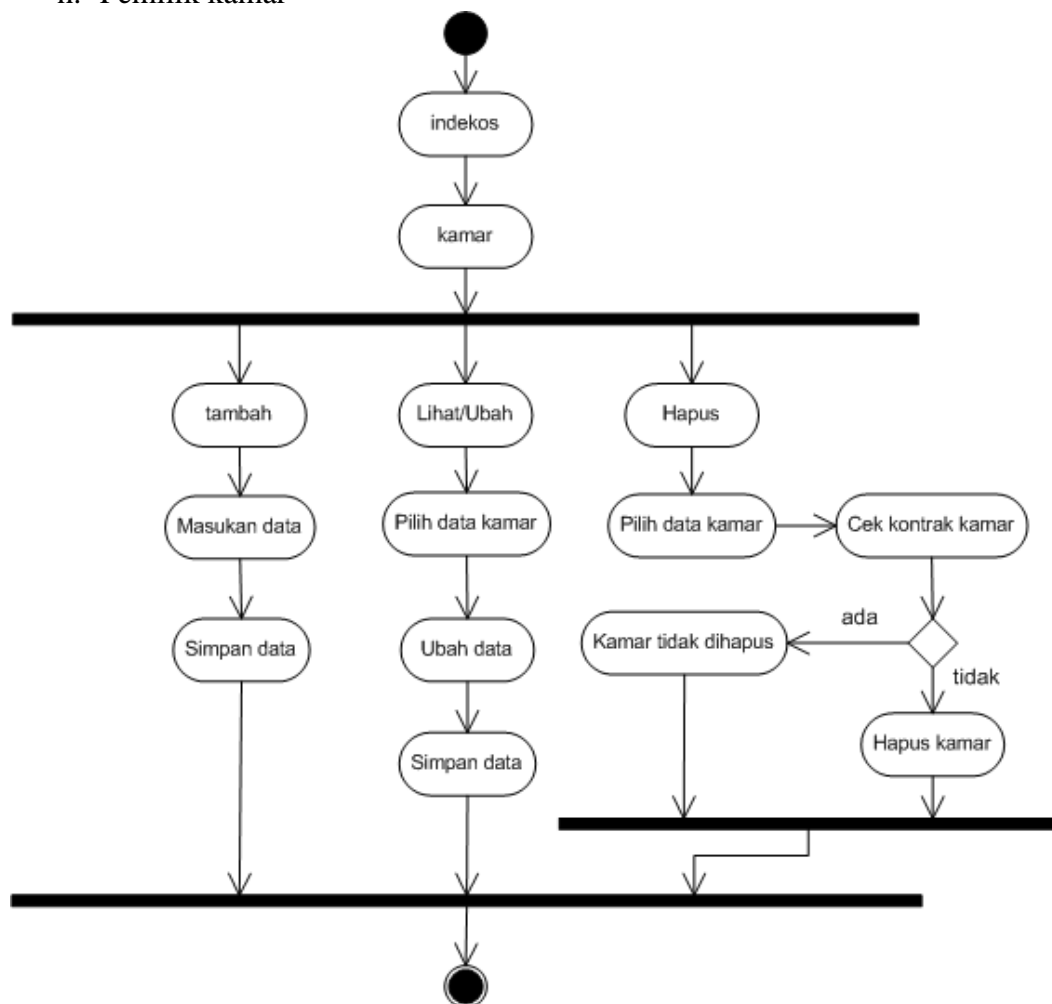
g. Pemilik indekos



Gambar 4.10 (Diagram aktivitas pemilik indekos)

Diagram aktivitas pemilik indekos diatas memiliki 3 prilaku/aktivitas yaitu tambah, ubah dan hapus. Pada prilaku/aktivitas hapus indekos proses yang terjadi pertama pilih ID indekos, selanjutnya melakukan cek kontrak kamar apakah kamar pada indekos tersebut ada yang dikontrakan atau tidak, jika ada maka indekos tersebut tidak dapat dihapus dan jika tidak ada kamar yang dikontrakan maka data indekos dan semua kamar yang ada pada indekos tersebut dan juga fasilitas internal yang ada pada setiap kamar dihapus.

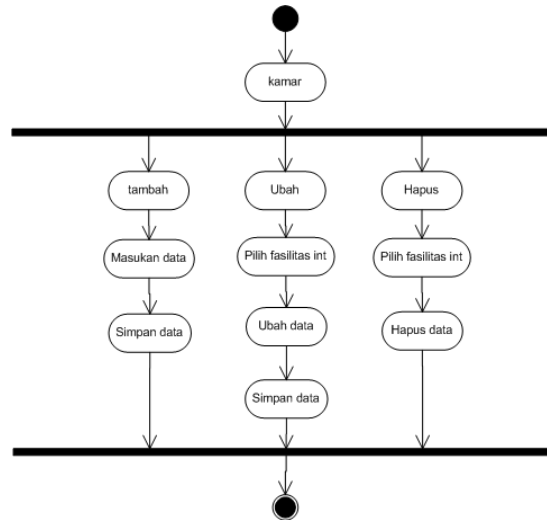
h. Pemilik kamar



Gambar 4.11 (Diagram aktivitas pemilik kamar)

Diagram aktivitas pemilik kamar diatas mempunyai 3 aktivitas yaitu tambah ubah dan hapus kamar. Pada aktivitas hapus kamar terlebih dahulu pilih ID kamar yang akan dihapus. Kemudia cek apakah ID apakah kamar tersebut dikontrak atau tidak, jika tidak dikontrakan maka data kamar dan fasilitas internal yang ada dikamar tersebut dihapus.

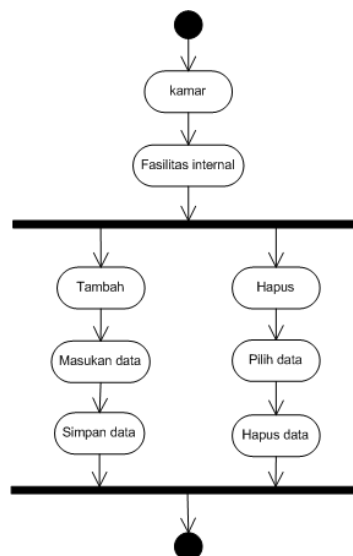
i. Pemilik fasilitas internal



Gambar 4.12 (Diagram aktivitas fasilitas internal)

Diagram aktivitas fasilitas internal memiliki 3 aktivitas atau perilaku yaitu tambah, ubah dan hapus fasilitas internal.

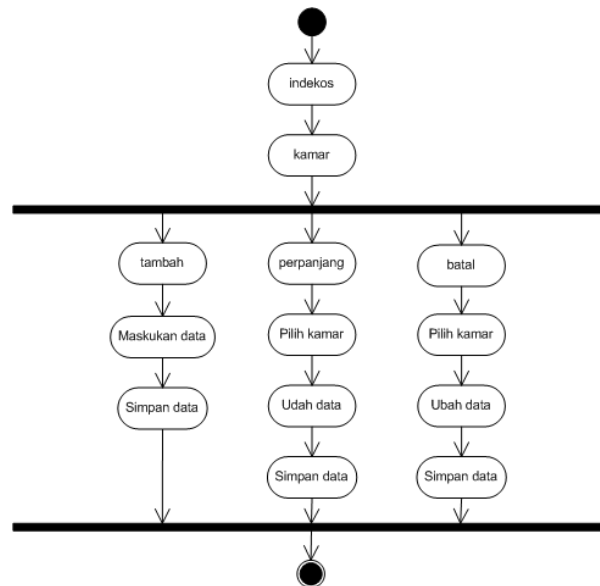
j. Pemilik kamar fasilitas internal



Gambar 4.13 (Diagram aktivitas kamar fasilitas internal)

Diagram aktivitas kamar fasilitas internal memiliki 2 aktivitas yaitu tambah dan hapus kamar fasilitas internal.

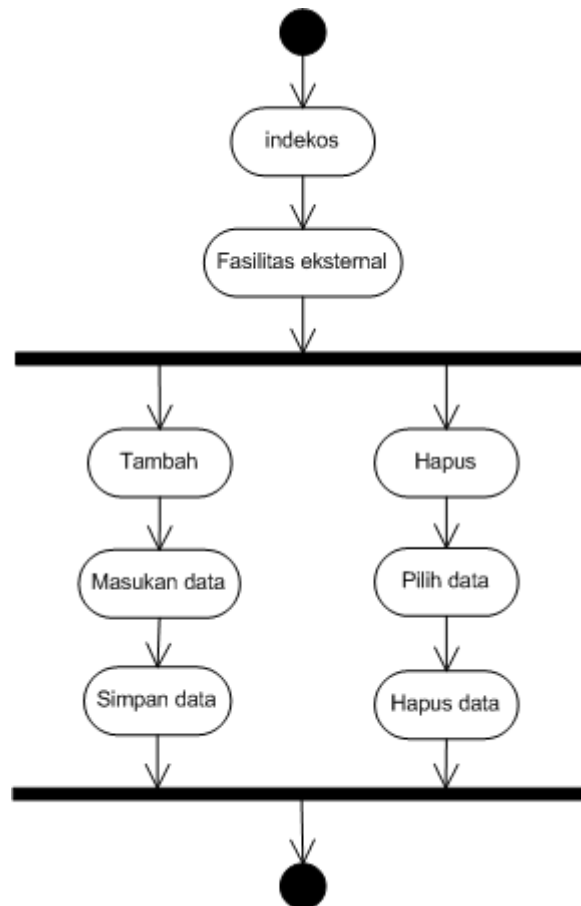
k. Pemilik kontrak



Gambar 4.14 (Diagram aktivitas pemilik kontrak)

Diagram aktivitas pemilik kontrak diatas memiliki 3 aktivitas/prilaku yaitu tambah/buat kontrak, perpanjang kontrak dan batal kontrak. Semua aktivitas terjadi setelah pemilik memilih ID kamar dan setelah itu cek status kontrak kamar apakah masih dikotrak atau tidak, jika tidak maka aktivitas yang bisa dilakukan hanya tambah/buat kontrak baru dan jika status kontrak kamar masih dikontrak maka aktivitas yang ada adalah perpanjang kontrak dan batal kontrak, pada aktivitas batal kontrak data status kontrak hanya diubah menjadi kosong dan data tidak dihapus. Pada aktivitas kontrak ini tidak ada *history* (Sejarah) kontrak yang sudah terjadi.

1. Pemilik indekos fasilitas eksternal

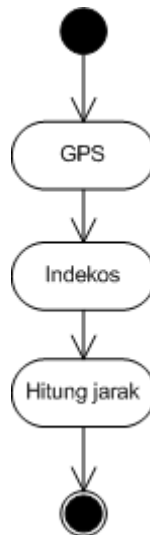


Gambar 4.15 (Diagram aktivitas indekos fasilitas eksternal)

Diagram aktivitas indekos fasilitas eksternal diatas memiliki 2 aktivitas/prilaku yaitu tambah dan hapus indekos fasilitas eksternal. Semua aktivitas indekos fasilitas eksternal terjadi setelah pemilik memilih ID indekos. Pada aktivitas tambah data yang dibutuhkan adalah data ID indekos dan ID fasilitas eksternal yang ingin dimasukan dalam indekos fasilitas eksternal, dan fasilitas eksternal tersebut adalah data yang dimasukan oleh admin dan bukan pemilik, jadi ketika aktivitas hapus data yang dihapus adalah indekos fasilitas

eksternal data tersebut adalah relasi antara indekos pemilik dan fasilitas eksternal yang dimasukan.

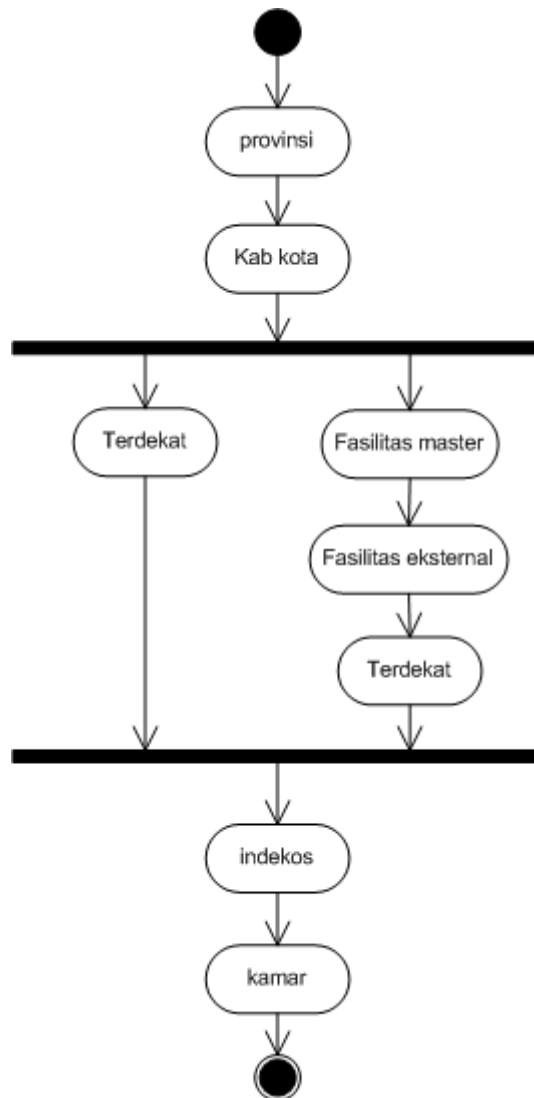
m. Pengguna indekos terdekat



Gambar 4.16 (Diagram aktivitas pengguna indekos terdekat)

Diagram aktivitas pengguna indekos terdekat diatas memiliki aktivitas/prilaku yaitu mencari jarak terdekat. Aktivitas diatas bergantung pada status GPS *smartphone* android jika status mati maka aktivitas/prilaku indekos terdekat tidak bisa dijalankan. Dan jika status aktif maka maka aktivitas pencarian indekos terdekat dapat dilakukan. Aktivitas terjadi di *client* dan mengambil data di *server* indekos.

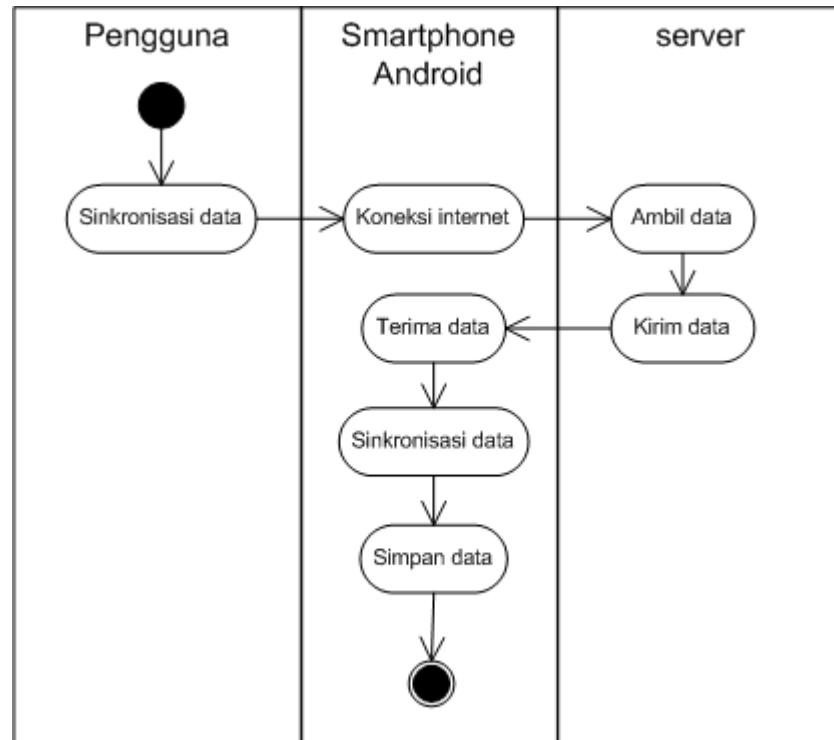
n. Pengguna pencarian indekos kamar



Gambar 4.17 (Diagram aktivitas pengguna pencarian indekos kamar)

Diagram aktivitas pengguna pencarian indekos kamar 2 aktivitas yaitu pencarian berdasarkan fasilitas terdekat, dan pencarian berdasarkan kab kota terdekat. Pada diagram pengguna pencarian indekos kamar ini data hanya mengambil dari *database client* SQLite yang datanya sudah diambil terlebih dahulu oleh pengguna.

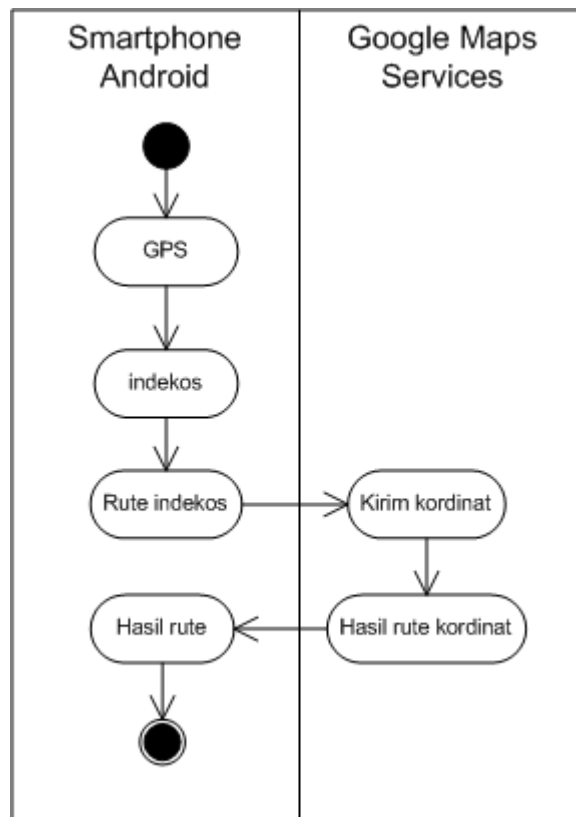
o. Pengguna sinkronisasi data



Gambar 4.18 (Diagram aktivitas pengguna sinkronisasi data)

Diagram aktivitas pengguna sinkronisasi data diatas memiliki aktivitas/prilaku yaitu sinkronisasi data, sinkronisasi data tersebut berinteraksi langsung dengan *server* dan dikirim ke aplikasi *client smartphone* android, dan sebelum data disimpan data terlebih dahulu cek data dari *server* dengan data yang ada pada *client*, jika benar data=0 maka data ditambah pada *client*, jika salah dan jika data=1 maka data *client* diubah dengan data *server*, dan jika salah dan data>1 maka data dari *client* dihapus kemudian ditambah data baru dari *server*.

p. Pengguna rute indekos



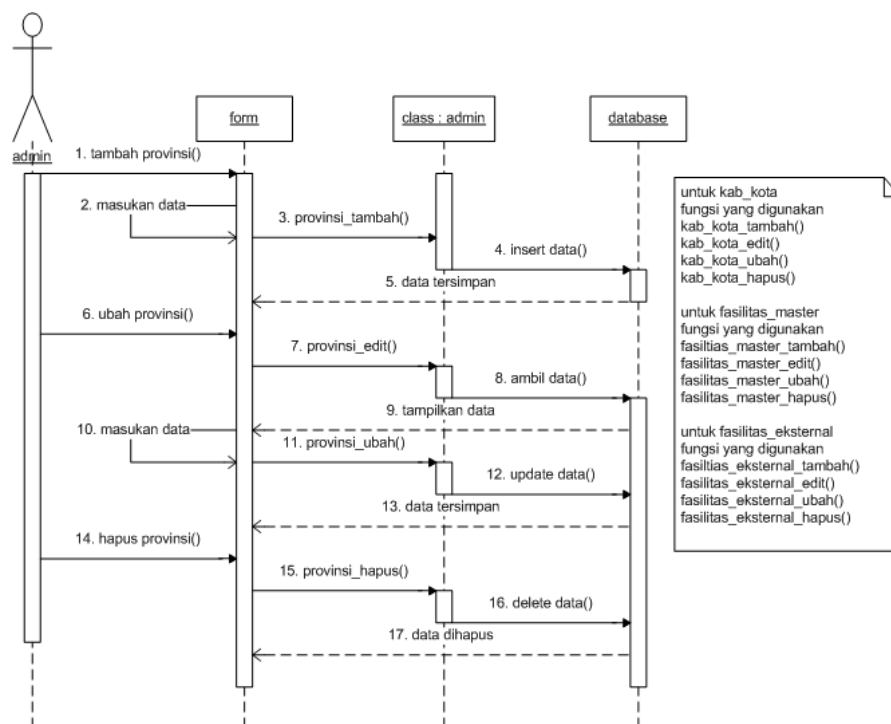
Gambar 4.19 (Diagram aktivitas pengguna rute indekos)

Diagram aktivitas pengguna rute indekos diatas ini memiliki aktivitas/prilaku yaitu mengambil data rute indekos dan aktivitas ini memerlukan fasilitas GPS yang ada pada *smartphone* android, aktivitas dimulai dari daftar indekos, daftar indekos dapat ambil bisa dari *database client* SQLite android yang datanya sudah diambil terlebih dahulu oleh pengguna, selanjutnya data diolah menggunakan *google client services* untuk mendapatkan hasil rute indekos, dan untuk menampilkan rute indekos.

3. Diagram Sekuensial

Diagram Sekuensial (*Sequential Diagram*) dapat menerangkan atau menjelaskan dari diagram aktivitas yang terjadi berdasarkan waktu, dan dapat juga melanjutkan dari diagram *Use Case* yang terjadi berdasarkan waktu terjadinya. Diagram sekuensial dapat menjelaskan apa saja yang terjadi dan apa saja yang terlibat pada saat aktivitas tersebut dijalankan, karena diagram sekuensial menggambarkan sebuah aktivitas berdasarkan waktu terjadinya aktivitas.

a. Admin

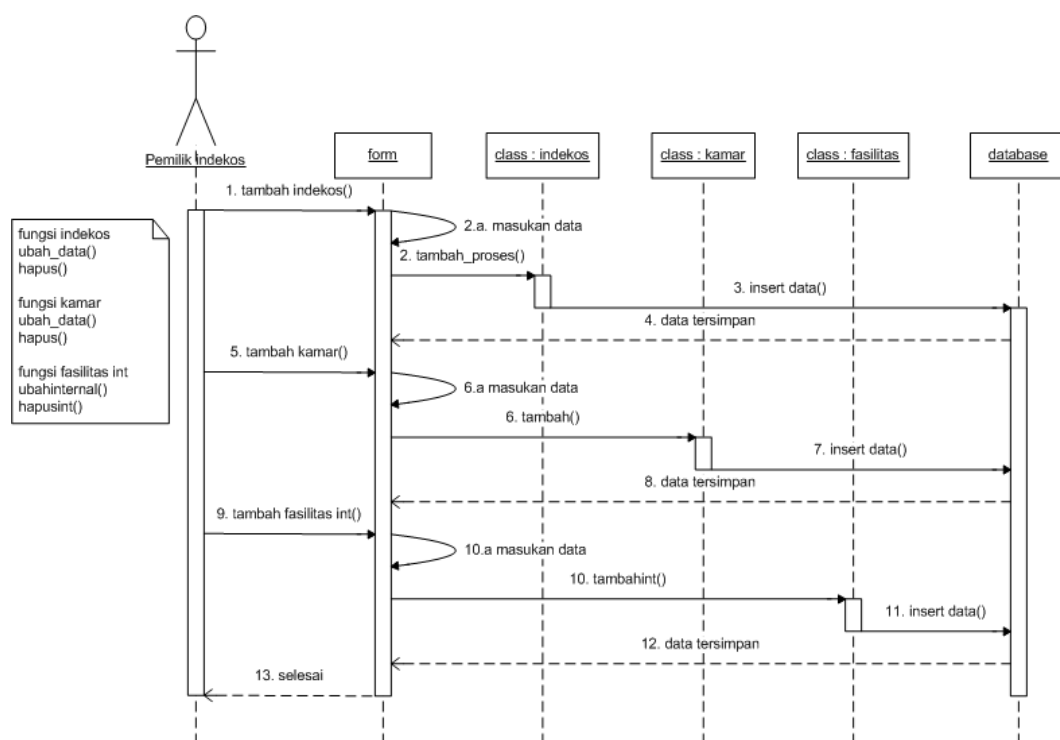


Gambar 4.20 (Diagram sekuensial admin)

Diagram sekuensial admin diatas menunjukan ketika pengguna admin memanggil fungsi proses akan meminta data untuk operasi tambah, ubah dan

hapus data, dan diagram sekuensial admin ini sama untuk proses provinsi, kab kota, fasilitas eksternal dan fasilitas master. Pada diagram sekuensial dapat dilihat objek apa sajakah yang terlibat pada saat proses atau aktivitas berlangsung, pada diagram sekuensial diatas terdapat empat objek yaitu admin, *class* admin, fungsi dan *database*.

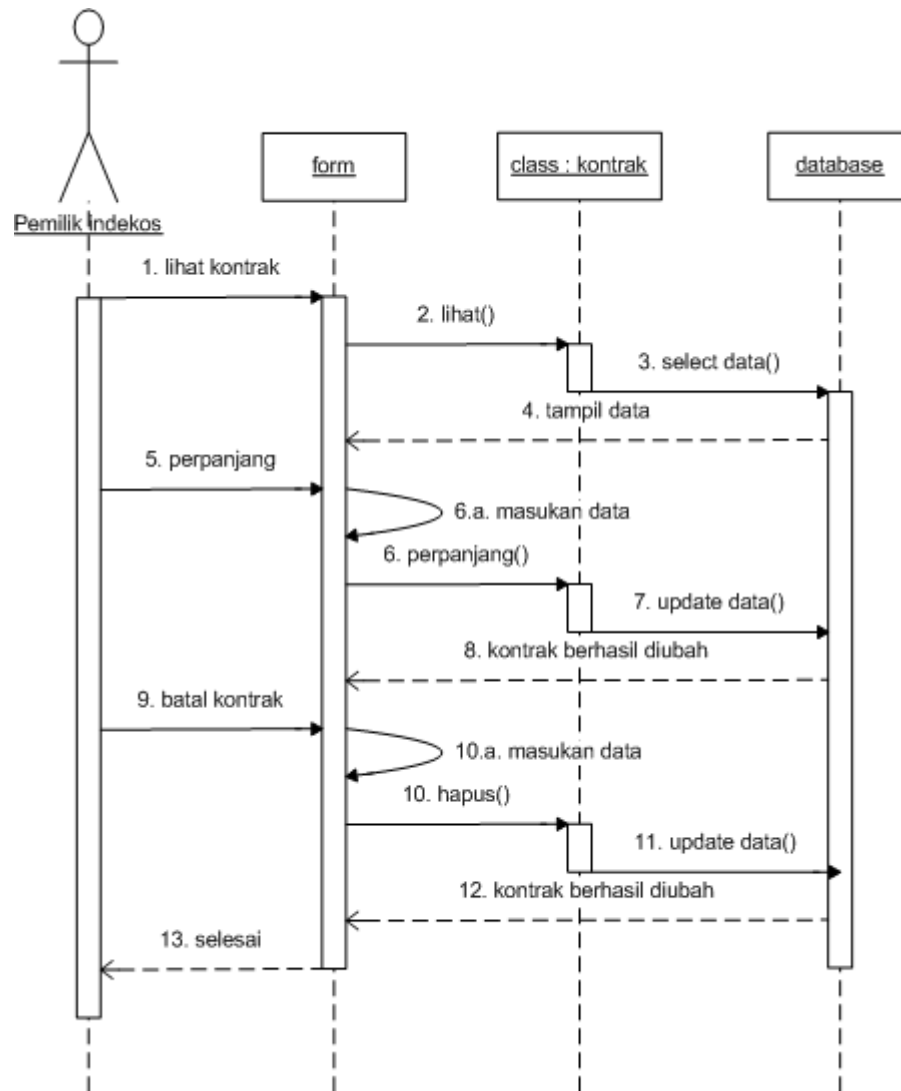
b. Pemilik indekos kamar



Gambar 4.21 (Diagram sekuensial pemilik indekos kamar)

Diagram sekuensial pemilik indekos kamar diatas terdapat 6 objek yang berinteraksi yaitu pemilik indekos, *form*, *class:indekos*, *class:kamar*, *class:fasilitas* dan *database*. Pada diagram diatas hanya menampilkan alur ketika tambah data, untuk ubah dan hapus, proses alur yang terjadi sama, untuk tiap – tiap fungsinya terdapat pada *note* pada bagian samping gambar.

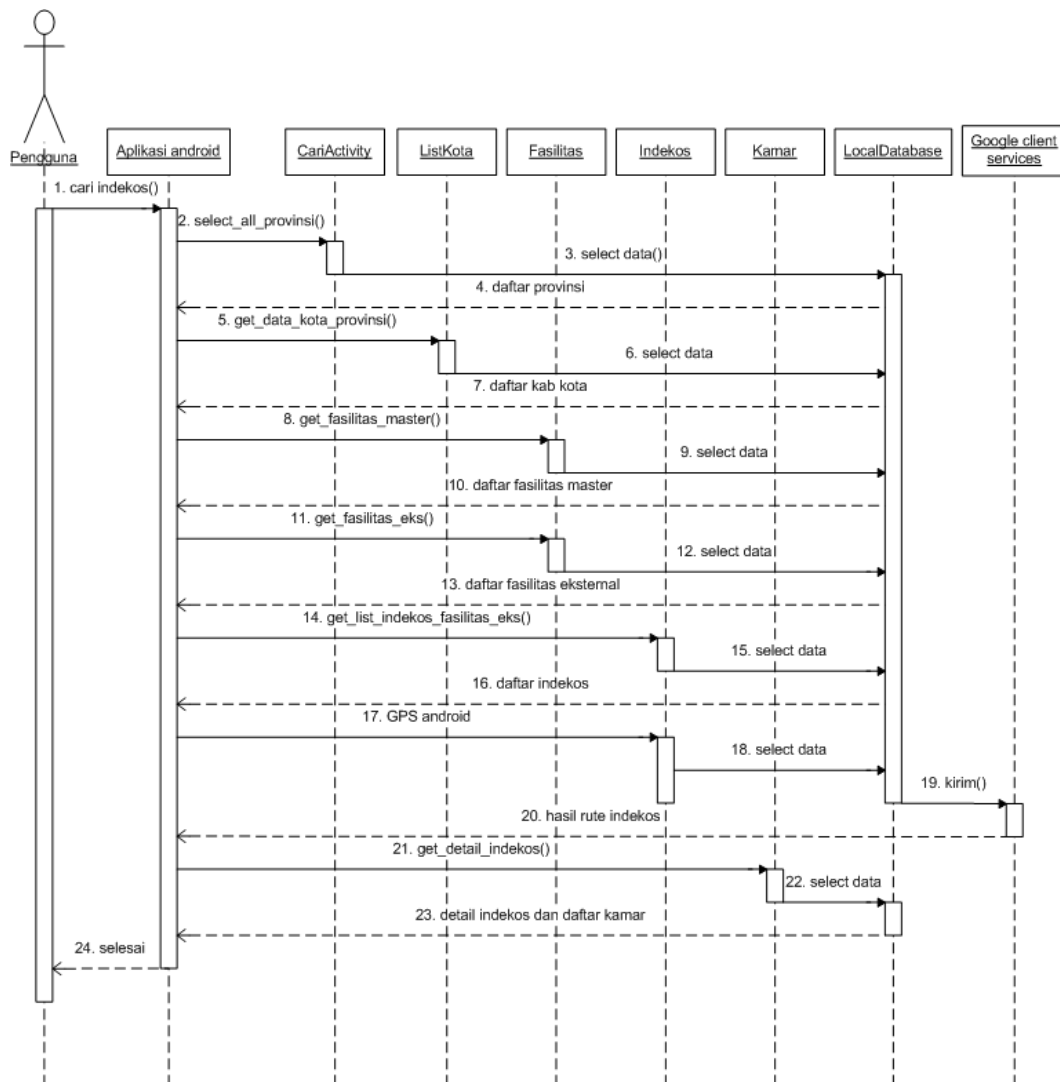
c. Pemilik kontrak



Gambar 4.22 (Diagram sekuensial pemilik kontrak)

Diagram sekuensial pemilik kontrak diatas objek yang terlibat yaitu pemilik indekos, *form*, *class:kontrak* dan *database*. Pada *class:kontrak* data diambil dari tabel indekos dan kamar untuk melihat indekos dan kamar mana saja yang dikontrak, dan untuk melakukan proses perpanjang / batal kontrak.

d. Pengguna cari indekos, rute indekos

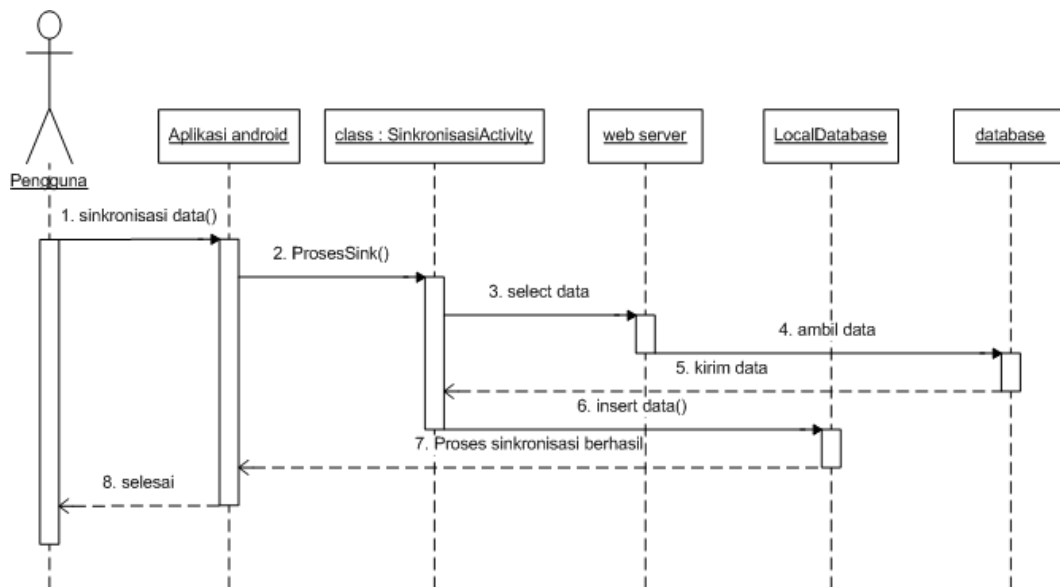


Gambar 4.23 (Diagram sekuensial pengguna cari indekos, rute indekos)

Diagram diatas dapat dilihat bahwa untuk mencari indekos banyak objek yang terlibat yaitu pengguna, aplikasi android, CariActivity, ListKota, Fasilitas, Indekos, Kamar, LocalDatabase dan *google client services*. Aktivitas pencarian indekos dapat melakukan proses pengambilan data dari *database* di aplikasi *client* dan dapat mengakses langsung dari *server* indekos. *Database* yang digunakan

pada server adalah *database* MySQL dan *client* menggunakan *database* SQLite dengan struktur yang sama pada kedua tempat penyimpanan data (*data storage*).

e. Pengguna sinkronisasi data

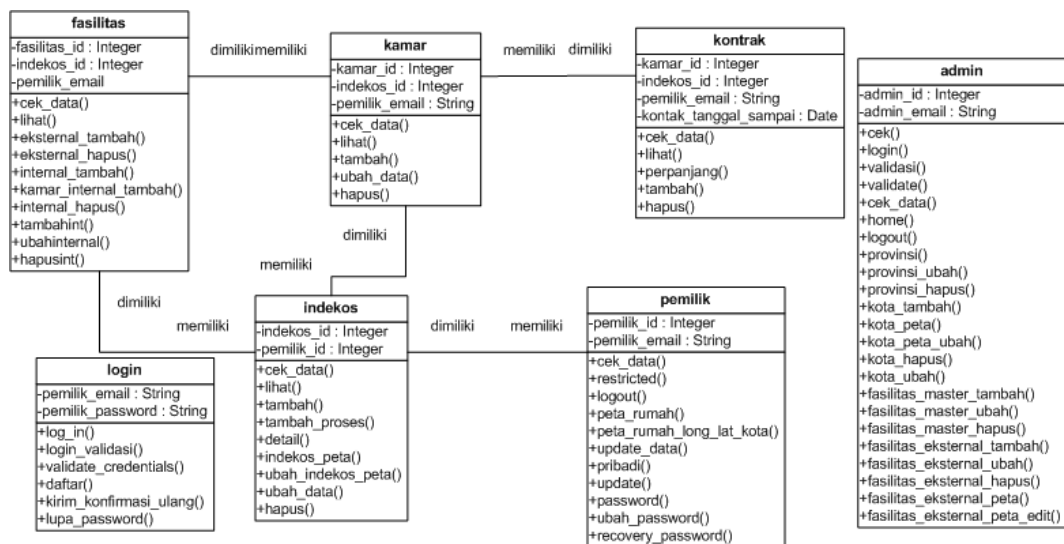


Gambar 4.24 (Diagram sekuensial pengguna sinkronisasi data)

Diagram sekuensial pengguna sinkronisasi data diatas terdapat 6 objek yang digunakan dalam melakukan aktivitasnya yaitu pengguna, aplikasi android, *class:SinkronisasiActivity*, *web server*, *LocalDatabase* dan *database*. Alur yang terjadi pertama pengguna meminta sinkronisasi data pada aplikasi, *class:SinkronisasiActivity* melakukan *prosesSink()* untuk mengambil data pada *web server* lalu aplikasi mengambil data dari *database*, *database* mengirim data yang akan disinkronisasikan dan *class:SinkronisasiAcitvity* memasukan data pada *LocalDatabase*.

4. Diagram Kelas

Diagram kelas dapat menjelaskan relasi antar kelas kelas yang terdapat pada sistem. Dan dapat melihat fungsi fungsi yang terdapat pada kelas tersebut. Didalam diagram kelas ini tidak akan dijelaskan proses – proses yang terjadi disetiap fungsi, hanya akan menjelaskan disetiap kelas terdapat fungsi apa saja.

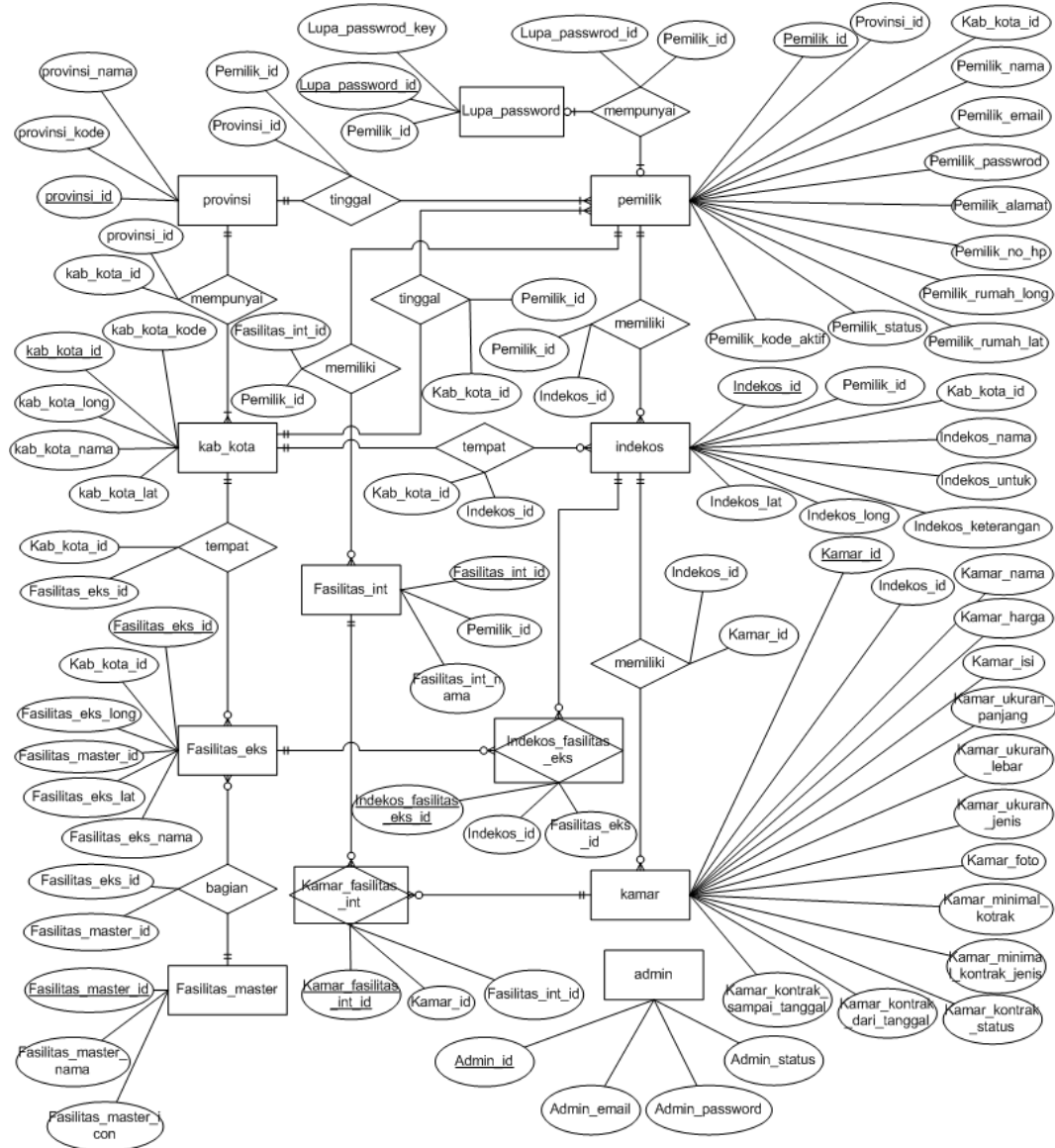


Gambar 4.25 (Diagram kelas)

Diagram kelas diatas terdapat tujuh kelas yaitu admin, pemilik, indekos, kamar, kontrak, fasilitas, login, tidak semua aktivitas/kegiatan yang terjadi dan objek - objek yang ada pada diagram *use case*, diagram aktivitas, dan diagram sekuensial akan menjadi sebuah kelas.

B. Perancangan Basis data (*Database*)

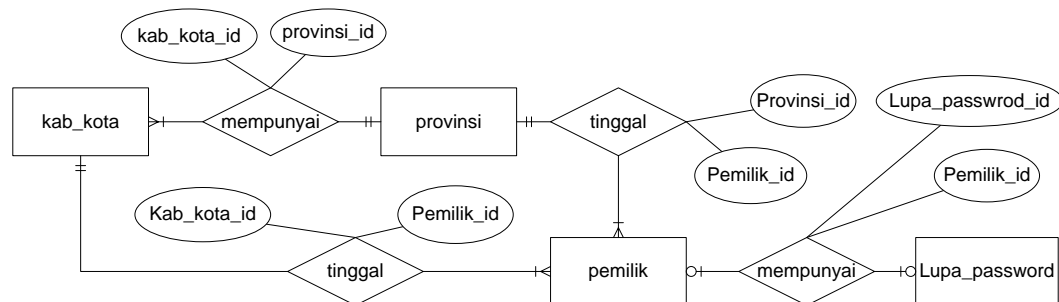
1. Diagram Entity Relationship (E-R)



Gambar 4.26 (Diagram *Entity Relationship* E-R)

Dari diagram *entity relationship* (E-R) ini dapat menjelaskan hubungan – hubungan atau relasi yang ada pada tiap entitas dan jenis relasi antar tiap entitas tersebut. Penjelasan antar relasi sebagai berikut

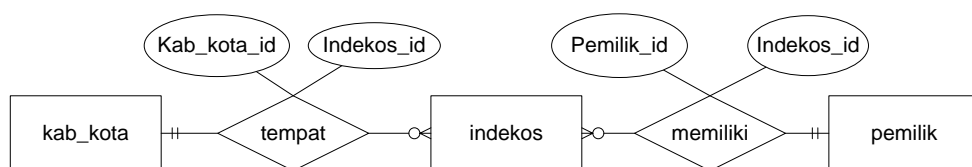
a. Relasi provinsi, kab_kota, pemilik dan lupa_password



Gambar 4.27 (Diagram E-R provinsi, kab_kota dan pemilik)

Dari diagram diatas dilihat bahwa relasi 1(satu) provinsi mempunyai 1-Banyak kab_kota, dan 1-banyak kab_kota dipunyai 1 dan hanya 1 provinsi. Relasi provinsi - pemilik bahwa 1-banyak pemilik tinggal di 1 dan hanya 1 provinsi, dan 1 provinsi bisa ditinggali 1-banyak seorang pemilik. Relasi kab_kota – pemilik bahwa 1-banyak pemilik tinggal di 1 dan hanya 1 kab_kota, dan 1 kab_kota bisa ditinggali 1-banyak seorang pemilik. Dan 1 pemilik mempunyai 0-1 lupa_password, dan 1 lupa_password dipunyai 0-1 pemilik.

b. Relasi kab_kota, indekos dan pemilik

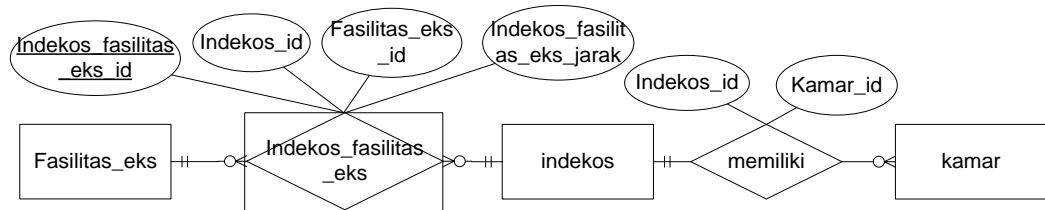


Gambar 4.28 (Diagram E-R kab_kota, indekos dan pemilik)

Relasi dari diagram E-R diatas menerangkan, 1 kab_kota dapat ditempati oleh 0-banyak indekos, dan 1-banyak indekos dapat menempati 1 dan hanya 1

kab_kota. Dan 1-banyak indekos dimiliki 1 dan hanya 1 pemilik, dan pemilik dapat memiliki 0-banyak indekos.

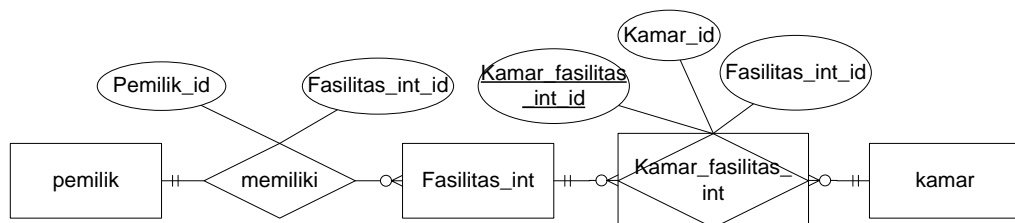
c. Relasi indekos, fasilitas_eks dan kamar



Gambar 4.29 (Diagram E-R indekos, fasilitas_eks dan kamar)

Relasi dari diagram E-R diatas menerangkan, 1-banyak fasilitas_eks dapat dimiliki 0-banyak indekos, dan 1-banyak indekos dapat memiliki 0-banyak fasilitas_eks, relasi antara fasilitas_eks dan indekos menjadi entitas asosiatif dengan nama entitas indekos_fasilitas_eks. Dan 1 indekos memiliki 0-banyak kamar, dan 1-banyak kamar memiliki 1 dan hanya 1 indekos.

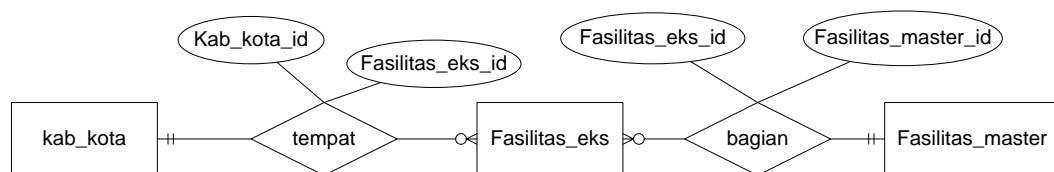
d. Relasi pemilik, fasilitas_int dan kamar



Gambar 4.30 (Diagram E-R pemilik, fasilitas_int dan kamar)

Relasi dari diagram E-R diatas menerangkan, 1 pemilik memiliki 0-banyak fasilitas_int, dan 1-banyak fasilitas_int dapat dimiliki 1 dan hanya 1 pemilik. Dan 1-banyak fasilitas_int dapat dimiliki 0-banyak kamar, dan 1-banyak kamar dapat memiliki 0-banyak fasilitas_int, relasi antara kamar dan fasilitas_int menjadi entitas asosiatif dengan nama entitas kamar_fasilitas_int.

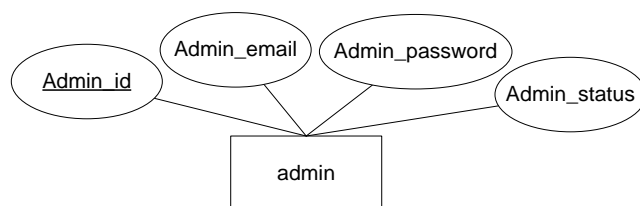
e. Relasi kab_kota, fasilitas_eks dan fasilitas_master



Gambar 4.31 (Diagram E-R kab_kota, fasilitas_eks dan fasilitas_master)

Relasi dari diagram E-R diatas menerangkan, 1 kab_kota dapat ditempati 0-banyak fasilitas_eks, dan 1-banyak fasilitas_eks dapat menempati 1 dan hanya 1 kab_kota. Dan 1-banyak fasilitas_eks adalah bagian dari 1 dan hanya 1 fasilitas_master, dan 1 fasilitas_master adalah sub – sub bagian 0-banyak fasilitas_eks.

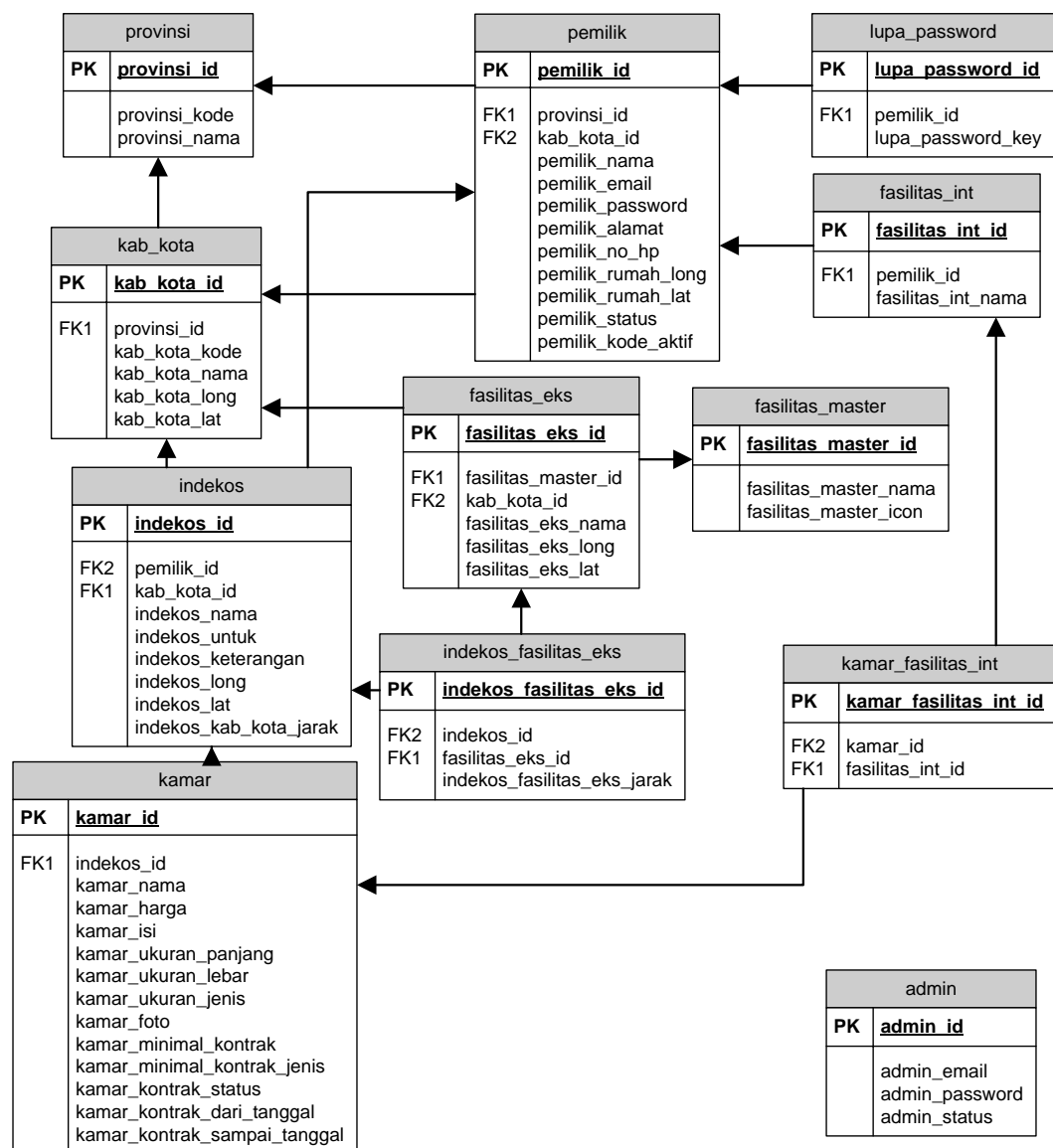
f. Entitas Admin



Gambar 4.32 (Diagram E-R entitas admin)

Entitas admin dalam diagram E-R tidak memiliki relasi antar entitas lain. *Fields* yang ada dalam entitas ini adalah `admin_id`, `admin_email`, `admin_password`, dan `admin_status`.

g. Diagram Lain *Entity Relationship*



Gambar 4.33 (Diagram lain *Entity Relationship*)

Diagram *Entity Relationship*(E-R) dapat juga ditampilkan seperti diagram kelas agar lebih mudah untuk melihat hubungan/relasi antar entitas, namun

kekurangannya akan lebih susah untuk membaca jenis/tipe relasi yang ada pada setiap entitas.

2. Perancangan Tabel

Dari diagram E-R yang sudah rancang, tabel – tabel yang dibuat dan relasi – relasi antar tabel dapat dilihat sebagai berikut

a. Provinsi

Tabel 4.1 (Provinsi)

| Nama | Tipe | Keterangan |
|--------------------|-------------|--------------------|
| <u>Provinsi_id</u> | Int(3) | <i>Primary key</i> |
| Provinsi_kode | Int(2) | |
| Provinsi_nama | Varchar(50) | |

b. Kab_kota

Tabel 4.2 (Kab_kota)

| Nama | Tipe | Keterangan |
|--------------------|-------------|---|
| <u>Kab_kota_id</u> | Int(3) | <i>Primary key</i> |
| Provinsi_id | Int(3) | <i>Foreign key (provinsi.provinsi_id)</i> |
| Kab_kota_kode | Int(11) | |
| Kab_kota_nama | Varchar(50) | |
| Kab_kota_long | Varchar(30) | |
| Kab_kota_lat | Varchar(30) | |

c. Pemilik

Tabel 4.3 (Pemilik)

| Nama | Tipe | Keterangan |
|--------------------|--------------|--|
| <u>Pemilik_id</u> | Int(5) | <i>Primary key</i> |
| Provinsi_id | Int(3) | <i>Foreign key</i> (provinsi.provinsi_id) |
| Kab_kota_id | Int(3) | <i>Foreign key</i> (kab_kota.kab_kota_id) |
| Pemilik_nama | Varchar(30) | |
| Pemilik_email | Varchar(30) | <i>Unique</i> |
| Pemilik_password | Varchar(45) | <i>Md5</i> |
| Pemilik_alamat | Varchar(100) | |
| Pemilik_no_hp | Varchar(12) | |
| Pemilik_rumah_long | Varchar(50) | |
| Pemilik_rumah_lat | Varchar(50) | |
| Pemilik_status | Varchar(10) | |
| Pemilik_kode_aktif | Varchar(20) | |

d. Lupa_Password

Tabel 4.4 (Lupa_password)

| Nama | Tipe | Keterangan |
|-------------------------|-------------|---|
| <u>Lupa_password_id</u> | Int(5) | <i>Primary key</i> |
| Pemilik_id | Int(5) | <i>Foreign key</i> (pemilik.pemilik_id) |
| Lupa_password_key | Varchar(50) | <i>Md5</i> |

e. Indekos

Tabel 4.5 (Indekos)

| Nama | Tipe | Keterangan |
|------------------------|-------------|--|
| <u>Indekos_id</u> | Int(5) | <i>Primary_key</i> |
| Pemilik_id | Int(5) | <i>Foreign key</i> (pemilik.pemilik_id) |
| Kab_kota_id | Int(3) | <i>Foreign key</i> (kab_kota.kab_kota_id) |
| Indekos_nama | Varchar(30) | |
| Indekos_untuk | Varchar(10) | |
| Indekos_keterangan | Text | |
| Indekos_long | Varchar(50) | |
| Indekos_lat | Varchar(50) | |
| Indekos_kab_kota_jarak | Varchar(50) | Menghitung jarak indekos - kab_kota |

f. Kamar_fasilitas_int

Tabel 4.6 (Kamar_fasilitas_int)

| Nama | Tipe | Keterangan |
|------------------------|--------|--|
| Kamar_fasilitas_int_id | Int(5) | <i>Primary key</i> |
| Kamar_id | Int(5) | <i>Foreign key</i> (kamar.kamar_id) |
| Fasilitas_int_id | Int(5) | <i>Foreign key</i> (fasilitas_int . fasilitas_int_id) |

g. Fasilitas_master

Tabel 4.7 (Fasilitas_master)

| Nama | Tipe | Keterangan |
|----------------------------|-------------|--------------------|
| <u>Fasilitas_master_id</u> | Int(5) | <i>Primary key</i> |
| Fasilitas_master_nama | Varchar(20) | |
| Fasilitas_master_icon | Varchar(30) | |

h. Fasilitas_eks

Tabel 4.8 (Fasilitas_eks)

| Nama | Tipe | Keterangan |
|----------------------|-------------|--|
| <u>Fasilitas_eks</u> | Int(5) | <i>Primary key</i> |
| Fasilitas_master_id | Int(5) | <i>Foreign key</i> (fasilitas_master . fasilitas_master_id) |
| Kab_kota_id | Int(5) | <i>Foreign key</i> (kab_kota.kab_kota_id) |
| Fasilitas_eks_nama | Varchar(50) | |
| Fasilitas_eks_long | Varchar(50) | |
| Fasilitas_eks_lat | Varchar(50) | |

i. Fasilitas_int

Tabel 4.9 (Fasilitas_int)

| Nama | Tipe | Keterangan |
|--------------------|-------------|---|
| Fasilitas_int_id | Int(5) | <i>Primary key</i> |
| Pemilik_id | Int(5) | <i>Foreign key</i> (pemilik.pemilik_id) |
| Fasilitas_int_nama | Varchar(50) | |

j. Kamar

Tabel 4.10 (Kamar)

| Nama | Tipe | Keterangan |
|------------------------------|-------------|--|
| <u>Kamar_id</u> | Int(5) | <i>Primary key</i> |
| Indekos_id | Int(5) | <i>Foreign key</i> (indekos.indekos_id) |
| Kamar_nama | Varchar(30) | |
| Kamar_harga | Int(8) | |
| Kamar_isi | Int(2) | |
| Kamar_ukuran_panjang | Int(3) | |
| Kamar_ukuran_lebar | Int(3) | |
| Kamar_ukuran_jenis | Varchar(5) | |
| Kamar_foto | Varchar(40) | |
| Kamar_minimal_kontrak | Int(3) | |
| Kamar_minimal_kontrak_jenis | Varchar(5) | |
| Kamar_kontrak_status | Varchar(10) | |
| Kamar_kontrak_dari_tanggal | Date | |
| Kamar_kontrak_sampai_tanggal | Date | |

k. Indekos_fasilitas_eks

Tabel 4.11 (Indekos_fasilitas_eks)

| Nama | Tipe | Keterangan |
|-----------------------------|-------------|--|
| Indekos_fasilitas_eks_id | Int(5) | <i>Primary key</i> |
| Indekos_id | Int(5) | <i>Foreign key</i> (indekos.indekos_id) |
| Fasilitas_eks_id | Int(5) | <i>Foreign key</i> (fasilitas_eks.fasilitas_eks_id) |
| Indekos_fasilitas_eks_jarak | Varchar(50) | Menghitung jarak antara fasilitas eks – indekos. |

l. Admin

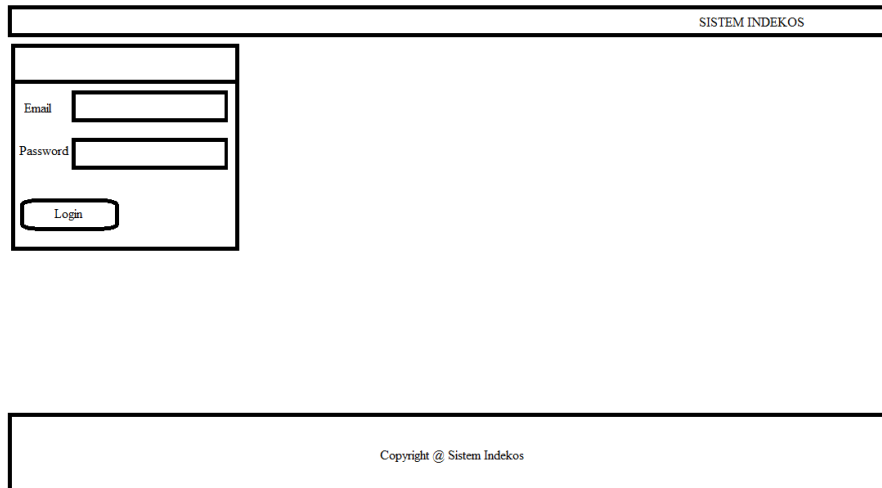
Tabel 4.12 (Admin)

| Nama | Tipe | Keterangan |
|-----------------|-------------|--------------------|
| <u>Admin_id</u> | Int(5) | <i>Primary key</i> |
| Admin_email | Varchar(30) | <i>Unique</i> |
| Admin_password | Varchar(45) | <i>Md5</i> |
| Admin_status | Varchar(10) | |

C. Perancangan Antarmuka

1. Antarmuka Admin

a. Halaman login admin

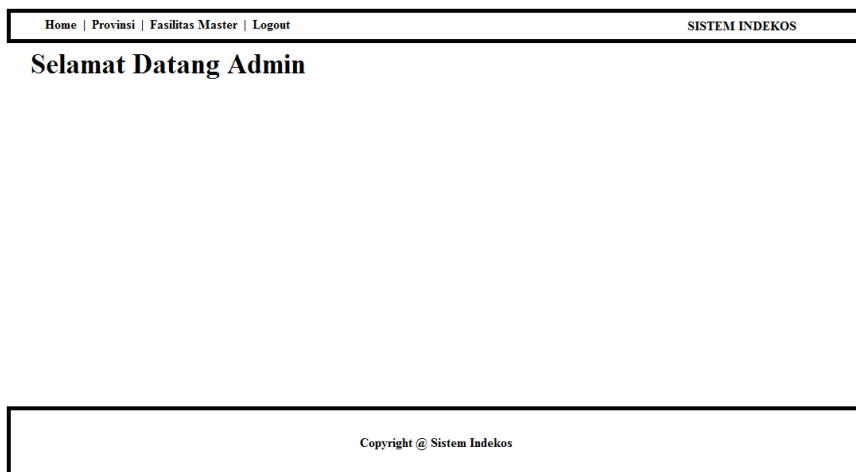


The wireframe shows a login page layout. At the top, a horizontal bar contains the text 'SISTEM INDEKOS' on the right side. Below this, on the left, is a login form. The form has two input fields: one labeled 'Email' and another labeled 'Password'. Below these fields is a button labeled 'Login'. At the bottom of the page, there is another horizontal bar containing the text 'Copyright @ Sistem Indekos'.

Gambar 4.34 (Halaman login admin)

Rancangan antarmuka untuk menu login admin diatas ini, rancangan menampilkan *form* login dan terdapat *field – field* yang dibutuhkan *email*, *password* dan tombol login dibawah untuk proses melakukan login.

b. Halaman depan (*homepage*)



The wireframe shows the homepage layout for an admin. At the top, a horizontal bar contains navigation links 'Home | Provinsi | Fasilitas Master | Logout' on the left and the text 'SISTEM INDEKOS' on the right. Below this bar, the text 'Selamat Datang Admin' is displayed. At the bottom of the page, there is another horizontal bar containing the text 'Copyright @ Sistem Indekos'.

Gambar 4.35 (Halaman depan *homepage*)

Rancangan untuk halaman depan admin dibawah ini tampil setelah admin melakukan login, dan rancangan tampilan menambahkan menu admin seperti menu *home*, provinsi, fasilitas master dan logout pada bagian atas dari desain tampilan.

c. Halaman provinsi

| | | | |
|---|--|----------------|--|
| Home Provinsi Fasilitas Master Logout | | SISTEM INDEKOS | |
|---|--|----------------|--|

Tambah Provinsi

Kode

Nama

Simpan

Provinsi

| PROVINSI KODE | PROVINSI NAMA | ACTION |
|---------------|---------------|---|
| 34 | DI YOGYAKARTA | <div style="display: flex; gap: 5px;"> <div style="border: 1px solid black; padding: 2px 5px;">Kota</div> <div style="border: 1px solid black; padding: 2px 5px;">Ubah</div> <div style="border: 1px solid black; padding: 2px 5px;">Hapus</div> </div> |
| | | |

Copyright @ Sistem Indekos

Gambar 4.36 (Halaman provinsi)

Perancangan untuk menu provinsi admin membagi dua *form* yang pertama untuk melakukan operasi tambah, dan yang lain untuk menampilkan daftar provinsi yang ada, dan tampilan membatasi daftar provinsi perhalaman ada 20 provinsi dan jika provinsi lebih dari 20 maka pada bagian bawah daftar akan tampil menu halaman (*pagination*), dan tiap – tiap daftar provinsi ditambah proses untuk melakukan operasi lain seperti lihat kota yang ada diprovinsi, ubah provinsi untuk *form* ubah rancangan tampilan tidak ada yang berubah, dan hapus provinsi.

d. Halaman kab kota

| Home Provinsi Fasilitas Master Logout | | | SISTEM INDEKOS | | | | | | | | | | | | | | | | | |
|---|------------------|---|---|--|--|------|---------------|--------|------|-------------|---|------|-------------|---|------|------------------|---|--|--|--|
| Tambah Kota Kode <input style="width: 100%;" type="text"/> Nama <input style="width: 100%;" type="text"/> Peta <input type="checkbox"/> Longitude <input type="checkbox"/> Latitude <input type="checkbox"/> <input type="button" value="Simpan"/> | | | Provinsi [NAMA PROVINSI DIPILIH] <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">KODE</th> <th style="width: 55%;">KAB KOTA NAMA</th> <th style="width: 30%;">ACTION</th> </tr> </thead> <tbody> <tr> <td>3402</td> <td>KAB. BANTUL</td> <td> <input type="button" value="Eksternal"/> <input type="button" value="Ubah"/> <input type="button" value="Hapus"/> </td> </tr> <tr> <td>3404</td> <td>KAB. SLEMAN</td> <td> <input type="button" value="Eksternal"/> <input type="button" value="Ubah"/> <input type="button" value="Hapus"/> </td> </tr> <tr> <td>3471</td> <td>KOTA. YOGYAKARTA</td> <td> <input type="button" value="Eksternal"/> <input type="button" value="Ubah"/> <input type="button" value="Hapus"/> </td> </tr> <tr> <td colspan="3" style="height: 100px;"></td> </tr> </tbody> </table> | | | KODE | KAB KOTA NAMA | ACTION | 3402 | KAB. BANTUL | <input type="button" value="Eksternal"/> <input type="button" value="Ubah"/> <input type="button" value="Hapus"/> | 3404 | KAB. SLEMAN | <input type="button" value="Eksternal"/> <input type="button" value="Ubah"/> <input type="button" value="Hapus"/> | 3471 | KOTA. YOGYAKARTA | <input type="button" value="Eksternal"/> <input type="button" value="Ubah"/> <input type="button" value="Hapus"/> | | | |
| KODE | KAB KOTA NAMA | ACTION | | | | | | | | | | | | | | | | | | |
| 3402 | KAB. BANTUL | <input type="button" value="Eksternal"/> <input type="button" value="Ubah"/> <input type="button" value="Hapus"/> | | | | | | | | | | | | | | | | | | |
| 3404 | KAB. SLEMAN | <input type="button" value="Eksternal"/> <input type="button" value="Ubah"/> <input type="button" value="Hapus"/> | | | | | | | | | | | | | | | | | | |
| 3471 | KOTA. YOGYAKARTA | <input type="button" value="Eksternal"/> <input type="button" value="Ubah"/> <input type="button" value="Hapus"/> | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | |
| Copyright @ Sistem Indekos | | | | | | | | | | | | | | | | | | | | |

Gambar 4.37 (Antarmuka halaman kab kota)

Rancangan untuk menu kab kota sama seperti rancangan provinsi, tampilan dibagi menjadi dua *form* yang pertama tambah kab kota, dan yang lain untuk menampilkan daftar kab kota yang ada pada provinsi dan setiap daftar kab kota ditambah proses eksternal ubah dan hapus.

e. Halaman fasilitas master

| Home Provinsi Fasilitas Master Logout | | | SISTEM INDEKOS | | | | | | | | | | | | | | |
|---|---------|--|--|--|--|------|------|--------|------------------------------------|--------|--|------------------------------------|---------|--|--|--|--|
| Tambah Fasilitas Master Nama <input style="width: 100%;" type="text"/> Icon <input type="button" value="Pilih gambar"/> <input type="button" value="Simpan"/> | | | Fasilitas Master <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">ICON</th> <th style="width: 55%;">NAMA</th> <th style="width: 30%;">ACTION</th> </tr> </thead> <tbody> <tr> <td><input type="button" value="IMG"/></td> <td>KAMPUS</td> <td> <input type="button" value="Ubah"/> <input type="button" value="Hapus"/> </td> </tr> <tr> <td><input type="button" value="IMG"/></td> <td>SEKOLAH</td> <td> <input type="button" value="Ubah"/> <input type="button" value="Hapus"/> </td> </tr> <tr> <td colspan="3" style="height: 100px;"></td> </tr> </tbody> </table> | | | ICON | NAMA | ACTION | <input type="button" value="IMG"/> | KAMPUS | <input type="button" value="Ubah"/> <input type="button" value="Hapus"/> | <input type="button" value="IMG"/> | SEKOLAH | <input type="button" value="Ubah"/> <input type="button" value="Hapus"/> | | | |
| ICON | NAMA | ACTION | | | | | | | | | | | | | | | |
| <input type="button" value="IMG"/> | KAMPUS | <input type="button" value="Ubah"/> <input type="button" value="Hapus"/> | | | | | | | | | | | | | | | |
| <input type="button" value="IMG"/> | SEKOLAH | <input type="button" value="Ubah"/> <input type="button" value="Hapus"/> | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | |
| Copyright @ Sistem Indekos | | | | | | | | | | | | | | | | | |

Gambar 4.38 (Halaman fasilitas master)

Rancangan untuk fasilitas master tampilan *form* ada dua yang pertama untuk melakukan operasi tambah fasilitas master dan yang lain untuk menampilkan daftar fasilitas master yang dimasukan dan pada setiap daftar ditambah proses untuk melakukan operasi ubah dan hapus fasilitas master.

f. Halaman fasilitas eksternal

| | | |
|--|--|----------------|
| Home Provinsi Fasilitas Master Logout | | SISTEM INDEKOS |
| <div style="display: flex; justify-content: space-between;"> <div style="width: 40%;"> <p>Tambah Fasilitas Eksternal</p> <div style="border: 1px solid black; padding: 5px; margin-bottom: 5px;"> Kota [NAMA KOTA DIPILIH] </div> <div style="border: 1px solid black; padding: 5px; margin-bottom: 5px;"> Master FASILITAS MASTER ▼ </div> <div style="border: 1px solid black; padding: 5px; margin-bottom: 5px;"> Nama </div> <div style="display: flex; justify-content: space-between; margin-bottom: 5px;"> <div style="border: 1px solid black; padding: 2px 5px;">Peta</div> <div style="border: 1px solid black; padding: 2px 5px;">Longitude</div> <div style="border: 1px solid black; padding: 2px 5px;">Latitude</div> </div> <div style="border: 1px solid black; padding: 5px; text-align: center;"> Simpan </div> </div> <div style="width: 55%;"> <p>[NAMA KOTA DIPILIH] Fasilitas Eksternal</p> <div style="border: 1px solid black; padding: 5px; margin-bottom: 5px;"> [NAMA FASILITAS MASTER] </div> <div style="border: 1px solid black; padding: 5px;"> [Nama fasilitas Eksternal] Ubah Hapus </div> </div> </div> | | |
| Copyright @ Sistem Indekos | | |

Gambar 4.39 (Halaman fasilitas eksternal)

Rancangan untuk fasilitas eksternal dibagi menjadi dua *form* yang pertama untuk melakukan proses tambah fasilitas eksternal dan yang lain untuk menampilkan daftar fasilitas master dan disetiap daftar terdapat daftar fasilitas eksternal yang termasuk bagian dari fasilitas eksternal dan disetiap daftar fasilitas eksternal terdapat proses ubah dan hapus fasilitas eksternal, tampilan untuk bagian admin ini tidak jauh berbeda.

2. Antarmuka Pemilik Indekos

a. Halaman login, daftar, konfirmasi email dan lupa password

The wireframe illustrates the user interface for the 'SISTEM INDEKOS' application. It features a header bar with the title 'SISTEM INDEKOS'. Below the header, there are three main sections: 'Login', 'Daftar' (Registration), and 'Kirim konfirmasi ulang' (Resend Confirmation). The 'Login' section includes fields for 'Email' and 'Password', and a 'Login' button. The 'Daftar' section includes a dropdown menu for '[PILIH PROVINSI]', fields for 'Email', 'Password', and 'Confirm Password', and a 'Daftar' button. The 'Kirim konfirmasi ulang' section includes a field for 'masukan email' and a 'Kirim' button. Below these sections, there is a footer bar with the text 'Copyright @ Sistem Indekos'.

Gambar 4.40 (Halaman login, daftar konfirmasi email dan lupa password)

Rancangan halaman awal sebelum login untuk pemilik indekos, pada rancangan diatas dibagi menjadi tiga *form* yang pertama *form* login untuk melakukan proses login bagi pemilik indekos, yang kedua *form* daftar untuk pemilik indekos yang belum mempunyai akun didalam sistem dan yang ketiga *form* dibagi menjadi dua yaitu *form* kirim konfirmasi ulang, digunakan untuk mengirim ulang *key* aktivasi ke email yang dimasukan jika pada saat daftar *key* tidak terkirim ke email pemilik indekos dan *form* lupa password digunakan pada saat pemilik indekos lupa password untuk masuk kedalam sistem indekos dan mengirimkan *key* untuk melakukan perbaikan password yang dikirim ke email yang dimasukan pada *field* yang ada pada *form* lupa password.

b. Halaman pemilik pengisian data pribadi

| SISTEM INDEKOS | | |
|---|---|--|
| <div style="border: 1px solid black; border-radius: 10px; padding: 5px; text-align: center;">SELAMAT DATANG</div> <div style="border: 1px solid black; border-radius: 10px; padding: 5px; text-align: center;">LOGOUT</div> | <p>LENGKAPI DATA PRIBADI ANDA.</p> <hr/> <div style="display: flex; justify-content: space-between;"> <div style="width: 40%;">EMAIL</div> <div style="width: 55%; border: 1px solid black; height: 20px;"></div> </div> <div style="display: flex; justify-content: space-between;"> <div style="width: 40%;">PROVINSI</div> <div style="width: 55%; border: 1px solid black; height: 20px;"></div> </div> <div style="display: flex; justify-content: space-between;"> <div style="width: 40%;">KOTA</div> <div style="width: 55%; border: 1px solid black; height: 20px;"></div> </div> <div style="display: flex; justify-content: space-between;"> <div style="width: 40%;">NAMA</div> <div style="width: 55%; border: 1px solid black; height: 20px;"></div> </div> <div style="display: flex; justify-content: space-between;"> <div style="width: 40%;">NO HP</div> <div style="width: 55%; border: 1px solid black; height: 20px;"></div> </div> <div style="display: flex; justify-content: space-between;"> <div style="width: 40%;">PETA RUMAH</div> <div style="width: 55%;"> <div style="display: flex; justify-content: space-around; margin-bottom: 5px;"> <div style="border: 1px solid black; padding: 2px 5px;">MAPS</div> <div style="border: 1px solid black; padding: 2px 5px;">LONGITUDE</div> <div style="border: 1px solid black; padding: 2px 5px;">LATITUDE</div> </div> <div style="border: 1px solid black; height: 30px;"></div> </div> </div> <div style="display: flex; justify-content: space-between;"> <div style="width: 40%;">ALAMAT</div> <div style="width: 55%; border: 1px solid black; height: 30px;"></div> </div> | |
| Copyright @ Sistem Indekos | | |

Gambar 4.41 (Halaman pemilik pengisian data pribadi)

Rancangan untuk pengisian data pribadi, pada halaman ini dibagi dua *form* yang pertama *form* menu, jika belum mengisi data pribadi menu yang tampil hanya *home* dan *logout*, dan *form* yang lain menampilkan *form* untuk melakukan pengisian data pribadi.

c. Halaman depan (*homepage*)

| SISTEM INDEKOS | |
|--|--|
| <div style="border: 1px solid black; border-radius: 10px; padding: 5px; text-align: center;">SELAMAT DATANG</div> <div style="border: 1px solid black; border-radius: 10px; padding: 5px;"> <div style="border: 1px solid black; padding: 2px; text-align: center;">DATA PRIBADI</div> <div style="border: 1px solid black; padding: 2px; text-align: center;">UBAH PASSWORD</div> <div style="border: 1px solid black; padding: 2px; text-align: center;">INDEKOS</div> <div style="border: 1px solid black; padding: 2px; text-align: center;">FASILITAS</div> <div style="border: 1px solid black; padding: 2px; text-align: center;">KONTRAK</div> <div style="border: 1px solid black; padding: 2px; text-align: center;">LOGOUT</div> </div> | <p>SELAMAT DATANG DISISTEM INDEKOS</p> <hr/> |
| Copyright @ Sistem Indekos | |

Gambar 4.42 (Halaman depan *homepage*)

Rancangan untuk halaman depan pemilik indekos setelah melakukan pengisian data pribadi, tampilan awal hanya menampilkan menu data pribadi, ubah password, indekos, fasilitas, kontrak dan logout.

d. Halaman profil

| SISTEM INDEKOS | | |
|---|---|--|
| <div style="border: 1px solid black; padding: 5px; margin-bottom: 5px; text-align: center;">SELAMAT DATANG</div> <div style="border: 1px solid black; padding: 5px;"> DATA PRIBADI UBAH PASSWORD INDEKOS FASILITAS KONTRAK LOGOUT </div> | <div style="border-bottom: 1px solid black; margin-bottom: 5px;">DATA PRIBADI</div> <div style="display: flex; margin-bottom: 5px;"> <div style="width: 30%;">EMAIL</div> <div style="flex-grow: 1;"><input style="width: 95%;" type="text"/></div> </div> <div style="display: flex; margin-bottom: 5px;"> <div style="width: 30%;">PROVINSI</div> <div style="flex-grow: 1;"><input style="width: 95%;" type="text"/></div> </div> <div style="display: flex; margin-bottom: 5px;"> <div style="width: 30%;">KOTA</div> <div style="flex-grow: 1;"><input style="width: 95%;" type="text"/></div> <div style="width: 30px; text-align: center;"><input checked="" type="checkbox"/></div> </div> <div style="display: flex; margin-bottom: 5px;"> <div style="width: 30%;">NAMA</div> <div style="flex-grow: 1;"><input style="width: 95%;" type="text"/></div> </div> <div style="display: flex; margin-bottom: 5px;"> <div style="width: 30%;">NO HP</div> <div style="flex-grow: 1;"><input style="width: 95%;" type="text"/></div> </div> <div style="display: flex; margin-bottom: 5px;"> <div style="width: 30%;">PETA RUMAH</div> <div style="flex-grow: 1; display: flex;"> <div style="border: 1px solid black; padding: 2px 5px; margin-right: 5px;">MAPS</div> <div style="border: 1px solid black; padding: 2px 5px; margin-right: 5px;">LONGITUDE</div> <div style="border: 1px solid black; padding: 2px 5px;">LATITUDE</div> </div> </div> <div style="display: flex; margin-bottom: 5px;"> <div style="width: 30%;">ALAMAT</div> <div style="flex-grow: 1;"><input style="width: 95%;" type="text"/></div> </div> | |
| Copyright @ Sistem Indekos | | |

Gambar 4.43 (Halaman profil)

Rancangan untuk halaman data pribadi/profil pemilik indekos tampilan sama seperti pada rancangan halaman pengisian data pribadi, dan yang berbeda ada pada bagian *form* menu.

e. Halaman ubah password

| SISTEM INDEKOS | | |
|---|--|--|
| <div style="border: 1px solid black; padding: 5px; margin-bottom: 5px; text-align: center;">SELAMAT DATANG</div> <div style="border: 1px solid black; padding: 5px;"> DATA PRIBADI UBAH PASSWORD INDEKOS FASILITAS KONTRAK LOGOUT </div> | <div style="border-bottom: 1px solid black; margin-bottom: 5px;">UBAH PASSWORD</div> <div style="display: flex; margin-bottom: 5px;"> <div style="width: 30%;">EMAIL</div> <div style="flex-grow: 1;"><input style="width: 95%;" type="text"/></div> </div> <div style="display: flex; margin-bottom: 5px;"> <div style="width: 30%;">PASSWORD LAMA</div> <div style="flex-grow: 1;"><input style="width: 95%;" type="text"/></div> </div> <div style="display: flex; margin-bottom: 5px;"> <div style="width: 30%;">PASSWORD BARU</div> <div style="flex-grow: 1;"><input style="width: 95%;" type="text"/></div> </div> <div style="display: flex; margin-bottom: 5px;"> <div style="width: 30%;">KONFIRMASI</div> <div style="flex-grow: 1;"><input style="width: 95%;" type="text"/></div> </div> <div style="text-align: right; margin-top: 5px;"> <div style="border: 1px solid black; padding: 2px 10px;">Simpan</div> </div> | |
| Copyright @ Sistem Indekos | | |

Gambar 4.44 (Halaman ubah password)

Rancangan untuk halaman ubah password pemilik indekos tampilan dibagi menjadi dua *form* yang pertama menu dan yang lain *form* ubah password lama ke password yang baru.

f. Halaman depan indekos

Gambar 4.45 (Halaman depan indekos)

Rancangan halaman depan indekos dibagi menjadi dua *form* yang pertama yaitu menu, dan daftar indekos yang dimiliki oleh pemilik dan pada setiap indekos terdapat proses untuk melakukan lihat, eksternal dan hapus.

g. Halaman tambah indekos

Gambar 4.46 (Halaman tambah indekos)

Rancangan halaman tambah indekos dibagi menjadi dua *form* yang pertama menu dan yang lain adalah *form* untuk melakukan proses tambah indekos, dan tampilan ini digunakan juga untuk lihat dan/atau ubah indekos.

h. Halaman kamar

| SISTEM INDEKOS | | | | | | | | | | | | | | | | | | | | | | | | |
|--|---|---|---------|---|---|--|------|-------|---------|--------|--------|---------|--------|-------|---------|--------|---|--|--------|-------|---------|---------|---|---|
| <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px; text-align: center;">SELAMAT DATANG</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">DATA PRIBADI</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">UBAH PASSWORD</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">INDEKOS</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">FASILITAS</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">KONTRAK</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">LOGOUT</div> | <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">TAMBAH KAMAR</div> <div style="border: 1px solid black; padding: 2px;">DAFTAR KAMAR</div> | <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: left;">NAMA</th> <th style="text-align: left;">HARGA</th> <th style="text-align: left;">KONTRAK</th> <th style="text-align: left;">STATUS</th> <th style="text-align: left;">ACTION</th> <th style="text-align: left;">KONTRAK</th> </tr> </thead> <tbody> <tr> <td>KAMAR1</td> <td>70000</td> <td>1 BULAN</td> <td>KOSONG</td> <td> <div style="border: 1px solid black; padding: 2px;">LIHAT</div> <div style="border: 1px solid black; padding: 2px;">FASILITAS</div> <div style="border: 1px solid black; padding: 2px;">HAPUS</div> </td> <td> <div style="border: 1px solid black; padding: 2px;">KONTRAK BARU</div> </td> </tr> <tr> <td>KAMAR2</td> <td>60000</td> <td>1 BULAN</td> <td>KONTRAK</td> <td> <div style="border: 1px solid black; padding: 2px;">LIHAT</div> <div style="border: 1px solid black; padding: 2px;">FASILITAS</div> <div style="border: 1px solid black; padding: 2px;">HAPUS</div> </td> <td> <div style="border: 1px solid black; padding: 2px;">PERPANJANG</div> <div style="border: 1px solid black; padding: 2px;">BATALL</div> </td> </tr> </tbody> </table> | | | | | NAMA | HARGA | KONTRAK | STATUS | ACTION | KONTRAK | KAMAR1 | 70000 | 1 BULAN | KOSONG | <div style="border: 1px solid black; padding: 2px;">LIHAT</div> <div style="border: 1px solid black; padding: 2px;">FASILITAS</div> <div style="border: 1px solid black; padding: 2px;">HAPUS</div> | <div style="border: 1px solid black; padding: 2px;">KONTRAK BARU</div> | KAMAR2 | 60000 | 1 BULAN | KONTRAK | <div style="border: 1px solid black; padding: 2px;">LIHAT</div> <div style="border: 1px solid black; padding: 2px;">FASILITAS</div> <div style="border: 1px solid black; padding: 2px;">HAPUS</div> | <div style="border: 1px solid black; padding: 2px;">PERPANJANG</div> <div style="border: 1px solid black; padding: 2px;">BATALL</div> |
| NAMA | HARGA | KONTRAK | STATUS | ACTION | KONTRAK | | | | | | | | | | | | | | | | | | | |
| KAMAR1 | 70000 | 1 BULAN | KOSONG | <div style="border: 1px solid black; padding: 2px;">LIHAT</div> <div style="border: 1px solid black; padding: 2px;">FASILITAS</div> <div style="border: 1px solid black; padding: 2px;">HAPUS</div> | <div style="border: 1px solid black; padding: 2px;">KONTRAK BARU</div> | | | | | | | | | | | | | | | | | | | |
| KAMAR2 | 60000 | 1 BULAN | KONTRAK | <div style="border: 1px solid black; padding: 2px;">LIHAT</div> <div style="border: 1px solid black; padding: 2px;">FASILITAS</div> <div style="border: 1px solid black; padding: 2px;">HAPUS</div> | <div style="border: 1px solid black; padding: 2px;">PERPANJANG</div> <div style="border: 1px solid black; padding: 2px;">BATALL</div> | | | | | | | | | | | | | | | | | | | |
| Copyright @ Sistem Indekos | | | | | | | | | | | | | | | | | | | | | | | | |

Gambar 4.47 (Halaman kamar)

Rancangan untuk halaman kamar ini sama seperti rancangan indekos, *form* yang pertama menu, dan yang lain daftar kamar, dan pada setiap daftar kamar memiliki proses untuk lihat, fasilitas internal dan hapus kamar, dan proses kontrak tambah, perpanjang dan batal kontrak.

i. Halaman kontrak

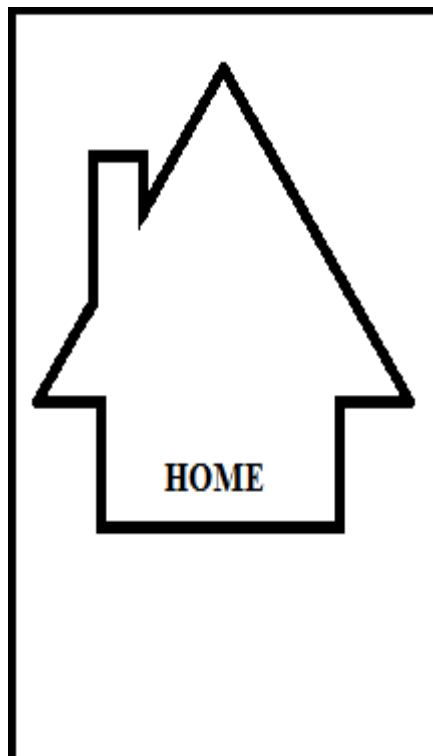
| SISTEM INDEKOS | | | | | | | | | | | | | | | |
|--|--|------------|-----------|---|--|--------------|-------|------|--------|--------|----------|--------|------------|-----------|---|
| <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px; text-align: center;">SELAMAT DATANG</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">DATA PRIBADI</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">UBAH PASSWORD</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">INDEKOS</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">FASILITAS</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">KONTRAK</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">LOGOUT</div> | <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: left;">NAMA INDEKOS</th> <th style="text-align: left;">KAMAR</th> <th style="text-align: left;">DARI</th> <th style="text-align: left;">SAMPAI</th> <th style="text-align: left;">ACTION</th> </tr> </thead> <tbody> <tr> <td>INDEKOS1</td> <td>KAMAR2</td> <td>12-12-2013</td> <td>12-4-2014</td> <td> <div style="border: 1px solid black; padding: 2px;">PERPANJANG</div> <div style="border: 1px solid black; padding: 2px;">BATALL KONTRAK</div> </td> </tr> </tbody> </table> | | | | | NAMA INDEKOS | KAMAR | DARI | SAMPAI | ACTION | INDEKOS1 | KAMAR2 | 12-12-2013 | 12-4-2014 | <div style="border: 1px solid black; padding: 2px;">PERPANJANG</div> <div style="border: 1px solid black; padding: 2px;">BATALL KONTRAK</div> |
| NAMA INDEKOS | KAMAR | DARI | SAMPAI | ACTION | | | | | | | | | | | |
| INDEKOS1 | KAMAR2 | 12-12-2013 | 12-4-2014 | <div style="border: 1px solid black; padding: 2px;">PERPANJANG</div> <div style="border: 1px solid black; padding: 2px;">BATALL KONTRAK</div> | | | | | | | | | | | |
| Copyright @ Sistem Indekos | | | | | | | | | | | | | | | |

Gambar 4.48 (Halaman kontrak)

Rancangan halaman kontrak pemilik indekos ini dibagi dua *form* yang pertama menu dan yang lain menampilkan daftar kamar pada setiap indekos yang dikontrak oleh pengguna dan terdapat proses untuk setiap kamar yang sudah dikontrak yaitu perpanjang dan batal kontrak.

3. Antarmuka Pengguna (*User*)

a. Halaman *splash screen*



Gambar 4.49 (Halaman *Splash screen*)

Rancangan halaman *splash screen* pada aplikasi pengguna, halaman *splash screen* akan ditampilkan terlebih dahulu sebelum halaman utama pada aplikasi pengguna ini dijalankan.

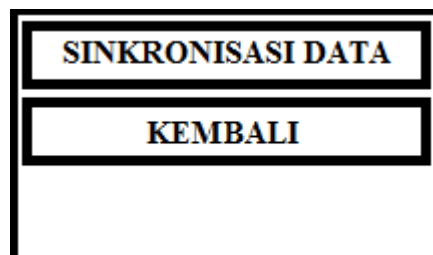
b. Halaman depan (*homepage*)



Gambar 4.50 (Halaman depan *homepage*)

Rancangan halaman depan (*homepage*) pada aplikasi pengguna, halaman ini menampilkan menu – menu yang terdapat pada aplikasi, yaitu cari indekos, indekos terdekat, sinkronisasi data dan keluar dari aplikasi

c. Halaman sinkronisasi data



Gambar 4.51 (Halaman sinkronisasi data)

Rancangan halaman sinkronisasi data pada aplikasi pengguna ini memiliki dua tombol yaitu tombol sinkronisasi data untuk mengambil data pada *server* indekos dan tombol kembali untuk kembali ke halaman depan (*homepage*).

d. Halaman cari indekos

| | |
|----------------------|------------------------|
| DI YOGAYKARTA | KAB. BANTUL |
| | KAB. SLEMAN |
| | KOTA YOGYAKARTA |
| | |

Gambar 4.52 (Halaman cari indekos)

Rancangan untuk halaman cari indekos, untuk tampilan pertama yaitu daftar provinsi, selanjutnya kab kota, daftar kategori fasilitas dan sampai daftar fasilitas eksternal rancangan tampilannya sama semua seperti rancangan tampilan diatas.

e. Halaman daftar indekos

| | |
|-----------------|--------------------|
| INDEKOS1 | untuk: LAKI |
| kamar: 2 kosong | 0 Km |
| INDEKOS2 | untuk: LAKI |
| kamar: 2 kosong | 0 Km |
| INDEKOS3 | untuk: LAKI |
| kamar: 2 kosong | 0 Km |
| | |

Gambar 4.53 (Halaman daftar indekos)

Rancangan halaman daftar indekos, pada rancangan ini menampilkan nama indekos, kamar yang kosong dan untuk siapa(Laki – laki, perempuan atau semua) dan jarak dari kab_kota/fasilitas eksternal ke indekos.

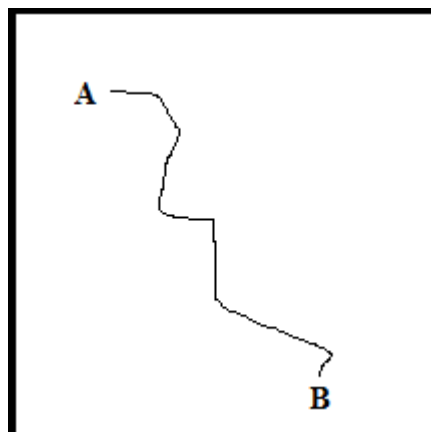
f. Halaman detail indekos

| | |
|-------------------------------------|---------------|
| NAMA INDEKOS | |
| Untuk: LAKI | |
| Keterangan | |
| <hr/> | |
| RUTE INDEKOS | |
| <hr/> | |
| KAMAR YANG MASIH KOSONG | |
| KAMARI | UKURAN: 5x5 M |
| HARGA: 50000 (1/BULAN) Isi: 1 Orang | |

Gambar 4.54 (Halaman detail indekos)

Rancangan untuk detail indekos menampilkan nama, untuk, keterangan, rute indekos, dan daftar setiap kamar yang masih kosong yang ada pada indekos yang dipilih.

g. Halaman rute indekos



Gambar 4.55 (Halaman rute indekos)

Rancangan rute indekos ini menampilkan jalan dari pengguna menuju ke indekos yang diinginkan.

h. Halaman detail kamar

| |
|---|
| NAMA KAMAR Isi : 1 Orang Ukuran : 5x5 M Harga : 50000 (1/BULAN) |
| <div>Lihat foto kamar</div> |
| FASILITAS YANG ADA DIKAMAR |
| FASILITAS 1 |
| Empty space for facilities |

Gambar 4.56 (Halaman detail kamar)

Rancangan detail kamar diatas menampilkan nama kamar, isi orang per kamar, ukuran kamar dan harga per minimal kontrak, dan menampilkan daftar fasilitas internal yang ada pada kamar yang dipilih.

BAB V

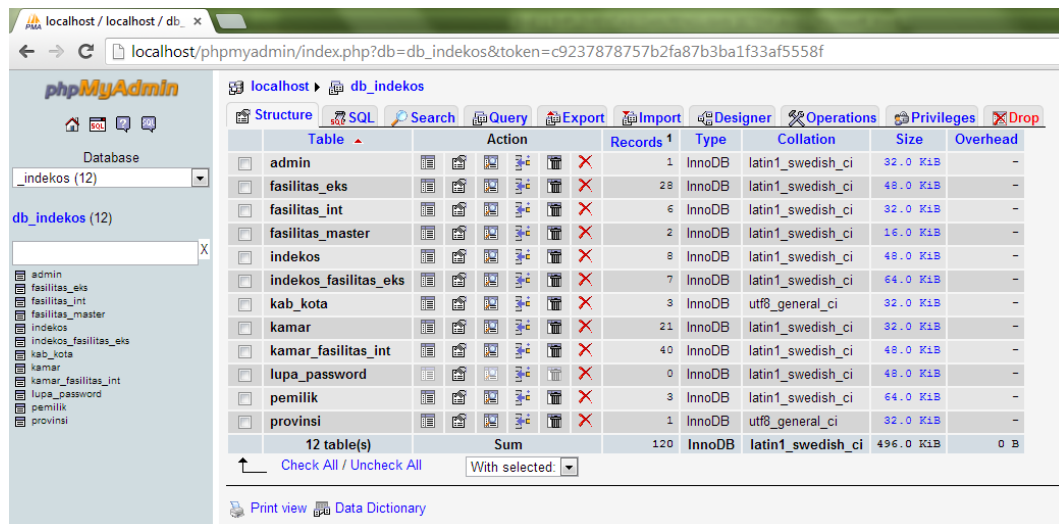
IMPLEMENTASI DAN PENGUJIAN SISTEM

A. Implementasi Sistem

Implementasi hasil dari penelitian ini dibagi menjadi tiga bagian yaitu bagian basis data menggunakan MySQL sebagai DBMS (*Database Management System*), dan bagian *website server* yang menggunakan *framework CodeIgniter* yang digunakan oleh admin dan pemilik indekos, dan bagian aplikasi *mobile client* yang diimplementasikan pada *platform* android yang digunakan oleh pengguna sistem indekos.

1. Basis data (*Database*)

a. db_indekos



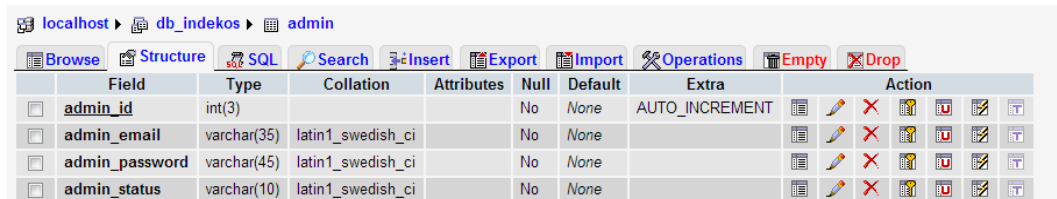
| Table | Action | Records | Type | Collation | Size | Overhead |
|-----------------------|--------|---------|--------|-------------------|---------|----------|
| admin | | 1 | InnoDB | latin1_swedish_ci | 32.0 K | - |
| fasilitas_eks | | 28 | InnoDB | latin1_swedish_ci | 48.0 K | - |
| fasilitas_int | | 6 | InnoDB | latin1_swedish_ci | 32.0 K | - |
| fasilitas_master | | 2 | InnoDB | latin1_swedish_ci | 16.0 K | - |
| indekos | | 8 | InnoDB | latin1_swedish_ci | 48.0 K | - |
| indekos_fasilitas_eks | | 7 | InnoDB | latin1_swedish_ci | 64.0 K | - |
| kab_kota | | 3 | InnoDB | utf8_general_ci | 32.0 K | - |
| kamar | | 21 | InnoDB | latin1_swedish_ci | 32.0 K | - |
| kamar_fasilitas_int | | 40 | InnoDB | latin1_swedish_ci | 48.0 K | - |
| lupa_password | | 0 | InnoDB | latin1_swedish_ci | 48.0 K | - |
| pemilik | | 3 | InnoDB | latin1_swedish_ci | 64.0 K | - |
| provinsi | | 1 | InnoDB | utf8_general_ci | 32.0 K | - |
| 12 table(s) | Sum | 120 | InnoDB | latin1_swedish_ci | 496.0 K | 0 B |

Gambar 5.1 (*Database db_indekos*)

Pada gambar diatas dapat dilihat bahwa rancangan database ini mempunyai 12 tabel yaitu admin, pemilik, provinsi, kab_kota, indekos, kamar,

fasilitas_master, fasilitas_eks, fasilitas_int, indekos_fasilitas_eks, kamar_fasilitas_int, dan lupa_password.

b. Struktur tabel admin




| Field | Type | Collation | Attributes | Null | Default | Extra | Action |
|----------------|-------------|-------------------|------------|------|---------|----------------|--------|
| admin_id | int(3) | | | No | None | AUTO_INCREMENT | |
| admin_email | varchar(35) | latin1_swedish_ci | | No | None | | |
| admin_password | varchar(45) | latin1_swedish_ci | | No | None | | |
| admin_status | varchar(10) | latin1_swedish_ci | | No | None | | |

Gambar 5.2 (Struktur tabel admin)

Tabel admin ini digunakan untuk daftar anggota admin, fungsi admin nantinya untuk memasukan data provinsi, kab kota, fasilitas eksternal dan fasilitas master yang nanti data tersebut bisa digunakan oleh pemilik indekos. Tabel ini hanya menyimpan data untuk login seperti admin_email, password, dan status.

c. Struktur tabel pemilik



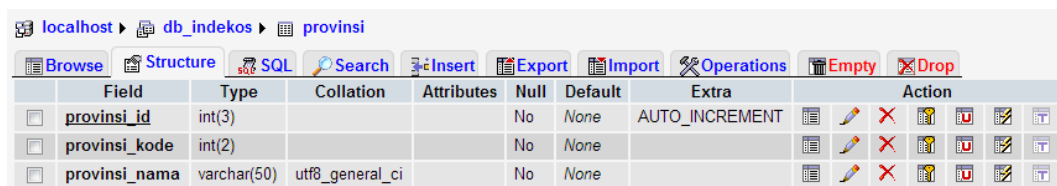
| Field | Type | Collation | Attributes | Null | Default | Extra | Action |
|--------------------|--------------|-------------------|------------|------|--------------------|----------------|--------|
| pemilik_id | int(5) | | | No | None | AUTO_INCREMENT | |
| provinsi_id | int(5) | | | No | None | | |
| kab_kota_id | int(3) | | | No | None | | |
| pemilik_nama | varchar(30) | latin1_swedish_ci | | No | None | | |
| pemilik_email | varchar(30) | latin1_swedish_ci | | No | None | | |
| pemilik_password | varchar(45) | latin1_swedish_ci | | No | None | | |
| pemilik_alamat | varchar(100) | latin1_swedish_ci | | No | None | | |
| pemilik_no_hp | varchar(12) | latin1_swedish_ci | | No | None | | |
| pemilik_rumah_long | varchar(50) | latin1_swedish_ci | | No | 119.970703125 | | |
| pemilik_rumah_lat | varchar(50) | latin1_swedish_ci | | No | -4.653079918274038 | | |
| pemilik_status | varchar(10) | latin1_swedish_ci | | No | None | | |
| pemilik_kode_aktif | varchar(20) | latin1_swedish_ci | | No | None | | |

Gambar 5.3 (Struktur tabel pemilik)

Tabel ini digunakan untuk daftar pemilik indekos yang sudah terdaftar pada sistem, tabel ini menyimpan data pribadi pemilik indekos dan memiliki relasi pada

tabel provinsi dan kab_kota sebagai kota tempat tinggalnya, dan tabel ini juga menyimpan pemilik_ nama *field* ini berisi data nama pemilik, email dan password *field* ini untuk data login pemilik, alamat , no_hp, rumah_long *field* ini untuk titik kordinat *logitude* rumah, rumah_lat *field* ini untuk titik kordinat *latitude* rumah, status *field* ini untuk bantuan login apakah pemilik sudah mengkonfirmasi pendaftarannya, dan kode_aktif *field* ini berisi kode *unique* untuk mengaktifkan status pendaftaran.

d. Struktur tabel provinsi



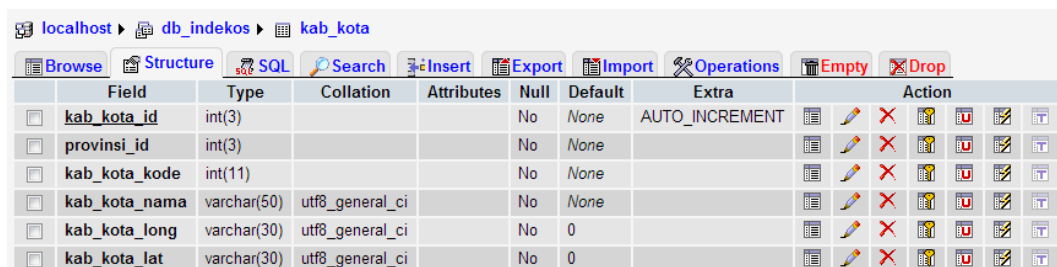
The screenshot shows the MySQL database structure for the 'provinsi' table. The table has three fields: 'provinsi_id' (int(3), AUTO_INCREMENT), 'provinsi_kode' (int(2)), and 'provinsi_nama' (varchar(50), utf8_general_ci). The 'provinsi_id' field is the primary key.

| Field | Type | Collation | Attributes | Null | Default | Extra | Action |
|--|-------------|-----------------|------------|------|---------|----------------|--------|
| <input type="checkbox"/> provinsi_id | int(3) | | | No | None | AUTO_INCREMENT | |
| <input type="checkbox"/> provinsi_kode | int(2) | | | No | None | | |
| <input type="checkbox"/> provinsi_nama | varchar(50) | utf8_general_ci | | No | None | | |

Gambar 5.4 (Struktur tabel provinsi)

Tabel provinsi ini digunakan untuk menyimpan data provinsi, *field* yang digunakan provinsi_kode dan nama.

e. Struktur tabel kab_kota



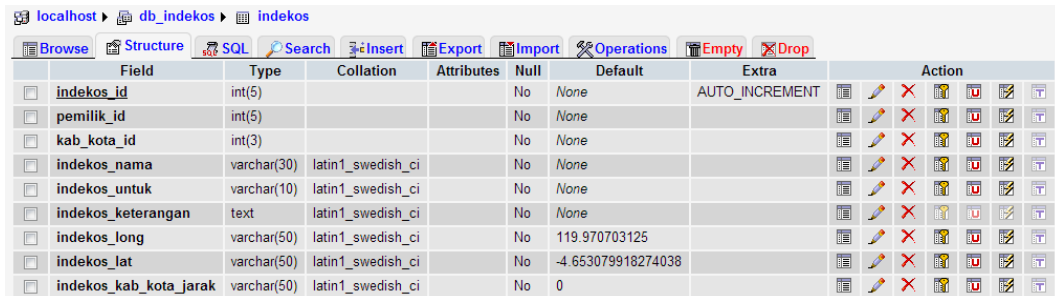
The screenshot shows the MySQL database structure for the 'kab_kota' table. The table has six fields: 'kab_kota_id' (int(3), AUTO_INCREMENT), 'provinsi_id' (int(3)), 'kab_kota_kode' (int(11)), 'kab_kota_nama' (varchar(50), utf8_general_ci), 'kab_kota_long' (varchar(30), utf8_general_ci), and 'kab_kota_lat' (varchar(30), utf8_general_ci). The 'kab_kota_id' field is the primary key.

| Field | Type | Collation | Attributes | Null | Default | Extra | Action |
|--|-------------|-----------------|------------|------|---------|----------------|--------|
| <input type="checkbox"/> kab_kota_id | int(3) | | | No | None | AUTO_INCREMENT | |
| <input type="checkbox"/> provinsi_id | int(3) | | | No | None | | |
| <input type="checkbox"/> kab_kota_kode | int(11) | | | No | None | | |
| <input type="checkbox"/> kab_kota_nama | varchar(50) | utf8_general_ci | | No | None | | |
| <input type="checkbox"/> kab_kota_long | varchar(30) | utf8_general_ci | | No | 0 | | |
| <input type="checkbox"/> kab_kota_lat | varchar(30) | utf8_general_ci | | No | 0 | | |

Gambar 5.5 (Struktur tabel kab_kota)

Tabel kota digunakan untuk menyimpan data kota dan titik kordinat dari kota tersebut. *Field* yang ada pada tabel ini kab_kota_kode, nama, long dan lat.

f. Struktur tabel indeks

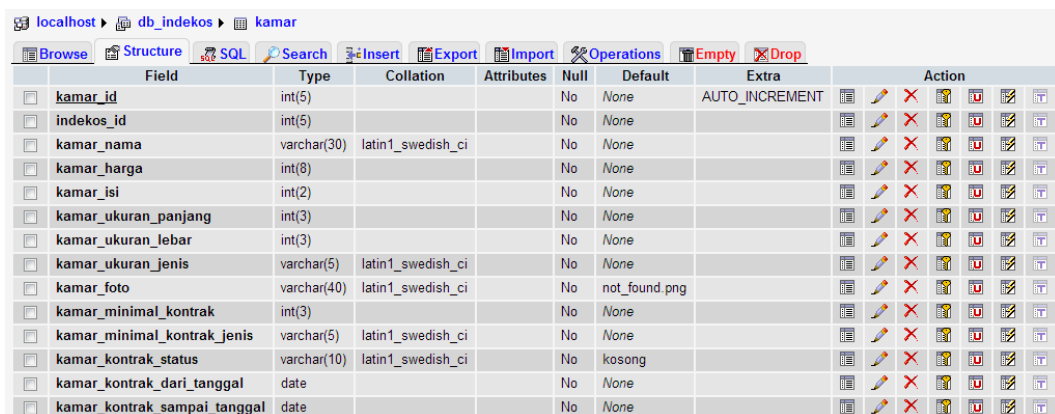


| Field | Type | Collation | Attributes | Null | Default | Extra | Action |
|---|-------------|-------------------|------------|------|--------------------|----------------|--------|
| <input type="checkbox"/> indekos_id | int(5) | | | No | None | AUTO_INCREMENT | |
| <input type="checkbox"/> pemilik_id | int(5) | | | No | None | | |
| <input type="checkbox"/> kab_kota_id | int(3) | | | No | None | | |
| <input type="checkbox"/> indekos_nama | varchar(30) | latin1_swedish_ci | | No | None | | |
| <input type="checkbox"/> indekos_untuk | varchar(10) | latin1_swedish_ci | | No | None | | |
| <input type="checkbox"/> indekos_keterangan | text | latin1_swedish_ci | | No | None | | |
| <input type="checkbox"/> indekos_long | varchar(50) | latin1_swedish_ci | | No | 119.970703125 | | |
| <input type="checkbox"/> indekos_lat | varchar(50) | latin1_swedish_ci | | No | -4.653079918274038 | | |
| <input type="checkbox"/> indekos_kab_kota_jarak | varchar(50) | latin1_swedish_ci | | No | 0 | | |

Gambar 5.6 (Struktur tabel indekos)

Tabel indekos ini untuk menyimpan data indekos bagi pemilik yang sudah mendaftar dan mengisi data indekos. *Field* yang ada pada tabel ini adalah indekos_ nama, untuk, keterangan, long, lat, dan kab_kota_jarak *field* ini untuk menghitung jarak indekos dengan kab_kota pemilik.

g. Struktur tabel kamar



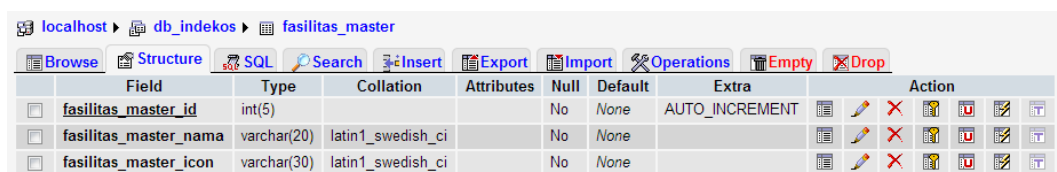
| Field | Type | Collation | Attributes | Null | Default | Extra | Action |
|---|-------------|-------------------|------------|------|---------------|----------------|--------|
| <input type="checkbox"/> kamar_id | int(5) | | | No | None | AUTO_INCREMENT | |
| <input type="checkbox"/> indekos_id | int(5) | | | No | None | | |
| <input type="checkbox"/> kamar_nama | varchar(30) | latin1_swedish_ci | | No | None | | |
| <input type="checkbox"/> kamar_harga | int(8) | | | No | None | | |
| <input type="checkbox"/> kamar_isi | int(2) | | | No | None | | |
| <input type="checkbox"/> kamar_ukuran_panjang | int(3) | | | No | None | | |
| <input type="checkbox"/> kamar_ukuran_lebar | int(3) | | | No | None | | |
| <input type="checkbox"/> kamar_ukuran_jenis | varchar(5) | latin1_swedish_ci | | No | None | | |
| <input type="checkbox"/> kamar_foto | varchar(40) | latin1_swedish_ci | | No | not_found.png | | |
| <input type="checkbox"/> kamar_minimal_kontrak | int(3) | | | No | None | | |
| <input type="checkbox"/> kamar_minimal_kontrak_jenis | varchar(5) | latin1_swedish_ci | | No | None | | |
| <input type="checkbox"/> kamar_kontrak_status | varchar(10) | latin1_swedish_ci | | No | kosong | | |
| <input type="checkbox"/> kamar_kontrak_dari_tanggal | date | | | No | None | | |
| <input type="checkbox"/> kamar_kontrak_sampai_tanggal | date | | | No | None | | |

Gambar 5.7 (Struktur tabel kamar)

Tabel kamar digunakan untuk menyimpan data kamar yang ada pada indekos, tabel ini mempunyai relasi ketabel indekos, tabel ini juga menyimpan data untuk

kontrak/sewa kamar, dan *field* yang ada pada tabel ini adalah kamar_ nama, harga, isi, ukuran_panjang, ukuran_lebar, ukuran_jenis, foto, minimal_kontrak, minimal_kontrak_jenis, kontrak_status, kontrak_dari_tanggal dan kontrak_sampai_tanggal.

h. Struktur tabel fasilitas_master

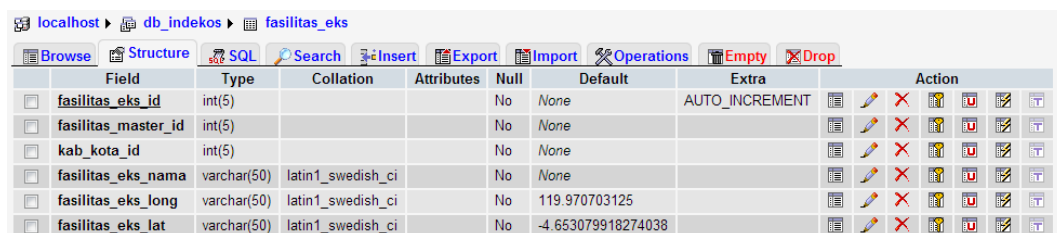


| Field | Type | Collation | Attributes | Null | Default | Extra | Action |
|-----------------------|-------------|-------------------|------------|------|---------|----------------|--------|
| fasilitas_master_id | int(5) | | | No | None | AUTO_INCREMENT | |
| fasilitas_master_nama | varchar(20) | latin1_swedish_ci | | No | None | | |
| fasilitas_master_icon | varchar(30) | latin1_swedish_ci | | No | None | | |

Gambar 5.8 (Struktur tabel fasilitas_master)

Tabel fasilitas_master ini digunakan untuk menyimpan data kategori – kategori untuk fasilitas eksternal, dan *field* yang ada pada tabel ini adalah fasilitas_master_nama dan icon.

i. Struktur tabel fasilitas_eks



| Field | Type | Collation | Attributes | Null | Default | Extra | Action |
|---------------------|-------------|-------------------|------------|------|--------------------|----------------|--------|
| fasilitas_eks_id | int(5) | | | No | None | AUTO_INCREMENT | |
| fasilitas_master_id | int(5) | | | No | None | | |
| kab_kota_id | int(5) | | | No | None | | |
| fasilitas_eks_nama | varchar(50) | latin1_swedish_ci | | No | None | | |
| fasilitas_eks_long | varchar(50) | latin1_swedish_ci | | No | 119.970703125 | | |
| fasilitas_eks_lat | varchar(50) | latin1_swedish_ci | | No | -4.653079918274038 | | |

Gambar 5.9 (Struktur tabel fasilitas_eks)

Tabel fasilitas_eks ini menyimpan data fasilitas eksternal, fasilitas eksternal adalah fasilitas yang tidak ada dalam indekos, fasilitas ini seperti tempat umum dan atau objek/tempat lainnya. Dan tabel ini mempunyai relasi pada tabel

fasilitas_master. *Field* yang ada pada tabel ini adalah fasilitas_eks_ nama, long dan lat. *Field* long dan lat ini untuk menyimpan titik kordinat dari fasilitas eksternal tersebut.

j. Struktur tabel fasilitas_int

| Field | Type | Collation | Attributes | Null | Default | Extra | Action |
|---|-------------|-------------------|------------|------|---------|----------------|--------|
| <input type="checkbox"/> fasilitas_int_id | int(5) | | | No | None | AUTO_INCREMENT | |
| <input type="checkbox"/> pemilik_id | int(5) | | | No | None | | |
| <input type="checkbox"/> fasilitas_int_nama | varchar(50) | latin1_swedish_ci | | No | None | | |

Gambar 5.10 (Struktur tabel fasilitas_int)

Tabel fasilitas_int ini untuk menyimpan fasilitas internal, fasilitas internal adalah fasilitas yang ada diindekos/kamar tersebut contohnya lemari, tempat tidur dan lainnya. Tabel ini mempunyai relasi dengan pemilik, artinya pemilik ini mempunyai fasilitas internal dan dapat digunakan untuk setiap indekos/kamar yang mempunyai fasilitas tersebut.

k. Struktur tabel indekos_fasilitas_eks

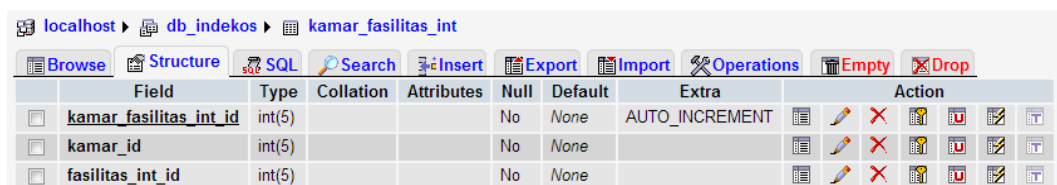
| Field | Type | Collation | Attributes | Null | Default | Extra | Action |
|--|-------------|-------------------|------------|------|---------|----------------|--------|
| <input type="checkbox"/> indekos_fasilitas_eks_id | int(5) | | | No | None | AUTO_INCREMENT | |
| <input type="checkbox"/> indekos_id | int(5) | | | No | None | | |
| <input type="checkbox"/> fasilitas_eks_id | int(5) | | | No | None | | |
| <input type="checkbox"/> indekos_fasilitas_eks_jarak | varchar(50) | latin1_swedish_ci | | No | 0 | | |

Gambar 5.11 (Struktur tabel indekos_fasilitas_eks)

Tabel indekos_fasilitas_eks ini menyimpan relasi dari tabel indekos dan fasilitas_eks, tabel ini digunakan untuk menyimpan data indekos yang mempunyai jarak dekat dengan fasilitas eksternal yang diinputkan oleh pemilik, artinya

pemilik mempunyai hak/tindakan untuk menyatakan bahwa indekos pemilik tersebut mempunyai jarak dekat dengan fasilitas eksternal yang dipilih. Dan setiap indekos dapat memiliki lebih dari satu fasilitas eksternal.

1. Struktur tabel kamar_fasilitas_int

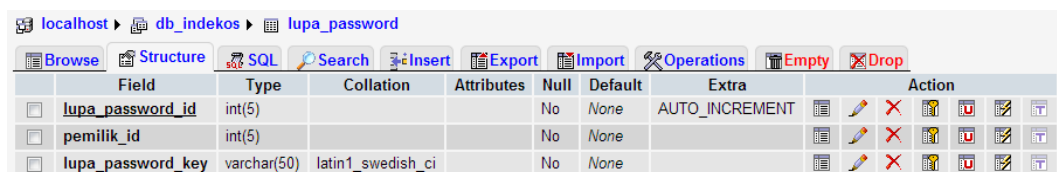


| Field | Type | Collation | Attributes | Null | Default | Extra | Action |
|---|--------|-----------|------------|------|---------|----------------|---------|
| <input type="checkbox"/> kamar_fasilitas_int_id | int(5) | | | No | None | AUTO_INCREMENT | [Icons] |
| <input type="checkbox"/> kamar_id | int(5) | | | No | None | | [Icons] |
| <input type="checkbox"/> fasilitas_int_id | int(5) | | | No | None | | [Icons] |

Gambar 5.12 (Struktur tabel kamar_fasilitas_int)

Tabel kamar_fasilitas_int ini digunakan untuk menyimpan data kamar dan fasilitas internal yang ada pada kamar tersebut, tabel ini mempunyai relasi dengan kamar dan fasilitas_int.

m. Struktur tabel lupa_password



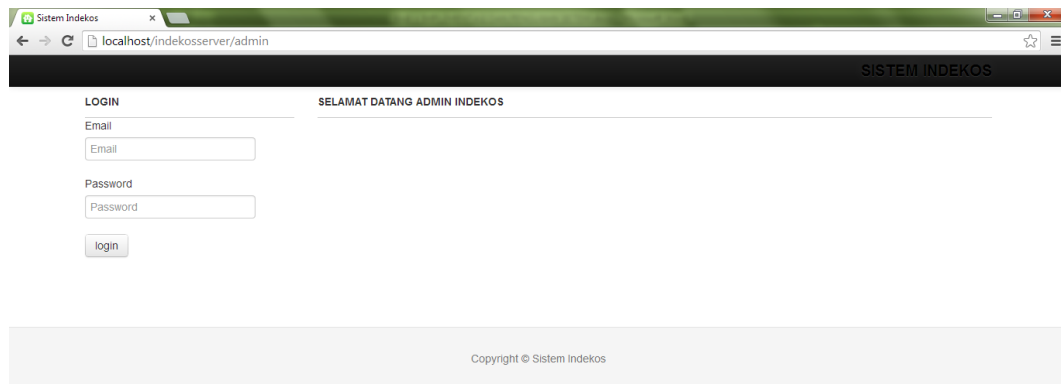
| Field | Type | Collation | Attributes | Null | Default | Extra | Action |
|--|-------------|-------------------|------------|------|---------|----------------|---------|
| <input type="checkbox"/> lupa_password_id | int(5) | | | No | None | AUTO_INCREMENT | [Icons] |
| <input type="checkbox"/> pemilik_id | int(5) | | | No | None | | [Icons] |
| <input type="checkbox"/> lupa_password_key | varchar(50) | latin1_swedish_ci | | No | None | | [Icons] |

Gambar 5.13 (Struktur tabel lupa_password)

Tabel ini digunakan untuk menyimpan data *key* pemilik indekos yang mengalami lupa *password* untuk masuk kedalam sistem, tabel ini mempunyai relasi pada tabel pemilik, dan bagi pemilik yang mengalami lupa *password* dapat memperbaiki dengan dikirim email yang berisi alamat dengan *key* yang tersimpan. Dan setelah tersimpan maka *password* akan diubah sesuai *email* yang ada.

2. Aplikasi Website server

a. Admin login

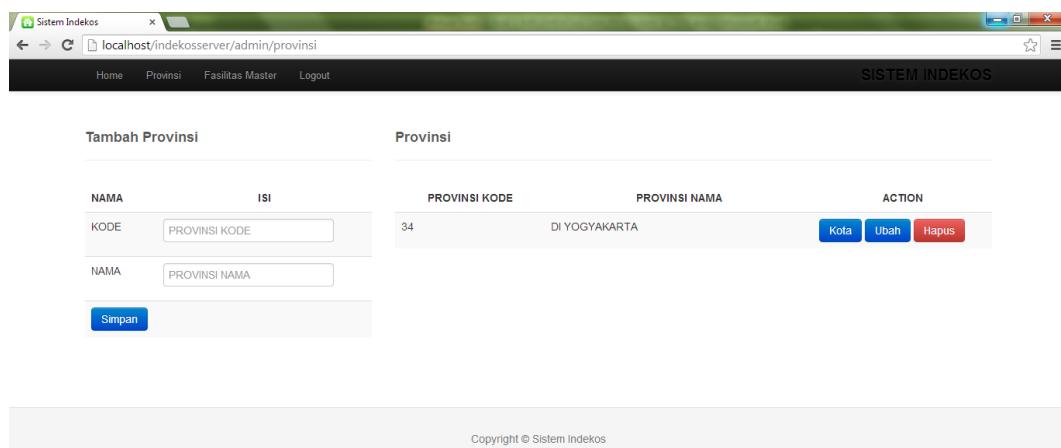


The screenshot shows a web browser window with the URL `localhost/indekossserver/admin`. The page title is "Sistem Indeksos". The header bar is dark with the text "SISTEM INDEKOS" on the right. Below the header, there is a "LOGIN" section on the left and a "SELAMAT DATANG ADMIN INDEKOS" message on the right. The login form contains two input fields: "Email" and "Password", followed by a "login" button. At the bottom, there is a footer with the text "Copyright © Sistem Indeksos".

Gambar 5.14 (Tampilan admin login)

Hasil implementasi dari rancangan halaman login untuk admin yang pada tampilan ini menampilkan *form* login yang mempunyai *field – field* yang dibutuhkan saat login kesistem admin.

b. Admin provinsi



The screenshot shows a web browser window with the URL `localhost/indekossserver/admin/provinsi`. The page title is "Sistem Indeksos". The header bar is dark with the text "SISTEM INDEKOS" on the right and navigation links "Home", "Provinsi", "Fasilitas Master", and "Logout" on the left. The main content area is divided into two sections: "Tambah Provinsi" (Add Province) and "Provinsi" (Province). The "Tambah Provinsi" section has two input fields: "KODE" (Province Code) and "NAMA" (Province Name), followed by a "Simpan" (Save) button. The "Provinsi" section displays a table of existing provinces.

| PROVINSI KODE | PROVINSI NAMA | ACTION |
|---------------|---------------|---|
| 34 | DI YOGYAKARTA | Kota Ubah Hapus |

At the bottom, there is a footer with the text "Copyright © Sistem Indeksos".

Gambar 5.15 (Tampilan admin provinsi)

Implementasi hasil dari perancangan halaman provinsi admin yang tampilan dibagi dua *form* untuk tambah dan *form* daftar provinsi yang sudah ada.

c. Admin kab kota

The screenshot shows the 'Admin kab kota' interface. On the left, there is a 'Tambah Kota' form with fields for 'NAMA' (KAB KOTA NAMA), 'ISI' (KAB KOTA KODE), and a location pin icon. Below the form is a 'Simpan' button. On the right, there is a table titled 'Provinsi DI YOGYAKARTA' with columns 'KODE', 'KAB KOTA NAMA', and 'ACTION'.

| KODE | KAB KOTA NAMA | ACTION |
|------|-----------------|----------------------|
| 3402 | KAB. BANTUL | Eksternal Ubah Hapus |
| 3404 | KAB. SLEMAN | Eksternal Ubah Hapus |
| 3471 | KOTA YOGYAKARTA | Eksternal Ubah Hapus |

Gambar 5.16 (Tampilan admin kab kota)

Implementasi hasil dari perancangan halaman kab kota, tampilan dibagi dua *form* yang pertama untuk tambah dan yang lain untuk daftar kab kota yang sudah ada pada sistem.

d. Admin fasilitas eksternal

The screenshot shows the 'Admin fasilitas eksternal' interface. On the left, there is a 'Tambah Fasilitas Eksternal' form with fields for 'NAMA' (KOTA YOGYAKARTA), 'ISI' (KOTA YOGYAKARTA), and a location pin icon. Below the form is a 'Simpan' button. On the right, there is a table titled 'KOTA YOGYAKARTA | Fasilitas Eksternal' with columns 'KAMPUS' and 'SEKOLAH'.

| KAMPUS | SEKOLAH |
|---------------|-------------|
| SMA Negeri 11 | Ubah Delete |
| SMA Negeri 21 | Ubah Delete |
| SMA Negeri 31 | Ubah Delete |
| SMA Negeri 41 | Ubah Delete |
| SMA Negeri 51 | Ubah Delete |
| SMA Negeri 61 | Ubah Delete |
| SMA Negeri 71 | Ubah Delete |

Gambar 5.17 (Tampilan admin fasilitas eksternal)

Implementasi hasil dari perancangan halaman fasilitas eksternal admin, tampilan dibagi dua *form* yang pertama untuk tambah dan yang lain untuk daftar yang sudah ada.

e. Admin fasilitas master

Gambar 5.18 (Tampilan admin fasilitas master)

Implementasi hasil perancangan untuk halaman fasilitas master dibagi menjadi dua *form* yang pertama untuk *form* tambah dan yang lain untuk daftar fasilitas master yang sudah ada.

f. Pemilik login

Gambar 5.19 (Tampilan pemilik login)

Implementasi hasil dari perancangan halaman login untuk pemilik indekos, dibagi menjadi tiga *form* yang pertama login, daftar dan *form recovery* akun.

g. Pemilik pengisian data pribadi

The screenshot shows a web browser window with the URL `localhost/indekosserver/pemilik/home`. The page title is "SISTEM INDEKOS". On the left, there is a sidebar with a "SELAMAT DATANG" button and a "LOG OUT" button. The main content area is titled "LENGKAPI DATA PRIBADI ANDA." and contains a form with the following fields:

| NAMA FIELD | ISI FIELD |
|--------------------|--|
| PEMILIK EMAIL | <input type="text" value="mfuadadib@yahoo.com"/> |
| PROVINSI | <input type="text" value="DI YOGYAKARTA"/> |
| KOTA | <input type="text" value="PILIH KOTA ANDA"/> |
| PEMILIK NAMA | <input type="text"/> |
| PEMILIK NO HP | <input type="text"/> |
| PEMILIK PETA RUMAH | <input type="text" value="119.97070312 -4.653079918"/> |

Gambar 5.20 (Tampilan pengisian data pribadi)

Implementasi hasil dari perancangan tampilan pengisian data pribadi, tampilan dibagi menjadi dua *form* yang pertama menu dan yang lain *form* untuk pengisian data pribadi pemilik indekos.

h. Pemilik lupa password

The screenshot shows a web browser window with the URL `localhost/indekosserver/pemilik/password`. The page title is "SISTEM INDEKOS". On the left, there is a sidebar with a "SELAMAT DATANG" button and a "LOG OUT" button. The main content area is titled "UBAH PASSWORD" and contains a form with the following fields:

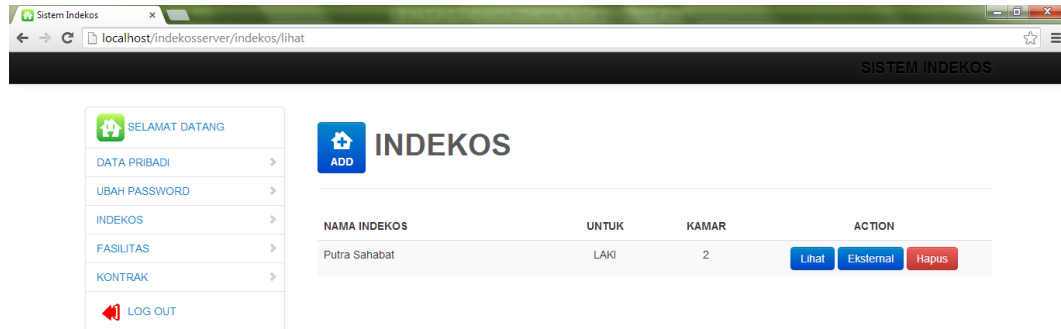
| NAMA FIELD | ISI FIELD |
|---------------------|--|
| PEMILIK EMAIL | <input type="text" value="mfuadadib@yahoo.com"/> |
| PASSWORD LAMA | <input type="text"/> |
| PASSWORD BARU | <input type="text"/> |
| PASSWORD KONFIRMASI | <input type="text"/> |

At the bottom of the form is a "SIMPAN" button.

Gambar 5.21 (Tampilan pemilik lupa password)

Implementasi hasil dari perancangan tampilan ubah password, tampilan dibagi dua *form* yang pertama menu, dan yang lain *form* untuk pengisian lupa password.

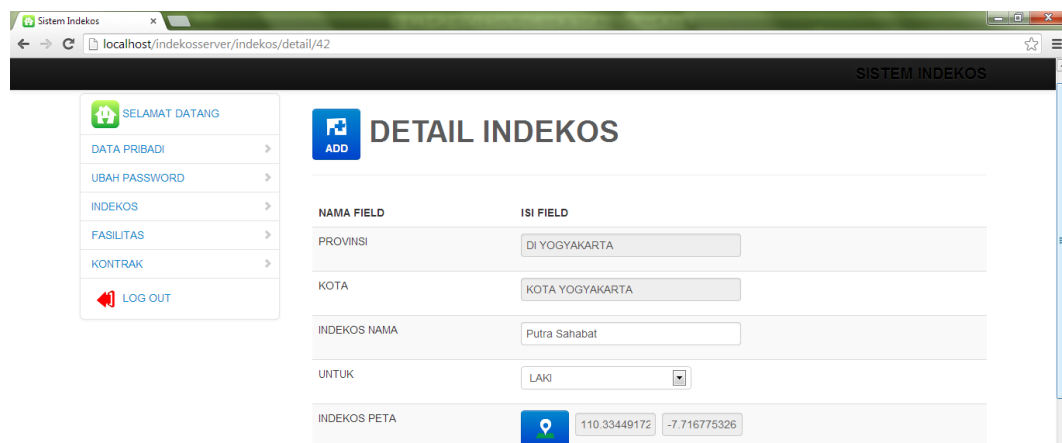
i. Pemilik indekos



Gambar 5.22 (Tampilan pemilik indekos)

Implementasi hasil dari perancangan tampilan pemilik indekos, tampilan dibagi menjadi dua *form* yang pertama untuk menu pemilik dan yang lain untuk daftar indekos yang sudah ada.

j. Pemilik indekos detail



Gambar 5.23 (Tampilan pemilik indekos detail)

Implementasi hasil rancangan untuk tampilan profil/data pribadi pemilik indekos, tampilan untuk profil/data pribadi ini sama seperti rancangan pengisian data pribadi, yang berbeda hanya pada *form* bagian menu.

k. Pemilik fasilitas eksternal

Gambar 5.24 (Tampilan pemilik fasilitas eksternal)

Implementasi hasil dari perancangan tampilan fasilitas eksternal pemilik indekos, tampilan dibagi dua *form* yang pertama menu, dan yang lain dibagi menjadi dua, yang pertama daftar fasilitas yang ada yang kedua fasilitas yang sudah dimiliki oleh indekos tersebut.

1. Pemilik kamar

| NAMA | HARGA | KONTRAK | STATUS | ACTION | KONTRAK |
|---------|-------|---------|--------|-----------------------|--------------|
| No Z001 | 56000 | 1 BULAN | KOSONG | Lihat Fasilitas Hapus | KONTRAK BARU |
| no X002 | 70000 | 1 BULAN | KOSONG | Lihat Fasilitas Hapus | KONTRAK BARU |

Gambar 5.25 (Tampilan pemilik kamar)

Implementasi hasil dari perancangan tampilan kamar indekos, *form* ini menyatu dengan *form* lihat indekos, *form* kamar ada dibagian bawah *form* lihat indekos.

m. Pemilik kamar tambah

Gambar 5.26 (Tampilan pemilik kamar tambah)

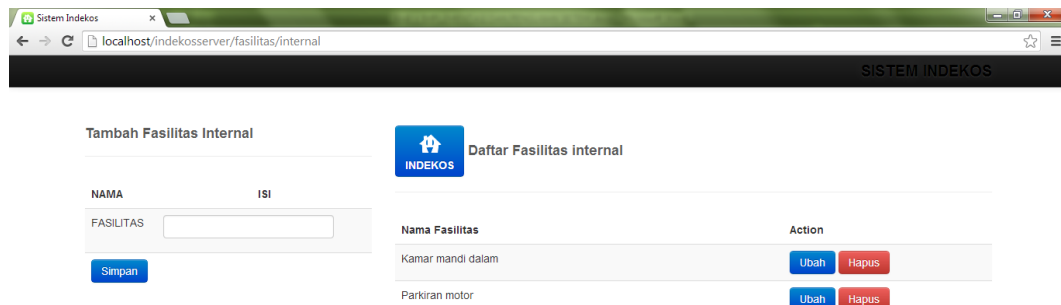
Implementasi hasil dari perancangan tambah kamar pemilik indekos, *form* ini dibagi menjadi dua yang pertama *form* menu dan yang lain *form* pengisian data kamar indekos.

n. Pemilik kamar fasilitas internal

Gambar 5.27 (Tampilan pemilik kamar fasilitas internal)

Implementasi hasil dari perancangan tampilan fasilitas internal kamar, dibagi menjadi dua *form* yang pertama dibagi menjadi dua, dibagian atas *form* tambah fasilitas internal yang baru dan yang lain daftar fasilitas yang ada, dan *form* yang kedua *form* daftar fasilitas internal yang dimiliki kamar.

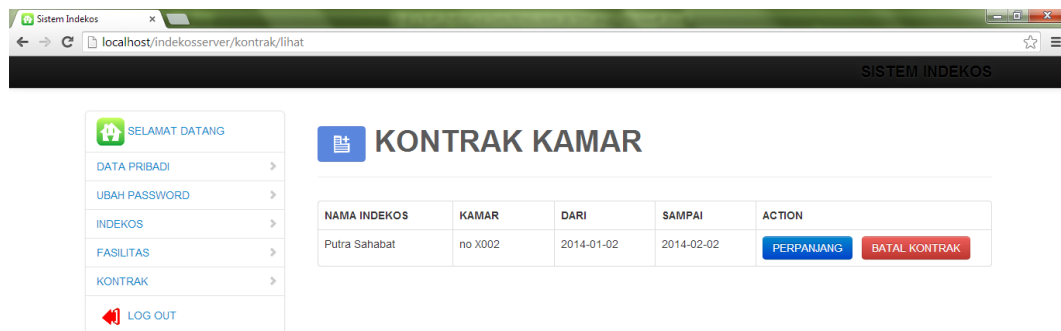
o. Pemilik fasilitas internal



Gambar 5.28 (Tampilan pemilik fasilitas internal)

Implementasi hasil dari fasilitas internal yang dimiliki oleh pemilik indekos, tampilan dibagi menjadi dua *form* yang pertama *form* tambah fasilitas internal dan yang lain daftar fasilitas internal yang dimiliki oleh pemilik indekos.

p. Pemilik kontrak



Gambar 5.29 (Tampilan pemilik kontrak)

Implementasi hasil dari perancangan halaman kontrak pemilik indekos, tampilan dibagi menjadi dua *form* yang pertama *form* menu, dan yang lain *form* daftar kamar yang sudah dikontrak.

3. Aplikasi Mobile client

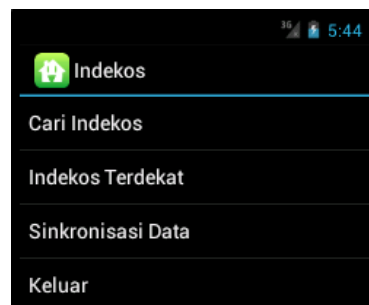
a. Pengguna *Splash Screen*



Gambar 5. 30 (Tampilan pengguna *splash screen*)

Implementasi hasil dari perancangan halaman *splash screen* untuk pengguna aplikasi indekso, halaman ini tampil pada bagian awal sebelum menu utama tampil.

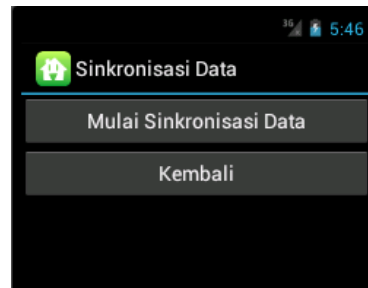
b. Pengguna halaman depan *homepage*



Gambar 5.31 (Tampilan halaman depan *homepage*)

Implementasi hasil dari perancangan halaman menu utama bagi pengguna sistem aplikasi indekos ini, menu utama ada 4 proses yaitu cari indekos, indekos terdekat, sinkronisasi data dan keluar.

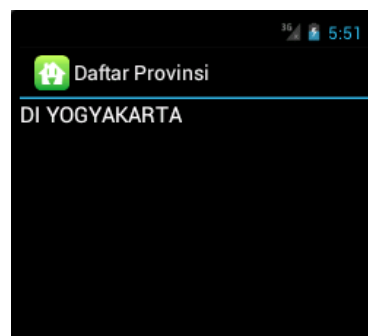
c. Pengguna sinkronisasi data



Gambar 5.32 (Tampilan sinkronisasi data)

Implementasi hasil dari perancangan tampilan sinkronisasi data bagi pengguna indekos, pada tampilan ini ada dua proses yang pertama proses sinkronisasi data, dan yang kedua proses kembali ke menu sebelumnya.

d. Pengguna daftar provinsi



Gambar 5.33 (Tampilan daftar provinsi)

Implementasi hasil dari perancangan halamana daftar provinsi bagi pengguna sistem indekos, dan pada tampilan terdapat menu kembali jika pengguna menekan tombol menu pada *smartphone* untuk kembali ke menu sebelumnya.

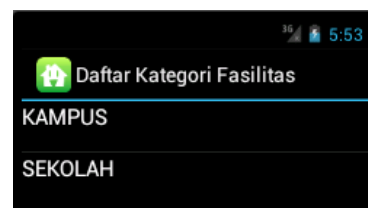
e. Pengguna daftar kab kota



Gambar 5.34 (Tampilan daftar kab kota)

Implementasi hasil dari perancangan halaman daftar kab kota untuk pengguna sistem indeks, pada tampilan ini terdapat menu kembali jika pengguna menekan tombol menu pada *smartphone* untuk kembali, dan terdapat fungsi lain fasilitas eksternal, dan jarak terdekat kab kota jika pengguna menekan lama pada daftar kab kota yang diinginkan.

f. Pengguna daftar fasilitas master



Gambar 5.35 (Tampilan daftar fasilitas master)

Implementasi hasil dari perancangan halaman kategori fasilitas eksternal untuk pengguna sistem indeks, pada tampilan ini terdapat menu kembali jika pengguna menekan tombol menu pada *smartphone* untuk kembali ke menu sebelumnya.

g. Pengguna daftar fasilitas eksternal



Gambar 5.36 (Tampilan daftar fasilitas eksternal)

Implementasi hasil dari perancangan halaman daftar fasilitas eksternal dari daftar kategori fasilitas eksternal untuk pengguna sistem indekos, pada tampilan ini terdapat menu kembali jika pengguna menekan tombol menu pada *smartphone* untuk kembali kemenu sebelumnya.

h. Pengguna daftar indekos



Gambar 5.37 (Tampilan daftar indekos)

Implementasi hasil dari perancangan halaman daftar indekos untuk pengguna sistem indekos, pada tampilan ini terdapat menu kembali jika pengguna menekan tombol menu pada *smartphone* untuk kembali, dan terdapat fungsi lain yaitu detail indekos untuk melihat keterangan tentang indekos dan melihat kamar yang masih

kosong, dan fungsi rute jalan indekos fungsi ini untuk menunjukkan rute jalan dari pengguna menuju indekos yang diinginkan, semua fungsi itu dapat dilihat jika pengguna menekan lama pada daftar indekos yang diinginkan.

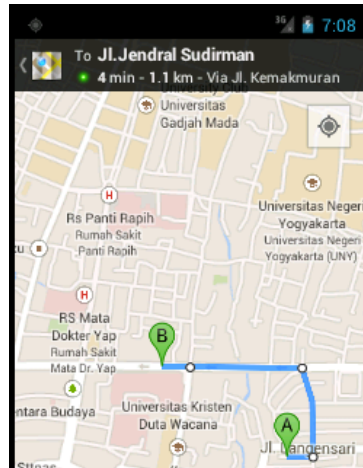
i. Pengguna detail indekos



Gambar 5.38 (Tampilan detail indekos)

Implementasi hasil dari perancangan halaman detail indekos untuk pengguna sistem indekos, pada tampilan ini halaman dibagi dua, bagian atas detail tentang indekos dan terdapat tombol untuk melihat rute jalan indekos, dan bagian lain untuk daftar kamar yang masih kosong didalam indekos yang diinginkan oleh pengguna, pada tampilan ini terdapat menu hubungi pemilik jika ingin lebih tahu tentang indekos, rute jalan indekos menu ini sama untuk menunjukkan rute jalan menuju keindekos yang diinginkan, menu tersebut dapat dilihat jika pengguna menekan tombol menu pada *smartphone*.

j. Pengguna rute indekos



Gambar 5.39 (Tampilan rute indekos)

Implementasi hasil dari perancangan halaman rute jalan menuju indekos yang diinginkan, kode “A” adalah posisi dimana pengguna berada dan kode “B” adalah posisi indekos yang diinginkan.

k. Pengguna detail kamar



Gambar 5.39 (Tampilan detail kamar)

Implementasi hasil dari perancangan halaman detail kamar untuk pengguna sistem indekos, pada tampilan ini halaman dibagi menjadi dua yang pertama detail

keterangan kamar dan pada bagian ini terdapat tombol untuk melihat foto kamar, dan yang lain daftar fasilitas internal yang ada pada kamar tersebut.

B. Pengujian Sistem

Pengujian sistem merupakan tahap terakhir pada penelitian ini, pengujian ini dilakukan untuk menguji kemampuan keseluruhan yang disediakan oleh sistem/aplikasi indekos. Pengujian yang dilakukan adalah pengujian *alpha*, sifat pengujian sistem adalah *white-box* dan *black-box* dan dilakukan pada saat pengembangan sistem. Dan pengujian *beta*, sifat pengujian adalah *black-box* dilakukan setelah sistem selesai dibangun. Teknik *white-box* merupakan metode pengujian yang memfokuskan pengujian pada fungsional sistem dan sumber kode (*source code*). Teknik *black box* merupakan metode pengujian dengan memfokuskan pada fungsional sistem yang telah dibangun serta memperhatikan hasil dari fungsional sistem tersebut apakah berjalan sesuai yang diharapkan.

1. Pengujian alpha

Pada tahap pengembangan sistem rencana pengujian alpha (*alpha testing*) yang akan dilakukan pada sistem adalah sebagai berikut

Tabel 5.1 (Rencana pengujian alpha)

| No | Item uji | Detail item | Detail pengujian |
|----|-----------------|---|--|
| 1. | Bagian pemilik | Login, kamar, indekos, pribadi, fasilitas int, fasilitas eks, kontrak | Verifikasi login, tambah, ubah, hapus, perpanjang, batal kontrak |
| 2. | Bagian pengguna | Cari indekos, sinkronisasi, rute indekos | Masukan data, alur, |

2. Pengujian beta

Rencana pengujian sistem yang akan dilakukan pada tahap pengujian beta(*beta testing*) pada sistem ini adalah sebagai berikut

a. Pemilik indekos

Tabel 5.2 (Rencana pengujian beta bagian pemilik)

| No | Item Uji | Penilaian | |
|----|---|-----------|-------|
| | | Ya | Tidak |
| 1. | Sistem dapat mengoperasikan data indekos (tambah, ubah dan hapus) | | |
| 2. | Sistem dapat mengoperasikan setiap kamar pada indekos (tambah, ubah, dan hapus) | | |
| 3. | Sistem dapat melakukan operasi waktu sewa – menyewa dan tambah kontrak, perpanjang dan hapus. | | |
| 4. | Sistem dapat menambahkan dan menghapus data fasilitas internal pada tiap kamar | | |
| 5. | Sistem dapat menambahkan dan menghapus data fasilitas eksternal pada tiap indekos | | |
| 6. | Sistem dapat melakukan operasi pada data fasilitas internal (tambah, ubah dan hapus) | | |

b. Pengguna indekos

Tabel 5.3 (Rencana pengujian beta bagian pengguna)

| No | Item uji | Penilaian | |
|----|---|-----------|-------|
| | | Ya | Tidak |
| 1. | Sistem dapat mempermudah pencarian tempat indekos | | |
| 2. | Sistem dapat melakukan sinkronisasi data indekos pada server | | |
| 3. | Sistem dapat menampilkan rute jalan menuju indekos yang dipilih | | |

C. Rencana Pengujian

Rencana untuk pengujian yang akan dilakukan untuk sistem dibagi menjadi dua yaitu rencana pengujian pada tahap alpha dan pada tahap beta rincian rencana pengujian adalah sebagai berikut

1. Pengujian Alpha

Pengujian alpha dilakukan oleh penulis sendiri, dan dilakukan pada saat pengembangan sistem berlangsung, dan jenis pengujian yang dilakukan adalah jenis *white box* dan item/komponen yang diuji adalah bagian pemilik indekos dan pengguna indekos.

Tabel 5.4 (Daftar penguji tahap alpha)

| Nama | Status | Jenis Pengujian |
|-------------|-----------|------------------|
| M Fuad Adib | Mahasiswa | <i>White box</i> |

2. Pengujian Beta

Pengujian beta dibagi dua yaitu pemilik dan pengguna indekos, untuk pemilik indekos pengujian dilakukan langsung mendatangi pemilik dan melakukan kuesioner yang dibagikan, dan bagi pengguna program diberikan secara online dan di *install* langsung di *smartphone* para penguji sistem dan kuesioner dilakukan secara online menggunakan *google drive* dengan alamat formulir <https://docs.google.com/forms/d/1K36pLJz1LHnKYsxEtQ7Bs7uD7x6zq0d18LXQoXwSIRc/viewform>.

a. Pemilik indekos

Tabel 5.5 (Daftar penguji tahap beta pemilik indekos)

| Nama | Status | Jenis Pengujian |
|-----------------------|-----------------|------------------|
| Bpk. Pardi Hadiwiyono | Pemilik Indekos | <i>Black box</i> |
| Bpk. Wongso | Pemilik Indekos | <i>Black box</i> |
| Bpk. Midin | Pemilik Indekos | <i>Black box</i> |
| Ibu. Ari Winarti | Pemilik Indekos | <i>Black box</i> |
| Bpk. Rudi harianto | Pemilik Indekos | <i>Black box</i> |

b. Pengguna Indekos

Tabel 5.6 (Daftar Penguji tahap beta pengguna indekos)

| Nama | Status | Jenis Pengujian |
|-----------------------|-----------|------------------|
| Komarudin | Pegawai | <i>Black box</i> |
| Alfian | Mahasiswa | <i>Black box</i> |
| Mujib | Mahasiswa | <i>Black box</i> |
| Habibah | Mahasiswa | <i>Black box</i> |
| Fajar | Mahasiswa | <i>Black box</i> |
| Mukhtar Halim | Mahasiswa | <i>Black box</i> |
| Laungan Nauli Siregar | Mahasiswa | <i>Black box</i> |
| Aziz Ardiansyah | Mahasiswa | <i>Black box</i> |
| Faradina | Mahasiswa | <i>Black box</i> |
| Alfi | Mahasiswa | <i>Black box</i> |

BAB VI

HASIL DAN PEMBAHASAN

A. Hasil Pengujian Sistem

Hasil pengujian pada tahap alpha dan beta yang dilakukan kepada responden yang sudah direncanakan pada saat perencanaan dengan pemilik indekos sebanyak 5 orang responden dan dengan rencanan pertanyaan yang berkaitan dengan fungsi yang ada pada sistem pemilik indekos berbasis *website*, dan dengan pengguna indekos sebanyak 10 responden dan dengan rencanan pertanyaan yang berkaitan dengan fungsi yang ada pada sistem berbasis *smartphone* android sebanyak 3 pertanyaan hasilnya dapat dilihat sebagai berikut

B. Pengujian Alpha

Tabel 6.1 (Hasil pengujian sistem tahap alpha)

| Item uji | Detail item | Detail pengujian | Hasil Pengujian |
|-----------------|---|--|-----------------|
| Bagian pemilik | Login, kamar, indekos, pribadi, fasilitas int, fasilitas eks, kontrak | Verifikasi login, tambah, ubah, hapus, perpanjang, batal kontrak | Sesuai |
| Bagian pengguna | Cari indekos, sinkronisasi, rute indekos | Masukan data, alur, | Sesuai |

Dari hasil pengujian sistem tahap alpha didapatkan hasil bahwa sistem sudah sesuai dan dapat berjalan sesuai fungsinya, dan dapat dilakukan pengujian tahap beta pada pengguna sistem.

C. Pengujian Beta

1. Hasil pengujian pemilik indekos

Tabel 6.2 (Hasil pengujian sistem pemilik indekos)

| No | Item Uji | Penilaian | |
|---------|---|-----------|-------|
| | | Ya | Tidak |
| 1. | Sistem dapat mengoperasikan data indekos (tambah, ubah dan hapus) | 5 | 0 |
| 2. | Sistem dapat mengoperasikan setiap kamar pada indekos (tambah, ubah, dan hapus) | 5 | 0 |
| 3. | Sistem dapat melakukan operasi waktu sewa – menyewa dan tambah kontrak, perpanjang dan hapus. | 5 | 0 |
| 4. | Sistem dapat menambahkan dan menghapus data fasilitas internal pada tiap kamar | 5 | 0 |
| 5. | Sistem dapat menambahkan dan menghapus data fasilitas eksternal pada tiap indekos | 5 | 0 |
| 6. | Sistem dapat melakukan operasi pada data fasilitas internal (tambah, ubah dan hapus) | 5 | 0 |
| HASIL : | | 30 | 0 |

Dari hasil pengujian sistem yang dilakukan kepada pemilik indekos didapati hasil bahwa dari 5 responden pemilik indekos sebanyak 100% menyatakan YA, dan sebanyak 0% menyatakan TIDAK.

2. Hasil pengujian pengguna indekos

Tabel 6.3 (Hasil pengujian sistem bagian pengguna)

| No | Item uji | Penilaian | |
|-------|---|-----------|-------|
| | | Ya | Tidak |
| 1. | Sistem dapat mempermudah pencarian tempat indekos | 10 | 0 |
| 2. | Sistem dapat melakukan sinkronisasi data indekos pada server | 10 | 0 |
| 3. | Sistem dapat menampilkan rute jalan menuju indekos yang dipilih | 10 | 0 |
| HASIL | | 30 | 0 |

Dari hasil pengujian yang dilakukan kepada pengguna sistem indekos di dapatkan hasil bahwa dari 10 responden pengguna sistem indekos sebanyak 100% menyatakan YA dan sebanyak 0% menyatakan TIDAK,.

Berdasarkan hasil pengujian tersebut, dapat disimpulkan bahwa sistem pencarian indekos yang dibangun menggunakan dua *platform*, yaitu *website* sebagai *server* dan android sebagai aplikasi *client* yang telah dirancang dan diimplementasikan ini layak digunakan. Akan tetapi perlu adanya pengembangan sistem yang lebih lanjut untuk mendapatkan hasil yang optimal.

BAB VII

PENUTUP

A. Kesimpulan

Berdasarkan hasil dari pengujian yang telah dilakukan penulis pada penelitian mengenai *Analisis dan Perancangan Sistem Indekos Menggunakan Metode Unified Modeling Language (UML)* ini, maka dapat diambil beberapa kesimpulan, yaitu :

1. Sistem dapat mengatur data ruangan indekos untuk pemilik indekos
2. Sistem dapat mengatur waktu sewa – menyewa ruangan dan atau perpanjang sewa bagi penyewa/pengguna dan pemilik indekos
3. Sistem dapat menjadi tempat untuk mencari indekos yang masih kosong atau masih bisa ditempati.

B. Saran

Penelitian yang dilakukan tidak terlepas dari kekurangan dan kelemahan. Oleh karena itu, masih perlu pengembangan sistem agar kinerja sistem menjadi lebih baik, antaranya :

1. Dapat menambahkan kategori untuk fasilitas master dan fasilitas eksternal agar lebih banyak pilihan tempat.
2. Membuat akun *members* untuk pengguna/pencari yang memesan/menyewa indekos agar dapat melihat kapan waktu penyewaan akan berakhir dan bisa

melakukan permintaan perpanjangan sewa secara *online* dengan menggunakan akun tersebut.

3. Dapat melakukan peringatan/*alarm* pada mobile pengguna yang sudah memiliki akun pada sistem untuk melakukan/permintaan perpanjangan sewa menyewa.
4. Dapat menambahkan diagram UML pada penelitian selanjutnya

DAFTAR PUSTAKA

- ‘Aini, Syifa Qurrotu. *Sistem Informasi Geografis Berbasis Mobile (Pemetaan Objek Wisata Religi Studi Kasus Jateng-DIY)*. Yogyakarta: Skripsi Jurusan Teknik Informatika UIN Sunan Kalijaga, 2012
- Android. (2013), *What is Android?* Dipetik Desember 31, 2013, dari <http://developer.android.com/about/index.html>
- Arief M, Rudyanto. *Pemrograman Web Dinamis menggunakan PHP dan MySQL*. Yogyakarta: Andi Offset, 2011
- Arif Huda, Akbarul. *24 Jam!! Pintar Pemrograman Android #1 Ebook Version 2.1*. Yogyakarta: Ebook 2012
- Hanif, Akhmad. *Pencarian Tempat Kos Dengan Teknologi Augmented Reality Berbasis Smartphone Android*. Yogyakarta: Skripsi Jurusan Teknik Informatika UIN Sunan Kalijaga, 2013
- Kadir, Abdul. *Dasar Perancangan dan Implementasi Database Relasional*. Yogyakarta: Andi Offset, 2009
- Kadir, Abdul. *Tuntunan Praktis Belajar Database Menggunakan MySQL*. Yogyakarta: Andi Offset, 2008
- Nugroho, Adi. *Rekayasa Perangkat Lunak Menggunakan UML dan Java*. Yogyakarta: Andi Offset, 2009
- Safaat H, Nazruddin. *Android, Pemrograman Aplikasi Mobile Smartphone dan Tablet PC Berbasis Android*. Bandung: Informatika, 2011
- Sakur, Stendy B. *PHP5 Pemrograman Berorientasi Objek Konsep & Implementasi*. Yogyakarta Andi Offset, 2010
- Saputra, Hardi. *Implementasi Global Positioning System (GPS) Untuk Pariwisata Daerah Istimewa Yogyakarta Pada Mobile Device Berbasis Android*. Yogyakarta: Skripsi UIN Sunan Kalijaga 2012
- Sholihah. *Sistem Pendukung Keputusan Untuk Pemilihan Pondok Pesantren Sebagai Tempat Tinggal Mahasiswa di D.I. Yogyakarta*. Yogyakarta: Skripsi UIN Sunan Kalijaga, 2012
- Sholihq. *Pemodelan Sistem Informasi Berorientasi Objek dengan UML*. Edisi Pertama Yogyakarta: Graha Ilmu, 2006

LAMPIRAN

A. Daftar kuesioner pemilik indekos

KUESIONER PENGUJIAN SISTEM PEMILIK INDEKOS

Nama : Pardi Hadiwiyono

Alamat : Papringan

| No | Item Uji | Penilaian | |
|----|---|-----------|-------|
| | | Ya | Tidak |
| 1. | Sistem dapat mengoperasikan data indekos (tambah, ubah dan hapus) | ✓ | |
| 2. | Sistem dapat mengoperasikan setiap kamar pada indekos (tambah, ubah, dan hapus) | ✓ | |
| 3. | Sistem dapat melakukan operasi waktu sewa – menyewa dan tambah kontrak, perpanjang dan hapus. | ✓ | |
| 4. | Sistem dapat menambahkan dan menghapus data fasilitas internal pada tiap kamar | ✓ | |
| 5. | Sistem dapat menambahkan dan menghapus data fasilitas eksternal pada tiap indekos | ✓ | |
| 6. | Sistem dapat melakukan operasi pada data fasilitas internal (tambah, ubah dan hapus) | ✓ | |

KUESIONER PENGUJIAN SISTEM PEMILIK INDEKOS

Nama : Ari Winarti

Alamat : Jl. Ori I/8 Papringan.

| No | Item Uji | Penilaian | |
|----|---|-----------|-------|
| | | Ya | Tidak |
| 1. | Sistem dapat mengoperasikan data indekos (tambah, ubah dan hapus) | ✓ | |
| 2. | Sistem dapat mengoperasikan setiap kamar pada indekos (tambah, ubah, dan hapus) | ✓ | |
| 3. | Sistem dapat melakukan operasi waktu sewa – menyewa dan tambah kontrak, perpanjang dan hapus. | ✓ | |
| 4. | Sistem dapat menambahkan dan menghapus data fasilitas internal pada tiap kamar | ✓ | |
| 5. | Sistem dapat menambahkan dan menghapus data fasilitas eksternal pada tiap indekos | ✓ | |
| 6. | Sistem dapat melakukan operasi pada data fasilitas internal (tambah, ubah dan hapus) | ✓ | |

KUESIONER PENGUJIAN SISTEM PEMILIK INDEKOS

Nama : Rudi Harianto

Alamat : Jl. Ori I/16 Papringan

| No | Item Uji | Penilaian | |
|----|---|-----------|-------|
| | | Ya | Tidak |
| 1. | Sistem dapat mengoperasikan data indekos (tambah, ubah dan hapus) | ✓ | |
| 2. | Sistem dapat mengoperasikan setiap kamar pada indekos (tambah, ubah, dan hapus) | ✓ | |
| 3. | Sistem dapat melakukan operasi waktu sewa – menyewa dan tambah kontrak, perpanjang dan hapus. | ✓ | |
| 4. | Sistem dapat menambahkan dan menghapus data fasilitas internal pada tiap kamar | ✓ | |
| 5. | Sistem dapat menambahkan dan menghapus data fasilitas eksternal pada tiap indekos | ✓ | |
| 6. | Sistem dapat melakukan operasi pada data fasilitas internal (tambah, ubah dan hapus) | ✓ | |

KUESIONER PENGUJIAN SISTEM PEMILIK INDEKOS

Nama : Midin

Alamat : Jl. Ori 1/43 papringan

| No | Item Uji | Penilaian | |
|----|---|-----------|-------|
| | | Ya | Tidak |
| 1. | Sistem dapat mengoperasikan data indekos (tambah, ubah dan hapus) | ✓ | |
| 2. | Sistem dapat mengoperasikan setiap kamar pada indekos (tambah, ubah, dan hapus) | ✓ | |
| 3. | Sistem dapat melakukan operasi waktu sewa – menyewa dan tambah kontrak, perpanjang dan hapus. | ✓ | |
| 4. | Sistem dapat menambahkan dan menghapus data fasilitas internal pada tiap kamar | ✓ | |
| 5. | Sistem dapat menambahkan dan menghapus data fasilitas eksternal pada tiap indekos | ✓ | |
| 6. | Sistem dapat melakukan operasi pada data fasilitas internal (tambah, ubah dan hapus) | ✓ | |

KUESIONER PENGUJIAN SISTEM PEMILIK INDEKOS

Nama : Wongso

Alamat : Jl. Sri 1/23 Papringan

| No | Item Uji | Penilaian | |
|----|---|-----------|-------|
| | | Ya | Tidak |
| 1. | Sistem dapat mengoperasikan data indekos (tambah, ubah dan hapus) | ✓ | |
| 2. | Sistem dapat mengoperasikan setiap kamar pada indekos (tambah, ubah, dan hapus) | ✓ | |
| 3. | Sistem dapat melakukan operasi waktu sewa – menyewa dan tambah kontrak, perpanjang dan hapus. | ✓ | |
| 4. | Sistem dapat menambahkan dan menghapus data fasilitas internal pada tiap kamar | ✓ | |
| 5. | Sistem dapat menambahkan dan menghapus data fasilitas eksternal pada tiap indekos | ✓ | |
| 6. | Sistem dapat melakukan operasi pada data fasilitas internal (tambah, ubah dan hapus) | ✓ | |

B. Daftar sekolah SMAN Daerah Istimewa Yogyakarta

| No | Nama | Kab Kota |
|----|--------------------------|----------------------|
| 1 | SMA Negeri 1 | Kotamadya Yogyakarta |
| 2 | SMA Negeri 2 | Kotamadya Yogyakarta |
| 3 | SMA Negeri 3 | Kotamadya Yogyakarta |
| 4 | SMA Negeri 4 | Kotamadya Yogyakarta |
| 5 | SMA Negeri 5 | Kotamadya Yogyakarta |
| 6 | SMA Negeri 6 | Kotamadya Yogyakarta |
| 7 | SMA Negeri 7 | Kotamadya Yogyakarta |
| 8 | SMA Negeri 8 | Kotamadya Yogyakarta |
| 9 | SMA Negeri 9 | Kotamadya Yogyakarta |
| 10 | SMA Negeri 10 | Kotamadya Yogyakarta |
| 11 | SMA Negeri 11 | Kotamadya Yogyakarta |
| 12 | SMA N 1 Cangkringan | Kabupaten Sleman |
| 13 | SMA N 1 Depok | Kabupaten Sleman |
| 14 | SMA N 1 Gamping | Kabupaten Sleman |
| 15 | SMA N 1 Godean | Kabupaten Sleman |
| 16 | SMA N Kalasan | Kabupaten Sleman |
| 17 | SMA N 1 Minggir | Kabupaten Sleman |
| 18 | SMA N 1 Mlati | Kabupaten Sleman |
| 19 | SMA N 1 Ngaglik | Kabupaten Sleman |
| 20 | SMA N 2 Ngaglik | Kabupaten Sleman |
| 21 | SMA N 1 Ngemplak | Kabupaten Sleman |
| 22 | SMA N 1 Pakem | Kabupaten Sleman |
| 23 | SMA N 1 Prambanan | Kabupaten Sleman |
| 24 | SMA N 1 Sayegan | Kabupaten Sleman |
| 25 | SMA N 1 Sleman | Kabupaten Sleman |
| 26 | SMA N 2 Sleman | Kabupaten Sleman |
| 27 | SMA N 1 Tempel | Kabupaten Sleman |
| 28 | SMA N 1 Turi | Kabupaten Sleman |
| 29 | SMA Negeri 1 Bantul | Kabupaten Bantul |
| 31 | SMA Negeri 2 Bantul | Kabupaten Bantul |
| 32 | SMA Negeri 3 Bantul | Kabupaten Bantul |
| 33 | SMA Negeri 1 Jetis | Kabupaten Bantul |
| 34 | SMA Negeri 1 Sewon | Kabupaten Bantul |
| 35 | SMA Negeri 1 Banguntapan | Kabupaten Bantul |
| 36 | SMA Negeri 2 Banguntapan | Kabupaten Bantul |

| | | |
|----|----------------------------|------------------|
| 37 | SMA Negeri 1 Bambanglipuro | Kabupaten Bantul |
| 38 | SMA Negeri 1 Imogiri | Kabupaten Bantul |
| 39 | SMA Negeri 1 Kasihan | Kabupaten Bantul |
| 40 | SMA Negeri 1 Kretek | Kabupaten Bantul |
| 41 | SMA Negeri 1 Pajangan | Kabupaten Bantul |
| 42 | SMA Negeri 1 Piyungan | Kabupaten Bantul |
| 43 | SMA Negeri 1 Pleret | Kabupaten Bantul |
| 44 | SMA Negeri 1 Pundong | Kabupaten Bantul |
| 45 | SMA Negeri 1 Sanden | Kabupaten Bantul |
| 46 | SMA Negeri 1 Sedayu | Kabupaten Bantul |
| 47 | SMA Negeri 1 Srandakan | Kabupaten Bantul |
| 48 | SMA Negeri 1 Dlingo | Kabupaten Bantul |

Sumber

http://id.wikipedia.org/wiki/Daftar_sekolah_menengah_atas_di_Yogyakarta

C. Daftar Universitas Negeri, Swasta dan Institut di Daerah Istimewa Yogyakarta

| No | Nama | Kab Kota | Kampus |
|----|--|------------|----------|
| 1 | UIN Sunan Kalijaga | Yogyakarta | Negeri |
| 2 | Universitas Gadjah Mada | Yogyakarta | Negeri |
| 3 | Universitas Negeri Yogyakarta | Yogyakarta | Negeri |
| 4 | Universitas Ahmad Dahlan | Yogyakarta | Swasta |
| 5 | Universitas Atma Jaya | Yogyakarta | Swasta |
| 6 | Universitas Cokroaminoto | Yogyakarta | Swasta |
| 7 | Universitas Dirgantara Indonesia | Yogyakarta | Swasta |
| 8 | Universitas Islam Indonesia | Yogyakarta | Swasta |
| 9 | Universitas Janabadra | Yogyakarta | Swasta |
| 10 | Universitas Kristen Duta Wacana | Yogyakarta | Swasta |
| 11 | Universitas Kristen Immanuel | Yogyakarta | Swasta |
| 12 | Universitas Muhammadiyah Yogyakarta | Yogyakarta | Swasta |
| 13 | Universitas Pembangunan Nasional Veteran | Yogyakarta | Swasta |
| 14 | Universitas PGRI Yogyakarta | Yogyakarta | Swasta |
| 15 | Universitas Proklamasi '45 | Yogyakarta | Swasta |
| 16 | Universitas Sanata Dharma | Yogyakarta | Swasta |
| 17 | Universitas Sarjanawiyata Tamansiswa | Yogyakarta | Swasta |
| 18 | Universitas Teknologi Yogyakarta | Yogyakarta | Swasta |
| 19 | Universitas Wangsa Manggala | Yogyakarta | Swasta |
| 20 | Universitas Widya Mataram | Yogyakarta | Swasta |
| 21 | IKIP PGRI Yogyakarta | Yogyakarta | Swasta |
| 22 | Institut Pertanian Intan | Yogyakarta | Institut |
| 23 | Institut Pertanian Stiper | Sleman | Institut |
| 24 | Institut Sains dan Teknologi Akprind | Yogyakarta | Institut |
| 25 | Institut Seni Indonesia | Yogyakarta | Institut |

Sumber

<http://mfatwah.wordpress.com/2011/04/17/daftar-perguruan-tinggi-negeri-dan-swasta-di-yogyakarta/>

D. Daftar potongan *Source Code* Program

```

NAMA BERKAS = database.php
$active_group = 'default';
$active_record = TRUE;

$db['default']['hostname'] = 'localhost';
$db['default']['username'] = 'root';
$db['default']['password'] = "";
$db['default']['database'] = 'db_indekos';
$db['default']['dbdriver'] = 'mysql';
$db['default']['dbprefix'] = "";
$db['default']['pconnect'] = TRUE;
$db['default']['db_debug'] = TRUE;
$db['default']['cache_on'] = FALSE;
$db['default']['cachedir'] = "";
$db['default']['char_set'] = 'utf8';
$db['default']['dbcollat'] = 'utf8_general_ci';
$db['default']['swap_pre'] = "";
$db['default']['autoinit'] = TRUE;
$db['default']['stricton'] = FALSE;

/* End of file database.php */
/* Location: ./application/config/database.php */

/=====/
NAMA BERKAS = admin.php
<?php

class Admin extends CI_Controller{

    public function cek_data(){
        if($this->session->userdata('admin_login')){
            $this->load->model('admin_m');
            $admin = $this->admin_m->cek_data($this->session-
>userdata('email'));
            if($admin->num_rows()==1){
                $data['admin'] = $admin->result();
                //$this->load->view('admin/data',$data);
                return true;
            }else{
                redirect('admin/login');
            }
        }else{

```

```

        redirect('admin/login');
    }
}

public function provinsi(){
    if($this->cek_data()){
        $this->load->model('daerah_m');
        $this->load->library('pagination');
        $prov = count($this->daerah_m->provinsi());
        $config['base_url'] = base_url().'admin/provinsi';
        $config['total_rows'] = $prov;
        $config['per_page'] = 20;

        $this->pagination->initialize($config);
        $data['page'] = $this->pagination->create_links();
        $data['provinsi'] = $this->daerah_m-
>provinsi_limit_offset($config['per_page'],$this->uri->segment(3));
        $this->load->view('admin/provinsi',$data);
    }
}

public function provinsi_tambah(){
    if($this->cek_data()){
        $this->load->model('daerah_m');
        $this->load->library('form_validation');
        $this->form_validation-
>set_rules('provinsi_kode','Kode','required|trim|numeric');
        $this->form_validation-
>set_rules('provinsi_nama','Nama','required|trim');
        if($this->form_validation->run()){
            if($this->daerah_m->provinsi_tambah()){
                echo"<script
language=\"javascript\">alert('Data berhasil
ditambah.');

```

```

        if($this->cek_data()){
            $provinsi_id = $this->uri->segment(3);
            $this->load->model('daerah_m');
            if($this->daerah_m->provinsi_hapus($provinsi_id)){
                echo"<script    language=\"javascript\">alert('Data
berhasil dihapus.');

```

```

                                if($this->daerah_m->provinsi_ubah()){
                                    echo"<script
language=\"javascript\">alert('Data                                     berhasil
diubah.');document.location=\"'".base_url()."admin/provinsi\"</script>";
                                    }else{
                                        echo"<script
language=\"javascript\">alert('Data                                     gagal
diubah.');document.location=\"'".base_url()."admin/provinsi\"</script>";
                                    }
                                }else{
                                    $this->provinsi();
                                }
                            }
                        }
                    }
                }
            }
        }
    }
}

=====//
NAMA BERKAS = android.php
<?php

mysql_connect("localhost","root","");
mysql_select_db("db_indekos");

extract($_REQUEST, EXTR_OVERWRITE);
if($act=="provinsi_id"){
    $query = mysql_query("SELECT * FROM provinsi");
    $send = "";$prov = array();
    while($data = mysql_fetch_array($query)){
        $dt = stripslashes($data['provinsi_id']);
        $send .= $dt."#";
        $prov[] =$data;
    }
    //json_encode($prov);
    echo $send;
}else if($act=="terdekatar_jarak"){
    $jarak = mysql_query("SELECT indekos_lat, indekos_long, indekos_id, (
        (acos(
            sin
((indekos_lat*pi()/180)) *
sin(($lat*pi()/180))+
cos((indekos_lat*pi()/180)) *

```



```

cos(($lat*pi()/180)) *

cos(((indekos_long - $lng)* pi()/180))
)
)*180/pi()
)*60*1.1515
) as jarak
FROM indekos ORDER BY jarak ASC");
$send = "";
while($data = mysql_fetch_array($jarak)){
    $dt = stripslashes($data['jarak']);
    $send .= $dt."#";
}
echo $send;
}
?>
/=====/
NAMA BERKAS = LocalDatabase.java
package com.clientindekos;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.SQLException;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.util.Log;

public class LocalDatabase {
    /* ==== PEMILIK ==== */
    public static final String KEY_ROWID = "_id";
    /* ==== PROVINSI ==== */
    public static final String KEY_PROVINSI_ID = "provinsi_id";
    public static final String KEY_PROVINSI_NAMA = "provinsi_nama";
    public static final String KEY_PROVINSI_KODE = "provinsi_kode";
    public static final String KEY_JARAK = "jarak";
    public static final String KEY_JUMLAH = "jumlah";

    private static final String TAG = "LocalDatabase";
    private DatabaseHelper mDbHelper;
    private SQLiteDatabase mDb;

    private static final String DATABASE_NAMA = "db_indekos";
    private static final String TABEL_PROVINSI = "provinsi";
    private static final int DATABASE_VERSI = 2;

```

```

private final Context mContext;

private static class DatabaseHelper extends SQLiteOpenHelper {

    DatabaseHelper(Context context) {
        // TODO Auto-generated constructor stub
        super(context, DATABASE_NAMA, null,
DATABASE_VERSION);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        // TODO Auto-generated method stub
        db.execSQL("CREATE TABLE provinsi(_id integer
primary key autoincrement, provinsi_id integer, provinsi_kode integer,
provinsi_nama text);");
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int
newVersion) {
        // TODO Auto-generated method stub
        Log.w(TAG, "Upgrading database dari " + oldVersion + "
ke "
+ newVersion
+ ", upgrading akan menghapus semua data
yang ada.");
        db.execSQL("Drop table if exists provinsi");
        onCreate(db);
    }

}

public LocalDatabase(Context ctx) {
    this.mContext = ctx;
}

public LocalDatabase open() throws SQLException {

    mDbHelper = new DatabaseHelper(mContext);
    mDb = mDbHelper.getWritableDatabase();
    return this;
}

public void close() {

```

```

        mDbHelper.close();
    }

    /* ==== AMBIL DATA UNTUK SINKRONISASI ==== */
    public Cursor select_all_provinsi() {
        return mDb.query(TABEL_PROVINSI, new String[] {
KEY_ROWID,
                                KEY_PROVINSI_ID, KEY_PROVINSI_KODE,
KEY_PROVINSI_NAMA }, null,
                                null, null, null, KEY_PROVINSI_NAMA);
    }
    // Update sinkronisasi data.
    public boolean updateProvinsi(String provinsi_id, String provinsi_kode,
        String provinsi_nama) {
        ContentValues args = new ContentValues();
        args.put(KEY_PROVINSI_ID, provinsi_id);
        args.put(KEY_PROVINSI_KODE, provinsi_kode);
        args.put(KEY_PROVINSI_NAMA, provinsi_nama);
        return mDb.update(TABEL_PROVINSI, args,
KEY_PROVINSI_ID + "="
                                + provinsi_id, null) > 0;
    }
    // Insert sinkronisasi data.
    public long insertProvinsi(String provinsi_id, String provinsi_kode,
        String provinsi_nama) {
        ContentValues args = new ContentValues();
        args.put(KEY_ROWID, provinsi_id);
        args.put(KEY_PROVINSI_ID, provinsi_id);
        args.put(KEY_PROVINSI_KODE, provinsi_kode);
        args.put(KEY_PROVINSI_NAMA, provinsi_nama);
        return mDb.insert(TABEL_PROVINSI, null, args);
    }
    public boolean deleteProvinsi(String provinsi_id) {
        // TODO Auto-generated method stub
        return mDb.delete(TABEL_PROVINSI, KEY_PROVINSI_ID +
"=" + provinsi_id,
                                null) > 0;
    }
    public Cursor select_provinsi(String provinsi_id) {
        // TODO Auto-generated method stub
        return mDb.query(TABEL_PROVINSI, new String[] {
KEY_ROWID,
                                KEY_PROVINSI_ID, KEY_PROVINSI_KODE,
KEY_PROVINSI_NAMA },
                                KEY_PROVINSI_ID + "=" + provinsi_id, null,
                                null, null,

```

```

        KEY_PROVINSI_NAMA);
    }
}

/=====
NAMA BERKAS = SinkronisasiActivity.java
package com.clientindekos;

import java.io.IOException;
import java.io.InputStream;
import java.util.ArrayList;

import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.client.methods.HttpRequestBase;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.message.BasicNameValuePair;

import android.app.Activity;
import android.app.AlertDialog;
import android.app.Dialog;
import android.content.DialogInterface;
import android.database.Cursor;
import android.os.AsyncTask;
import android.os.Bundle;
import android.os.Handler;
import android.os.StrictMode;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.Window;
import android.widget.Button;
import android.widget.ProgressBar;
import android.widget.TextView;

public class SinkronisasiActivity extends Activity {

    private Button mulai_sinkron, kembali;
    private LocalDatabase mDbHelper;
    ProsesSink proses;
    public Handler mHandler;
    //private String URL =
"http://www.android.daarelqurro.sch.id/android.php";
    private String URL = "http://10.0.2.2/indekosclient/android.php";
    public void onCreate(Bundle savedInstanceState) {

```

```

        super.onCreate(savedInstanceState);
        StrictMode.ThreadPolicy policy = new
StrictMode.ThreadPolicy.Builder()
        .permitAll().build();
        StrictMode.setThreadPolicy(policy);
        mDbHelper = new LocalDatabase(this);

        mDbHelper.open();
        setContentView(R.layout.sinkron);
        mulai_sinkron = (Button) findViewById(R.id.sinkron_mulai);
        mulai_sinkron.setOnClickListener(new OnClickListener() {

            public void onClick(View v) {
                // TODO Auto-generated method stub
                proses = new ProsesSink();
                proses.execute();
            }

        });
        kembali = (Button) findViewById(R.id.sinkron_kembali);
        kembali.setOnClickListener(new OnClickListener() {

            public void onClick(View v) {
                // TODO Auto-generated method stub
                finish();
            }

        });
    }

    class ProsesSink extends AsyncTask<Void, Integer, Void> {

        Dialog dialog;
        ProgressBar progressBar;
        TextView tvLoading, tvPer;
        Button btnCancel;

        protected void onPreExecute() {
            super.onPreExecute();
            dialog = new Dialog(SinkronisasiActivity.this);
            dialog.setCancelable(false);

            dialog.requestWindowFeature(Window.FEATURE_NO_TITLE);
            dialog setContentView(R.layout.proses_sink);

            progressBar = (ProgressBar)
dialog.findViewById(R.id.progressBar1);
            tvLoading = (TextView) dialog.findViewById(R.id.tv1);

```

```

tvPer = (TextView) dialog.findViewById(R.id.tvper);
btnCancel = (Button) dialog.findViewById(R.id.btncancel);

btnCancel.setOnClickListener(new OnClickListener() {

    public void onClick(View v) {
        proses.cancel(true);
        dialog.dismiss();
    }
});

dialog.show();
}

@Override
protected Void doInBackground(Void... params) {
    // TODO Auto-generated method stub
    String[] serverprovinsi_id
        = fetch(URL+"?act=provinsi_id");
    String[] serverprovinsi_kode
        = fetch(URL+"?act=provinsi_kode");
    String[] serverprovinsi_nama
        = fetch(URL+"?act=provinsi_nama");
    String[] clientprovinsi_id = selectProvinsi(1);
    if(serverprovinsi_id.length >= clientprovinsi_id.length){
        for (int i = 0; i < serverprovinsi_id.length; i++) {
            Cursor provinsi =
mDbHelper.select_provinsi(serverprovinsi_id[i]);
            if(provinsi.getCount()==0){

mDbHelper.insertProvinsi(serverprovinsi_id[i],
serverprovinsi_kode[i], serverprovinsi_nama[i]);
            }else if(provinsi.getCount()==1){

mDbHelper.updateProvinsi(serverprovinsi_id[i],
serverprovinsi_kode[i], serverprovinsi_nama[i]);
            }else{

mDbHelper.deleteProvinsi(serverprovinsi_id[i]);
mDbHelper.insertProvinsi(serverprovinsi_id[i],
serverprovinsi_kode[i], serverprovinsi_nama[i]);
            }

```

```

        }
    }else{
        for(int i=0;i<clientprovinsi_id.length;i++){

            ArrayList<NameValuePair>postParameters=new
            ArrayList<NameValuePair>();
            postParameters.add(new
            BasicNameValuePair("provinsi_id",clientprovinsi_id[i]));
            String res=null;
            try{

                res=CustomHttpClient.executeHttpPost(URL+"?act=provinsi",postParame
                ters);

                String rs=res.toString();
                rs=rs.trim();
                rs=rs.replaceAll("\\s+", "");
                if(rs.equals("0")){

                    mDbHelper.deleteProvinsi(clientprovinsi_id[i]);
                }else{

                    mDbHelper.updateProvinsi(serverprovinsi_id[i],
                    serverprovinsi_kode[i], serverprovinsi_nama[i]);
                }
            }catch(Exception e){
                e.printStackTrace();
            }
        }
    }
    return null;
}

@Override
protected void onPostExecute(Void result) {
    super.onPostExecute(result);

    dialog.dismiss();

    AlertDialog alert = new AlertDialog.Builder(
        SinkronisasiActivity.this).create();

    alert.setTitle("Sinkronisasi selesai");
    alert.setMessage("Sinkronisasi data berhasil dilakukan");
    alert.setButton("Selesai", new
    DialogInterface.OnClickListener() {

```

```

                                public void onClick(DialogInterface dialog, int
which) {
                                dialog.dismiss();

                                }
                                });
                                alert.show();
                                }
                                }

    public String LongData(String Data) {
        String LongData = "";
        for (int i = 0; i < Data.length(); i++) {
            if (Data.charAt(i) == ' ') {
                LongData += '~';
            } else {
                LongData += Data.charAt(i);
            }
        }
        return LongData;
    }

    public String[] selectProvinsi(int column) {
        Cursor provinsi = mDbHelper.select_all_provinsi();
        String result[] = new String[provinsi.getCount()];
        provinsi.moveToFirst();
        int i = 0;
        while (provinsi.isAfterLast() == false) {
            result[i++] = provinsi.getString(column);
            provinsi.moveToNext();
        }
        provinsi.close();
        return result;
    }

    public String[] fetch(String url) {
        HttpClient httpclient = new DefaultHttpClient();
        HttpRequestBase httpRequest = null;
        HttpResponse httpResponse = null;
        InputStream inputStream = null;
        String response = "";
        StringBuffer buffer = new StringBuffer();
        httpRequest = new HttpGet(url);
        try {
            httpResponse = httpclient.execute(httpRequest);

```



```

    } catch (ClientProtocolException el) {
        el.printStackTrace();
    } catch (IOException el) {
        // TODO Auto-generated catch block
        el.printStackTrace();
    }
    try {
        inputStream = httpResponse.getEntity().getContent();
    } catch (IllegalStateException el) {
        el.printStackTrace();
    } catch (IOException el) {
        el.printStackTrace();
    }

    byte[] data = new byte[512];
    int len = 0;
    try {
        while (-1 != (len = inputStream.read(data))) {
            buffer.append(new String(data, 0, len));
        }
    } catch (IOException el) {
        el.printStackTrace();
    }
    try {
        inputStream.close();
    } catch (IOException el) {
        el.printStackTrace();
    }
    response = buffer.toString();
    StringParser parser = new StringParser();
    ArrayList<Object> output = parser.Parse(response);
    Object[] Output = output.toArray();
    String[] content = new String[Output.length];
    for (int i = 0; i < content.length; i++) {
        content[i] = Output[i].toString();
    }
    return content;
}

}

/=====/
NAMA BERKAS = IndekosTerdekat.java
package com.clientindekos;

import java.io.IOException;
import java.io.InputStream;
import java.util.ArrayList;

```

```

import org.apache.http.HttpResponse;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.client.methods.HttpRequestBase;
import org.apache.http.impl.client.DefaultHttpClient;

import android.app.ListActivity;
import android.app.AlertDialog;
import android.app.ProgressDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.database.Cursor;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.ListView;
import android.widget.SimpleCursorAdapter;
import android.widget.Toast;
import android.widget.AdapterView.AdapterContextMenuInfo;

public class IndekosTerdekat extends ListActivity implements LocationListener {

    private static final int DETAIL = Menu.FIRST;
    private static final int RUTE = Menu.FIRST + 1;
    private static final int BATAL = Menu.FIRST + 2;

    private double terdekatlong, terdekatlat;
    private LocationManager locMgr;
    private LocalDatabase mDbHelper;
    private JarakTerdekat jarakTerdekat;
    //private String IndekosURL =
"http://www.android.daarelqurro.sch.id/android.php";
    private String IndekosURL = "http://10.0.2.2/indekosclient/android.php";

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.cari);

```

```

        mDbHelper = new LocalDatabase(this);
        mDbHelper.open();
        locMgr = (LocationManager)
getSystemService(LOCATION_SERVICE);
        Location loc = locMgr

        .getLastKnownLocation(LocationManager.GPS_PROVIDER);
        terdekat(loc);
        daftarTerdekat();
    }

    public void onResume() {
        super.onResume();

        locMgr.requestLocationUpdates(LocationManager.GPS_PROVIDER,
2000, 1,
            this);
    }

    public void onPause() {
        super.onPause();
        locMgr.removeUpdates(this);
    }

    public void onLocationChanged(Location loc) {
        //terdekat(loc);
    }

    public void onProviderDisabled(String provider) {
        Toast.makeText(getApplicationContext(),
            "GPS Tidak aktif, tidak bisa menentukan jarak
terdekat.",
            Toast.LENGTH_SHORT).show();
        finish();
    }

    public void onStatusChanged(String provider, int status, Bundle extras) {
    }

    public void onProviderEnabled(String provider) {
        // TODO Auto-generated method stub
    }

    private void terdekat(Location loc) {
        if(loc != null){

```

```

        registerForContextMenu(getListView());
        terdeklat = loc.getLatitude();
        terdeklong = loc.getLongitude();
        jarakTerdekat = new JarakTerdekat();
        jarakTerdekat.execute();
        //daftarTerdekat();
    }
}

private void daftarTerdekat() {
    Cursor jrk = mDbHelper.get_jarak_terdekat();
    startManagingCursor(jrk);
    String[] from = new String[] {
        LocalDatabase.KEY_INDEKOS_NAMA,
        LocalDatabase.KEY_JARAK,
        LocalDatabase.KEY_INDEKOS_UNTUK };
    int[] to = new int[] { R.id.tvlistindekosnama,
        R.id.tvlistindekosjarak,
        R.id.tvlistindekosuntuk };

    SimpleCursorAdapter adapter = new SimpleCursorAdapter(this,
        R.layout.tvlistindekos, jrk, from, to);
    setListAdapter(adapter);
}

private class JarakTerdekat extends AsyncTask<Void, Integer, Void> {

    private ProgressDialog Dialog;

    protected void onPreExecute() {
        Dialog = new ProgressDialog(IndekosTerdekat.this);
        Dialog.setMessage("Loading terdekat");
        Dialog.show();
    }

    @Override
    protected Void doInBackground(Void... params) {
        // TODO Auto-generated method stub
        String[] serverterdekat_indekos_id = fetch(IndekosURL
            + "?act=terdekat_indekos_id&lat=" +
            terdeklat + "&lng="
            + terdeklong);
        String[] serverterdekat_jarak = fetch(IndekosURL
            + "?act=terdekat_jarak&lat=" + terdeklat +
            "&lng="
            + terdeklong);
    }
}

```

```

String[] clientterdekat = selectHasilTerdekat(1);
Log.d("LATLNG", terdeklat + " = " + terdekatlong);
if (clientterdekat.length != 0) {
    for (int i = 0; i < clientterdekat.length; i++) {
        Log.d("Hapus", ">>>>>");

        mDbHelper.deleteJarakTerdekat(clientterdekat[i]);
    }
    for (int j = 0; j < serverterdekat_indekos_id.length;
j++) {
        Log.d("LATLNG", terdeklat + " = " +
terdekatlong);
        Log.d("Insert", "indekos id ="
+
serverterdekat_indekos_id[j] + ", jarak ="
+ serverterdekat_jarak[j]);

        mDbHelper.insertJarakTerdekat(serverterdekat_indekos_id[j],
serverterdekat_jarak[j]);
    }
} else {
    for (int j = 0; j < serverterdekat_indekos_id.length;
j++) {
        Log.d("LATLNG", terdeklat + " = " +
terdekatlong);
        Log.d("Insert", "indekos id ="
+
serverterdekat_indekos_id[j] + ", jarak ="
+ serverterdekat_jarak[j]);

        mDbHelper.insertJarakTerdekat(serverterdekat_indekos_id[j],
serverterdekat_jarak[j]);
    }
}
return null;
}

protected void onPostExecute(Void result) {
    super.onPostExecute(result);
    Dialog.dismiss();
    AlertDialog alert = new
AlertDialog.Builder(IndekosTerdekat.this)
.create();

    alert.setTitle("Jarak Terdekat");

```

```

        alert.setMessage("Daftar jarak terdekat");
        alert.setButton("Lihat",
DialogInterface.OnClickListener() {
new
        public void onClick(DialogInterface dialog, int
which) {
            // TODO Auto-generated method stub
            Dialog.dismiss();
            setContentView(R.layout.cari);
            daftarTerdekat();
        }
    });
    alert.show();
}

}

public String[] selectHasilTerdekat(int column) {
    Cursor terdekat = mDbHelper.select_all_jarak_terdekat();
    String result[] = new String[terdekat.getCount()];
    terdekat.moveToFirst();
    int i = 0;
    while (terdekat.isAfterLast() == false) {
        result[i++] = terdekat.getString(column);
        terdekat.moveToNext();
    }
    terdekat.close();
    return result;
}

public String[] fetch(String url) {
    HttpClient httpclient = new DefaultHttpClient();
    HttpRequestBase httpRequest = null;
    HttpResponse httpResponse = null;
    InputStream inputStream = null;
    String response = "";
    StringBuffer buffer = new StringBuffer();
    httpRequest = new HttpGet(url);
    try {
        httpResponse = httpclient.execute(httpRequest);
    } catch (ClientProtocolException el) {
        el.printStackTrace();
    } catch (IOException el) {
        // TODO Auto-generated catch block
        el.printStackTrace();
    }
}

```

```

try {
    inputStream = httpResponse.getEntity().getContent();
} catch (IllegalStateException el) {
    el.printStackTrace();
} catch (IOException el) {
    el.printStackTrace();
}

byte[] data = new byte[512];
int len = 0;
try {
    while (-1 != (len = inputStream.read(data))) {
        buffer.append(new String(data, 0, len));
    }
} catch (IOException el) {
    el.printStackTrace();
}
try {
    inputStream.close();
} catch (IOException el) {
    el.printStackTrace();
}
response = buffer.toString();
StringParser parser = new StringParser();
ArrayList<Object> output = parser.Parse(response);
Object[] Output = output.toArray();
String[] content = new String[Output.length];
for (int i = 0; i < content.length; i++) {
    content[i] = Output[i].toString();
}
return content;
}

public boolean onContextItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case DETAIL:
            AdapterContextMenuInfo detailindekos =
(AdapterContextMenuInfo) item
                .getMenuInfo();
            Intent i = new Intent(this, DetailIndekos.class);
            i.putExtra(LocalDatabase.KEY_INDEKOS_ID,
detailindekos.id);
            startActivity(i);
            return true;
        case RUTE:

```

```

        AdapterContextMenuInfo ruteindekos =
(AdapterContextMenuInfo) item
        .getMenuInfo();
        i = new Intent(this, RuteIndekos.class);
        i.putExtra(LocalDatabase.KEY_INDEKOS_ID,
ruteindekos.id);
        startActivity(i);
        return true;
    case BATAL:
        return true;
    }
    return super.onContextItemSelected(item);
}

protected void onListItemClick(ListView l, View v, int position, long id) {
    super.onListItemClick(l, v, position, id);
    try {
        Log.d("ID INDEKOS", String.valueOf(id).toString());
        Intent i = new Intent(this, DetailIndekos.class);
        i.putExtra(LocalDatabase.KEY_INDEKOS_ID, id);
        startActivity(i);
    } catch (Exception e) {
        e.printStackTrace();
    }
}

public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.terdekat_kembali, menu);
    return true;
}

public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case R.id.menu_terdekat:
            // Location loc;
            locMgr = (LocationManager)
getSystemService(LOCATION_SERVICE);
            Location loc = locMgr

            .getLastKnownLocation(LocationManager.GPS_PROVIDER);
            terdekat(loc);
            break;
        case R.id.menu_kembali:
            finish();
            break;
    }
}

```



```

        }
        return false;
    }

}

/=====/
NAMA BERKAS = RuteIndekos.java
package com.clientindekos;

import android.app.Activity;
import android.content.Intent;
import android.database.Cursor;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.net.Uri;
import android.os.Bundle;
import android.util.Log;
import android.webkit.WebView;
import android.widget.Toast;

public class RuteIndekos extends Activity implements LocationListener {

    private Long indekos_id;
    LocalDatabase mDbHelper;
    private LocationManager locMgr;
    private String lt,lg, lt1,lg1;
    private WebView webrute;
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.ruteindekos);
        mDbHelper = new LocalDatabase(this);
        mDbHelper.open();
        locMgr
(LocationManager)getSystemService(LOCATION_SERVICE);
        Location loc
locMgr.getLastKnownLocation(LocationManager.GPS_PROVIDER);
        indekos_id = savedInstanceState != null ? savedInstanceState
.getLong(LocalDatabase.KEY_INDEKOS_ID)
:
null;

        if (indekos_id == null) {
            Bundle extras = getIntent().getExtras();
            indekos_id = extras != null ? extras
.getLong(LocalDatabase.KEY_INDEKOS_ID) : null;

```

```

        }
        ruteIndekos(loc);
    }

    private void detailIndekos() {
        // TODO Auto-generated method stub
        if (indekos_id != null) {
            Cursor detail =
mDbHelper.get_detail_indekos(indekos_id);
            startManagingCursor(detail);
            lg = detail.getString(detail

                .getColumnIndexOrThrow(LocalDatabase.KEY_INDEKOS_LONG)).toSt
ring();

            lt = detail.getString(detail

                .getColumnIndexOrThrow(LocalDatabase.KEY_INDEKOS_LAT)).toStri
ng();
        }
    }

    public void onResume(){
        super.onResume();

        locMgr.requestLocationUpdates(LocationManager.GPS_PROVIDER,
2000, 1, this);
    }

    public void onPause(){
        super.onPause();
        locMgr.removeUpdates(this);
    }

    public void onLocationChanged(Location loc){
        ruteIndekos(loc);
    }

    public void onProviderDisabled(String provider){
        Toast.makeText(getApplicationContext(), "GPS Tidak aktif, tidak
bisa melihat rute jalan indekos", Toast.LENGTH_SHORT).show();
        finish();
    }

    public void onStatusChanged(String provider, int status, Bundle extras){
    }

```

```

private void ruteIndekos(Location loc){
    detailIndekos();
    if(loc!= null){
        lt1 = String.valueOf(loc.getLatitude()).toString();
        lg1 = String.valueOf(loc.getLongitude()).toString();

        webrute = (WebView)findViewById(R.id.webrute);
        webrute.getSettings().setJavaScriptEnabled(true);
        Log.d("Link",
"http://maps.google.com/maps?saddr="+lt1+", "+lg1+"&daddr="+lt+", "+lg);
        String uri =
"http://maps.google.com/maps?saddr="+lt1+", "+lg1+"&daddr="+lt+", "+lg;
        Intent intent = new
Intent(android.content.Intent.ACTION_VIEW, Uri.parse(uri));
        intent.setClassName("com.google.android.apps.maps",
"com.google.android.maps.MapActivity");
        startActivity(intent);

        //webrute.loadUrl("http://maps.google.com/maps?saddr="+lt+", "+lg+"&da
daddr="+lt1+", "+lg1);
    }
}

public void onProviderEnabled(String provider) {
    // TODO Auto-generated method stub
}
}

```